PROJECT REPORT (Simple Mail Transfer Protocol)

CMPE 208: NETWORK ARCHITECTURE AND PROTOCOL

Prof. Shai Silberman

Submitted by

Team 12

Team Members

Akash Kumar Athghara	Kshama Shalini	Gunveet Singh Arora	Sashank Malladi
akash.athghara@sjsu.edu	kshama.shalini@sjsu.edu	gunveetsingh.arora@sjsu.edu	sashank.malladi@sjsu.edu
010757929	010763792	010641904	010466651

CONTRIBUTION BY EACH TEAM MEMBER

Akash Kumar Athghara: Environment Setup, Overview/Contents, Handshake, Standards, Extended implementation and Security issues.

Gunveet Singh Arora: Protocol components, SMTP usefulness, SMTP deduction, Conclusion, References.

Kshama Shalini: Introduction, Message Formats, Packet Header, Sendmail.

Sashank Malladi: QoS, Sessions, Modeling, SSMTP.

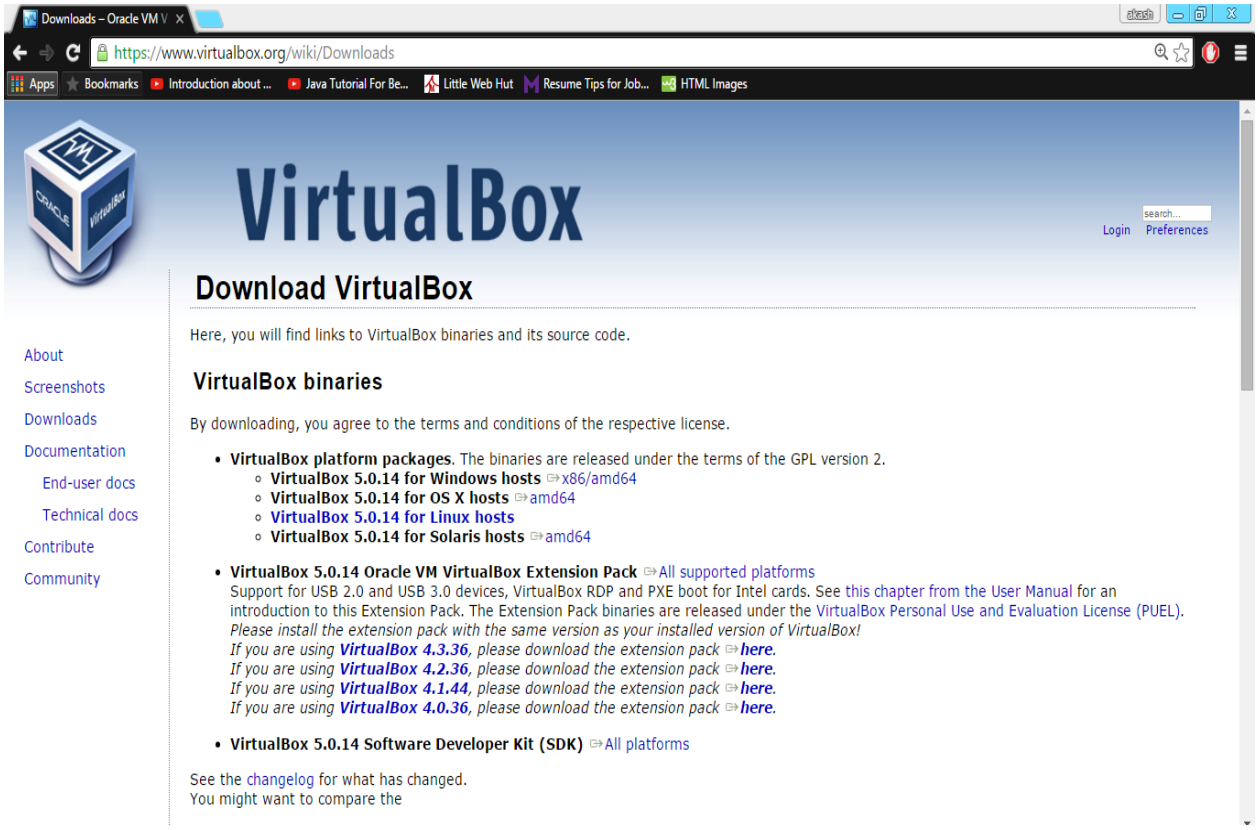
LAB SETUP

The following steps need to be followed to set up the Lab:

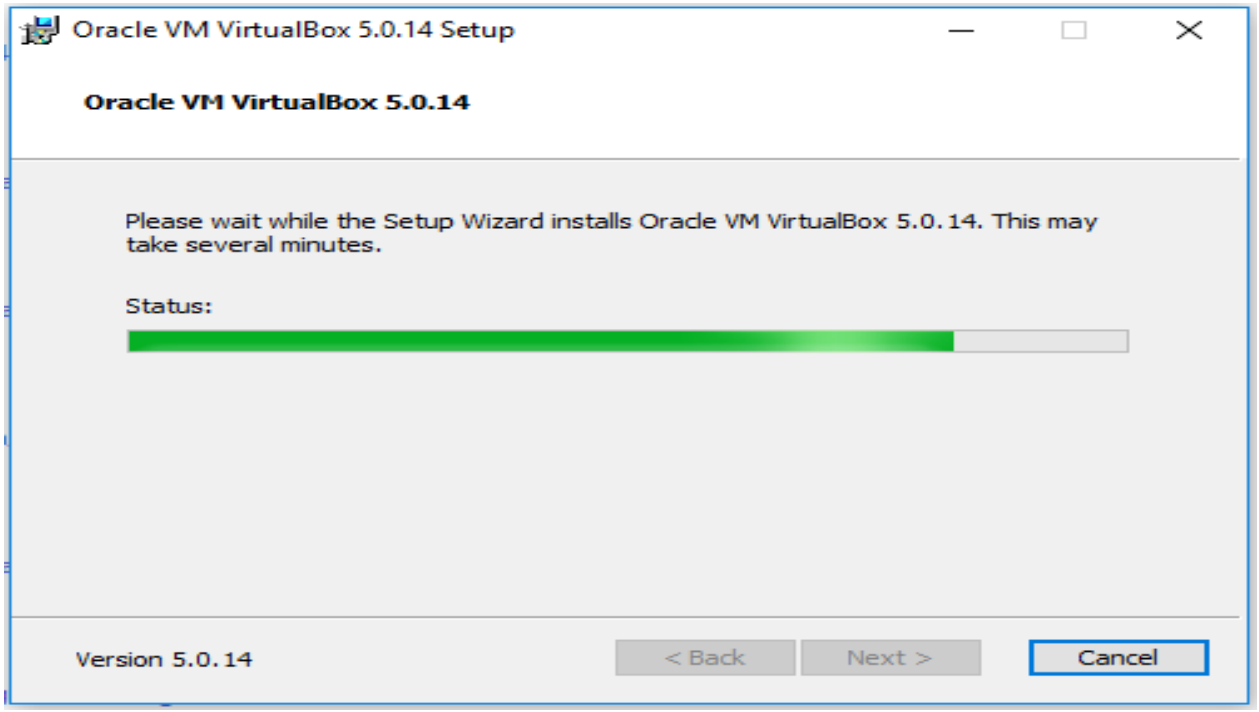
1. **Installing VirtualBox**
2. **Installing Ubuntu inside VirtualBox**
3. **Installing Wireshark inside Ubuntu**

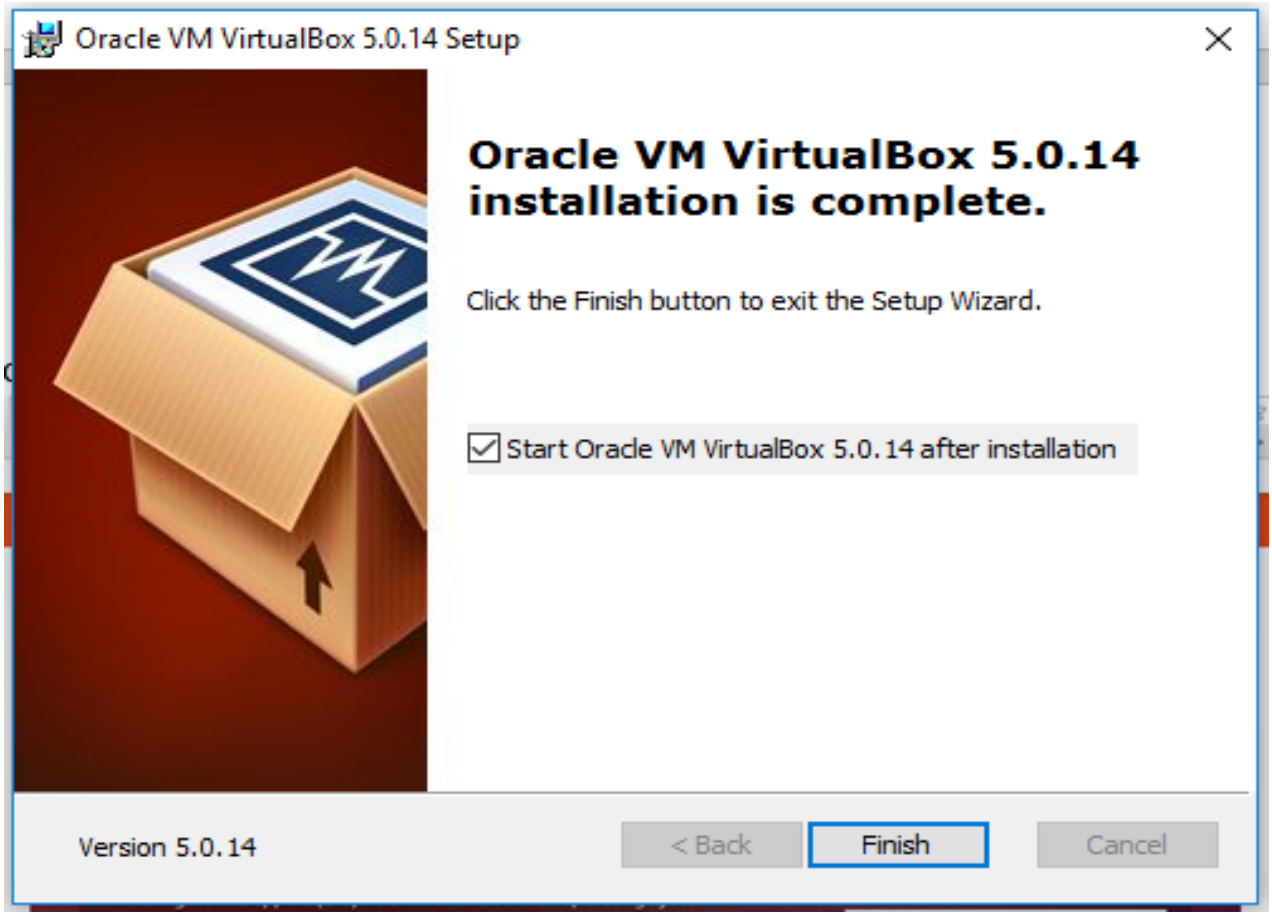
1. Installing VirtualBox

A: Downloading Windows VirtualBox Version (supported by my system):



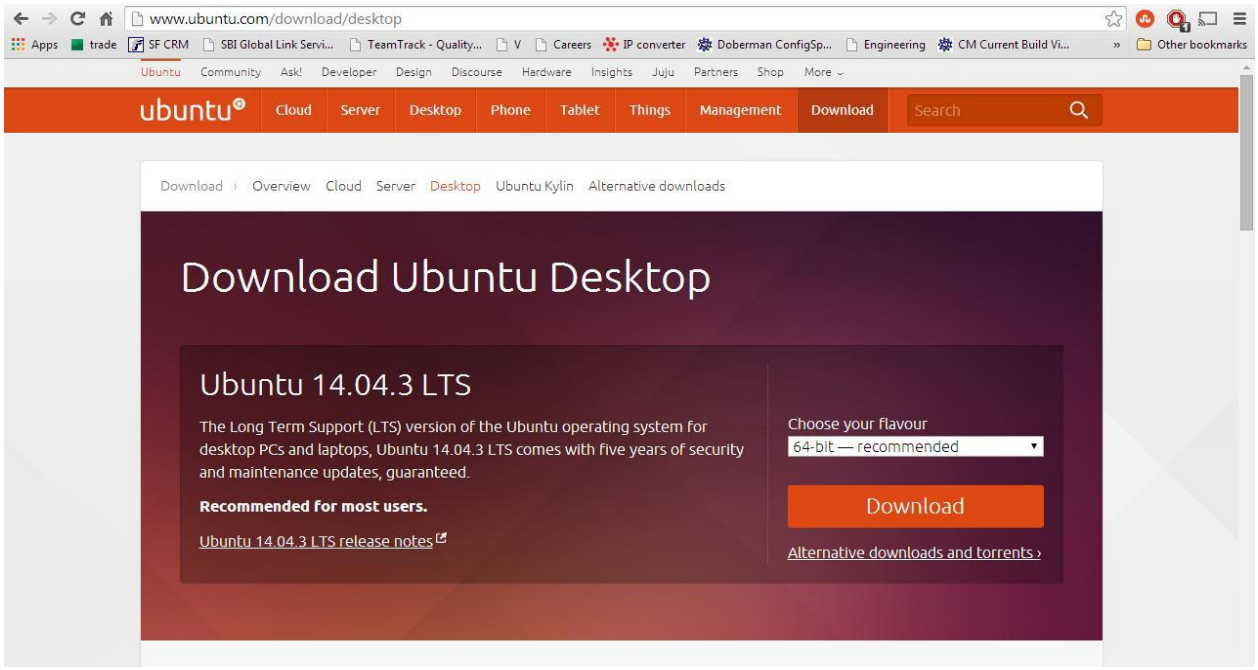
B: Installing VirtualBox in my system:


Oracle VM VirtualBox 5.0.14 Setup
Oracle VM VirtualBox 5.0.14
Please wait while the Setup Wizard installs Oracle VM VirtualBox 5.0.14. This may take several minutes.
Status:
Version 5.0.14
< Back
Next >
Cancel

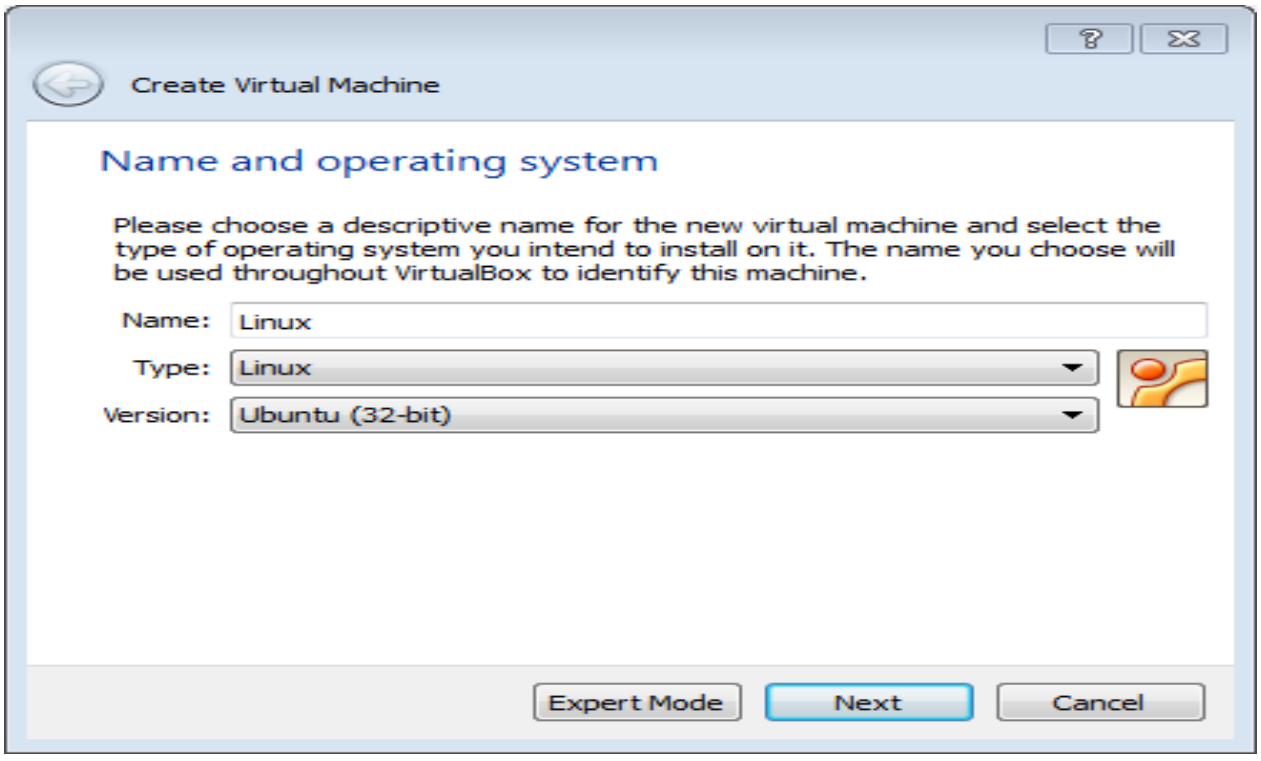

Oracle VM VirtualBox 5.0.14 Setup
Oracle VM VirtualBox 5.0.14 installation is complete.
Click the Finish button to exit the Setup Wizard.
Start Oracle VM VirtualBox 5.0.14 after installation
Version 5.0.14
< Back
Finish
Cancel

2. Installing Ubuntu inside VirtualBox

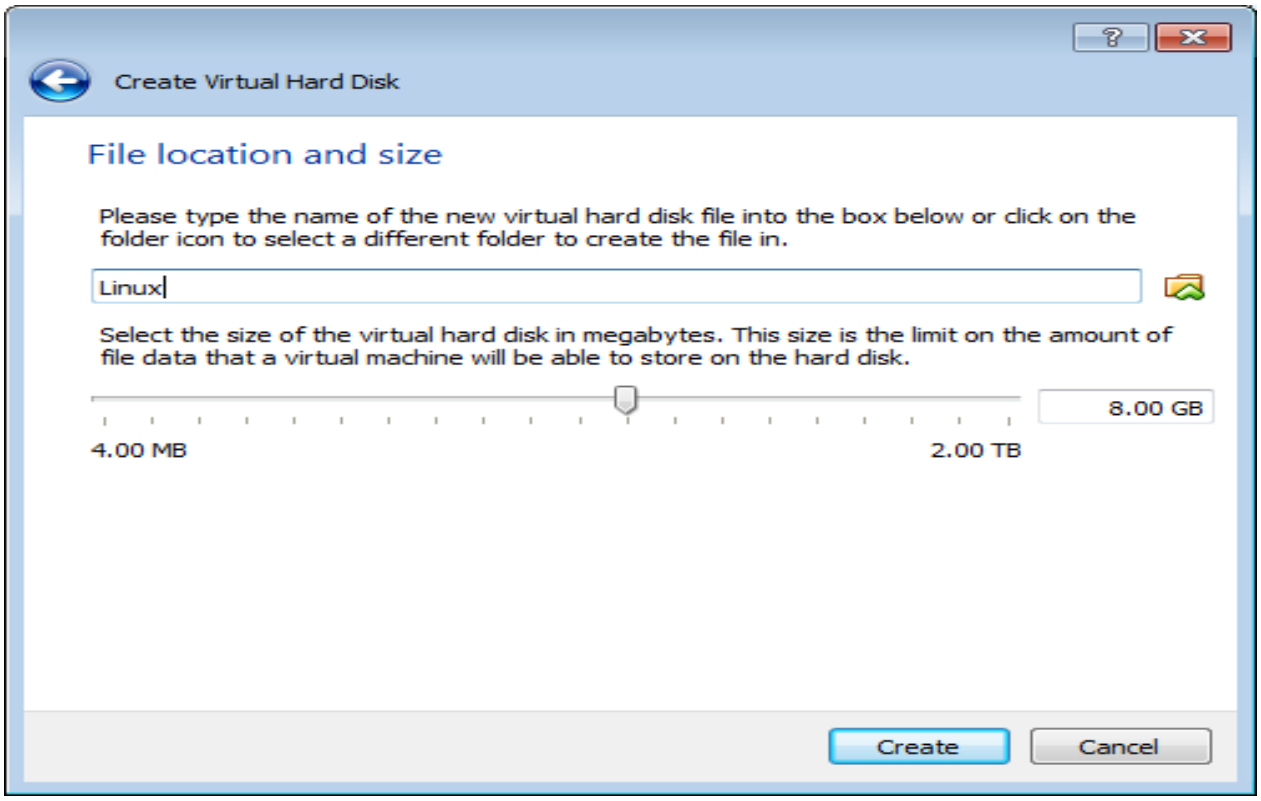
A: Downloading Ubuntu to be installed in VirtualBox:



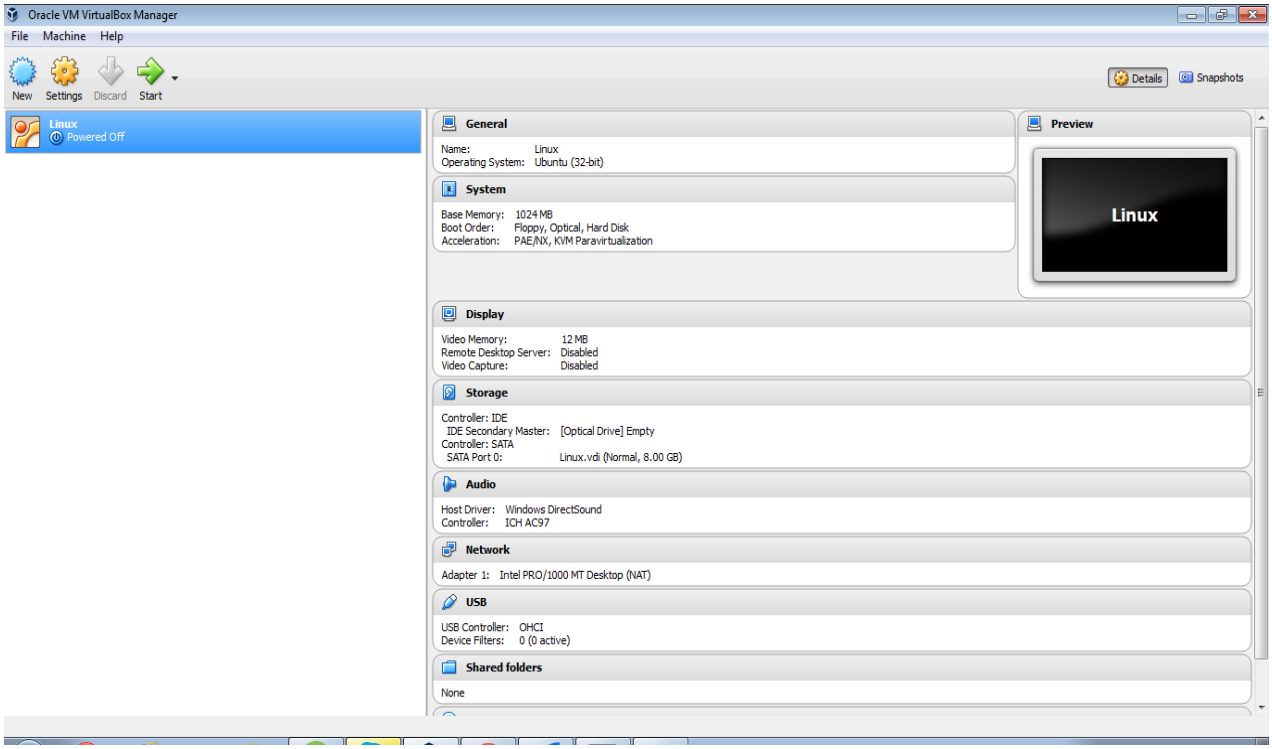
B: Creating the Linux Virtual Machine:



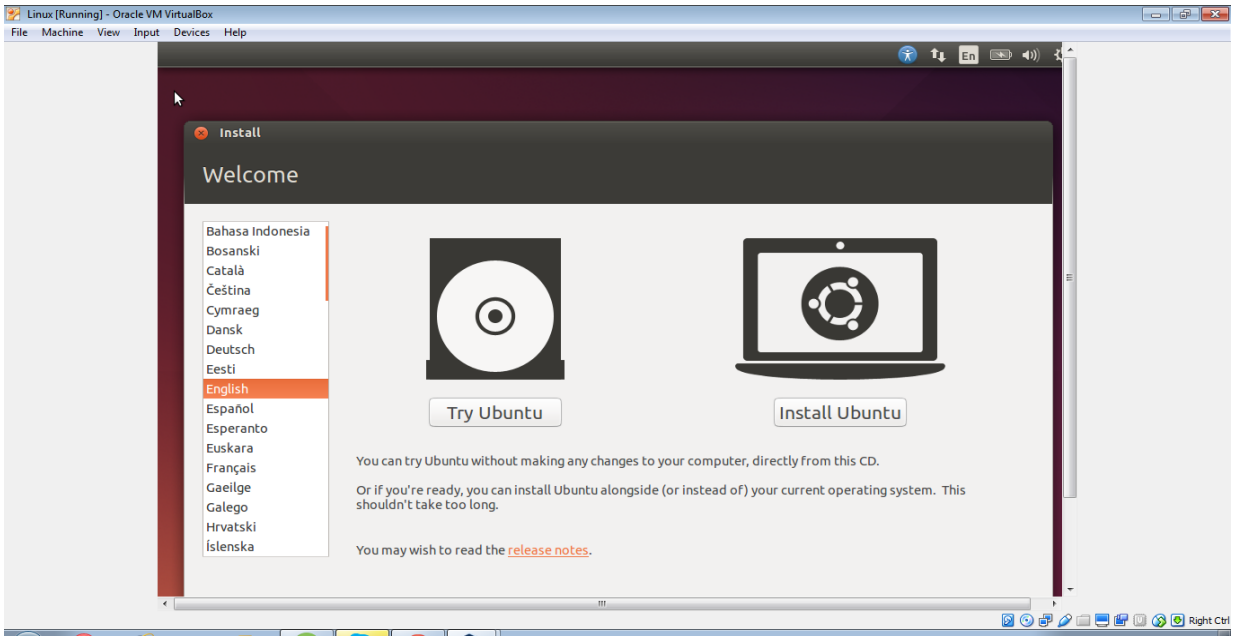
C: Assigning the file location and hard-disk size:



D: Loading and Starting the LINUX Virtual Machine:

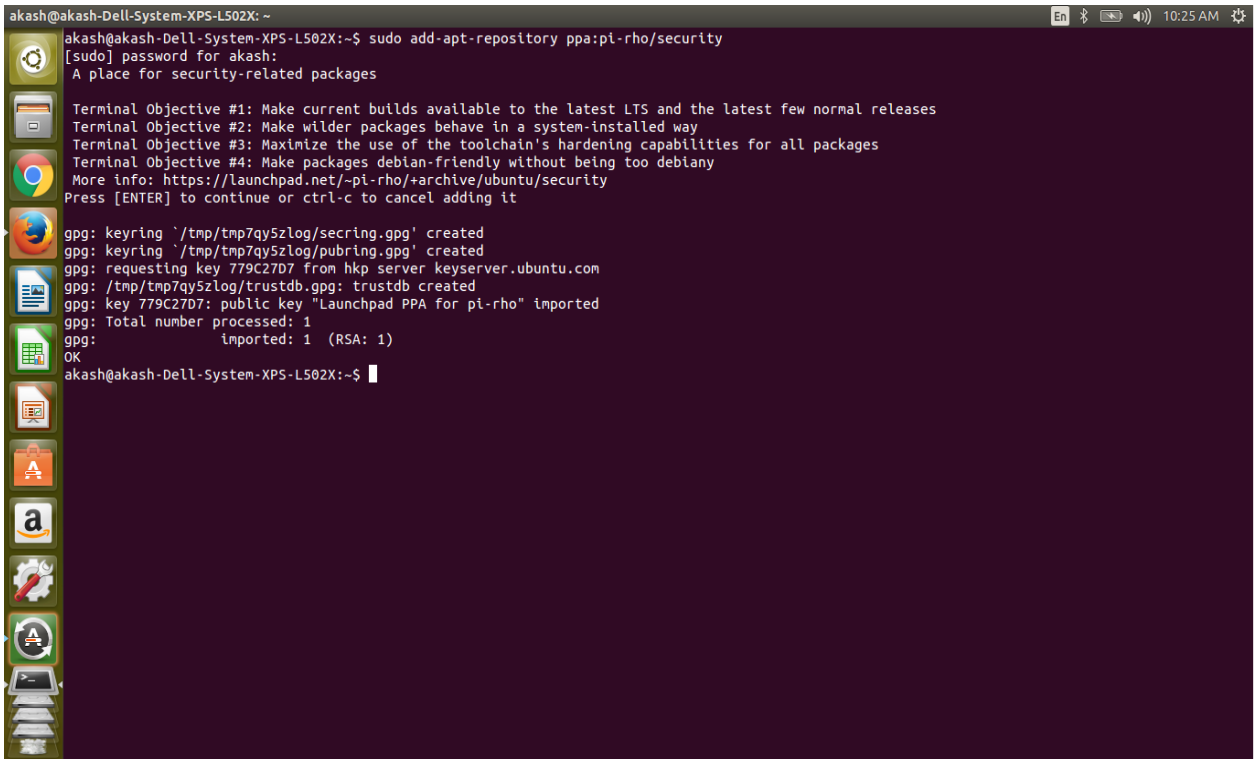


E: Installing Ubuntu:

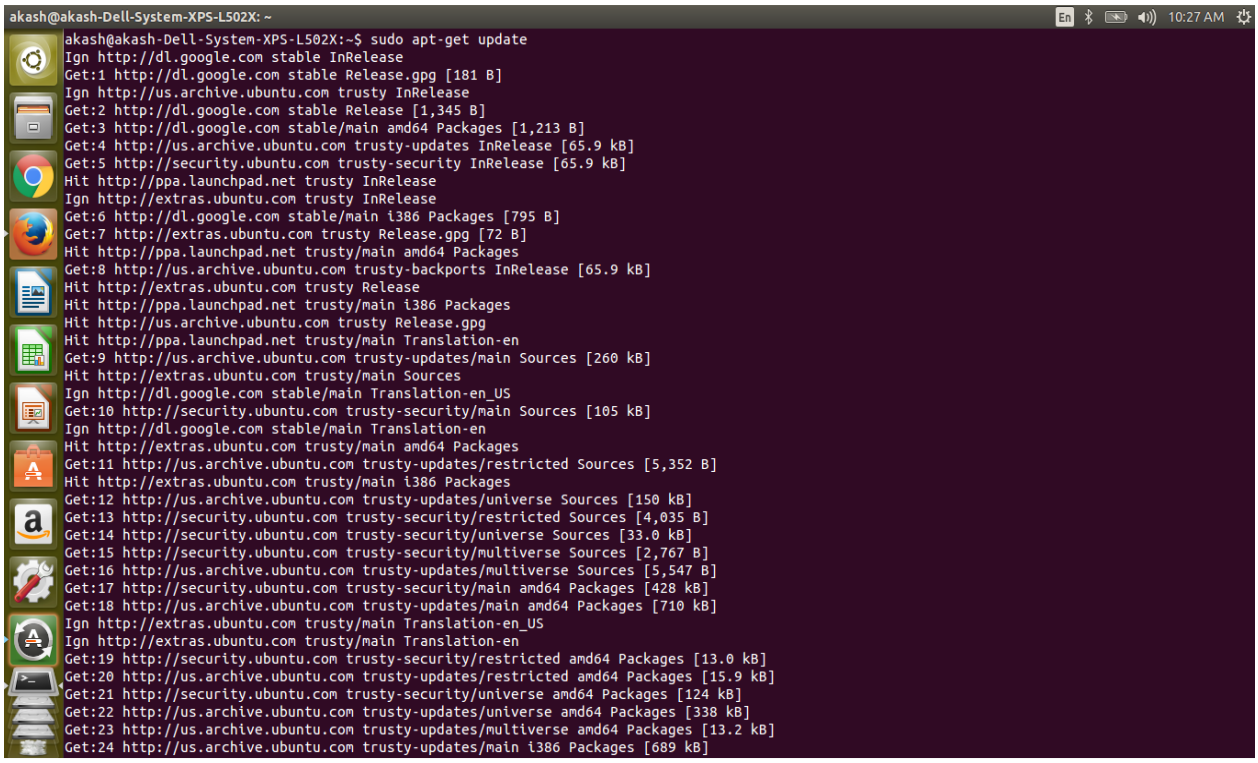


3. Installing Wireshark inside Ubuntu

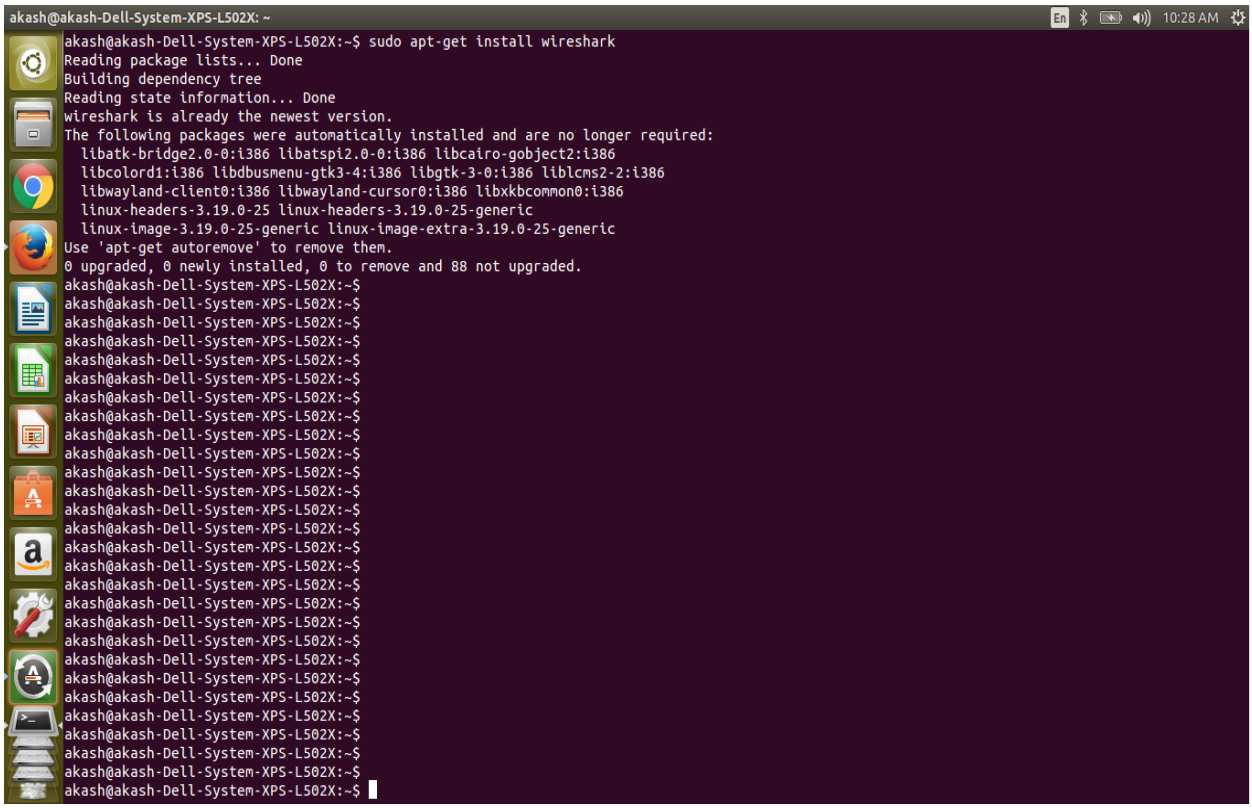
A: Using the command “sudo add-apt-repository ppa:pi-rho/security”:



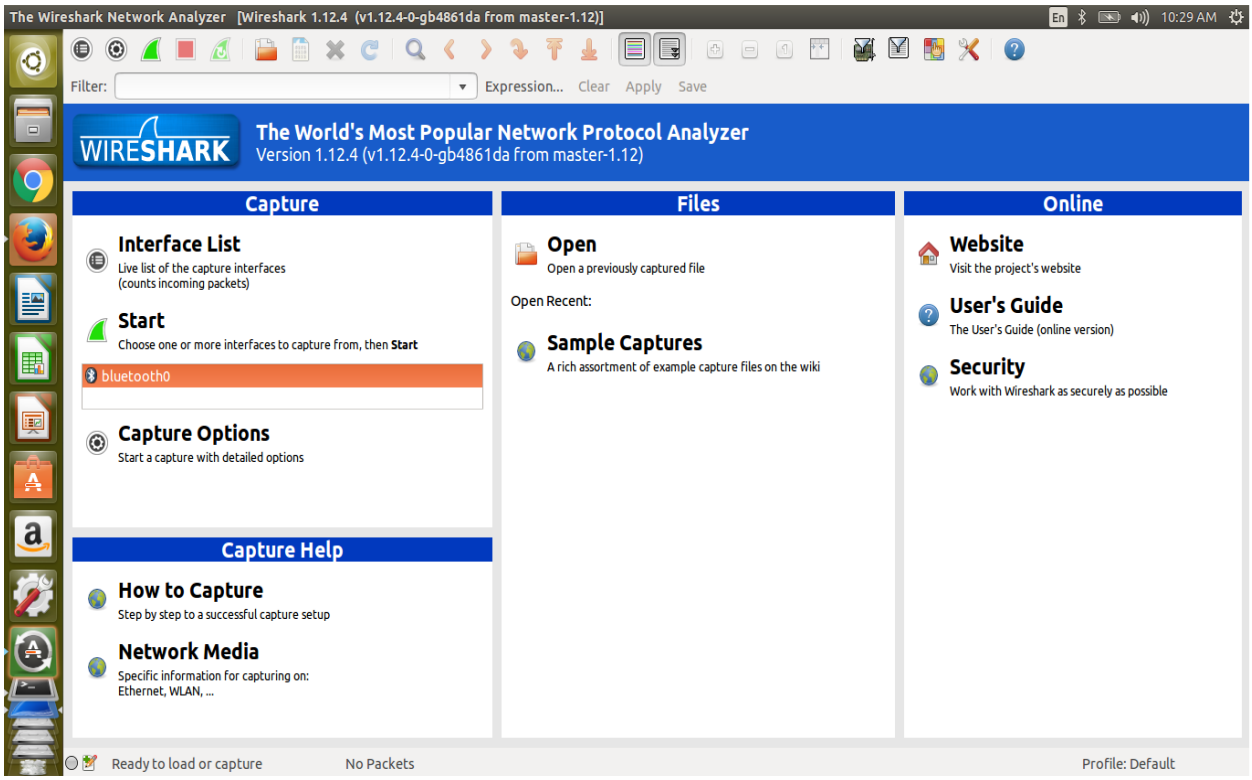
B: Using the command “sudo apt-get update”:



C: Using the command “sudo apt-get install Wireshark”:



D: Wireshark getting successfully installed. By selecting the network connection we can start the packet capture by pressing the “Start” button:

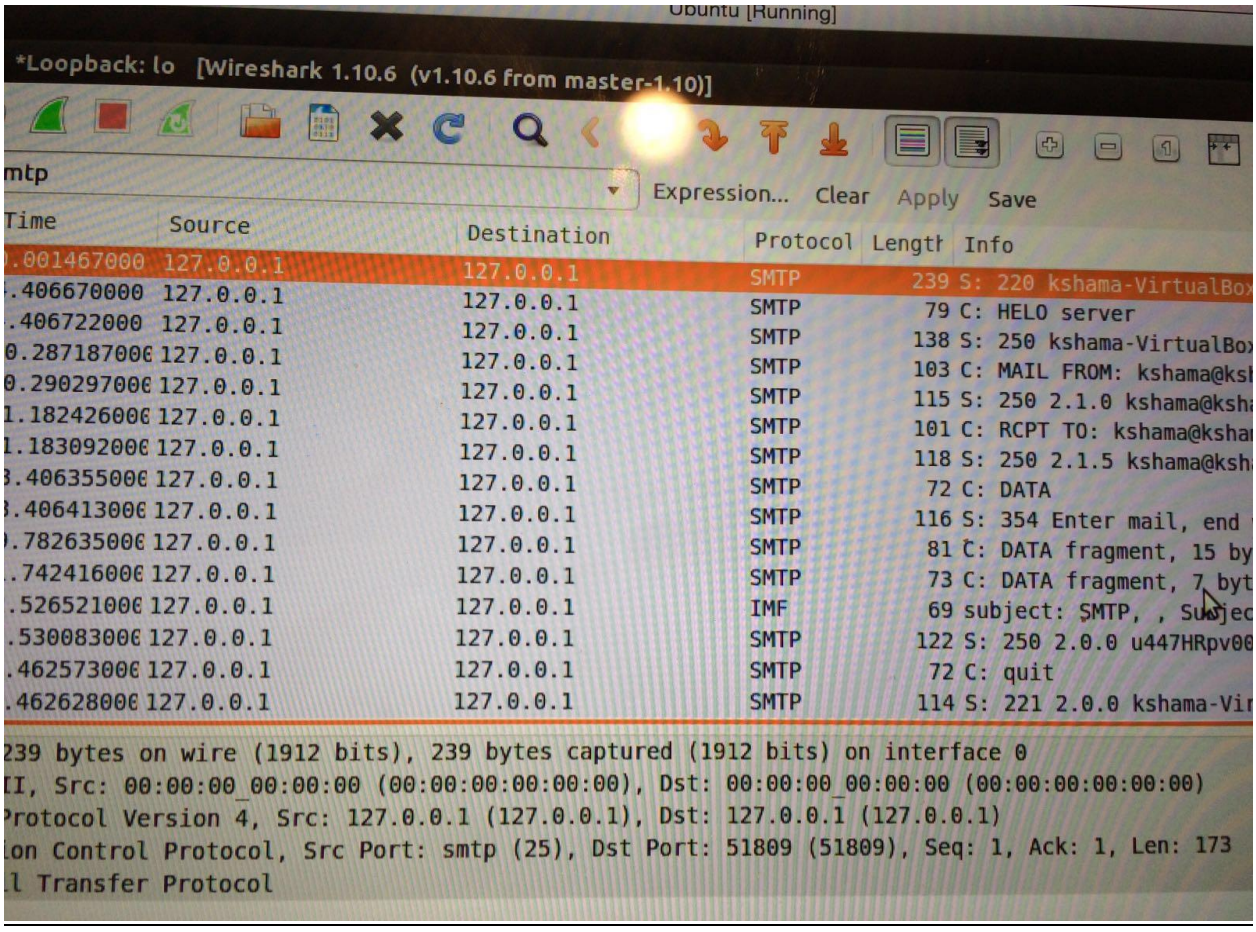


Wireshark: In this project we are using a free and open-source packet analyzer Wireshark to capture and examine a packet trace. It is a software used for network troubleshooting, analysis, software and communications protocol development, and education. A packet capture includes time stamped to every packet. Wireshark provides a graphical UI that helps us to capture the sequence of packets and understand the bit operation. It color-codes packets by their type, and has an inbuilt feature to filter and analyze packets to investigate the behavior of network protocols

The first step would be to install Wireshark on our system and then use the software to implement the smtp commands. Down shown below is the packet capture for smtp. Here we used the sendmail repository and to check the corresponding smtp capture we put smtp command in the filter option in Wireshark software. SMTP allows for communication between a variety of hosts and clients by sending and receiving mails.

```
gunveet@gunveet-VirtualBox: ~
gunveet@gunveet-VirtualBox:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:e6:22:21
          inet addr:10.0.0.195  Bcast:10.0.0.255  Mask:255.255.
255.0
          inet6 addr: 2601:646:8501:5d70:a00:27ff:fee6:2221/64
Scope:Global
          inet6 addr: 2601:646:8501:5d70:9809:9aea:c012:2054/64
 Scope:Global
          inet6 addr: fe80::a00:27ff:fee6:2221/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1070 errors:0 dropped:0 overruns:0 frame:0
          TX packets:472 errors:0 dropped:0 overruns:0 carrier:
0
          collisions:0 txqueuelen:1000
          RX bytes:839522 (839.5 KB)  TX bytes:59562 (59.5 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:275 errors:0 dropped:0 overruns:0 frame:0
          TX packets:275 errors:0 dropped:0 overruns:0 carrier:
0
```



OVERVIEW:

We will see how Simple Mail Transfer Protocol(SMTP) works using its various operations. Topics that we will discuss in the report are:

1) Introduction
   1.1: what is SMTP?
   1.2: Motivation
   1.3: Services
   1.4: Protocol Overview

2) Insights of SMTP protocol
   2.1: SMTP Protocol Components
   2.2: SMTP Sessions
   2.3: SMTP Message Format
   2.4: SMTP Handshake
   2.5: SMTP Standards
   2.6: QoS

3) Modeling
   3.1: Model and Protocol Description
   3.2: How SMTP works
   3.3: SMTP status codes
   3.4: SMTP command reference
   3.5: Extended SMTP commands

4) SMTP Implementation
   4.1: Implementation using localhost server(Sendmail)
   4.2: Implementation using SSMTP
   4.3: Wireshark capture analysis for SSMTP

5) Extended implementation and Security issues.
   5.1: why is it needed
   5.2: Size limits
   5.3: Time Outs
   5.4: Security concerns of SMTP

6) SMTP usefulness

7) SMTP deduction

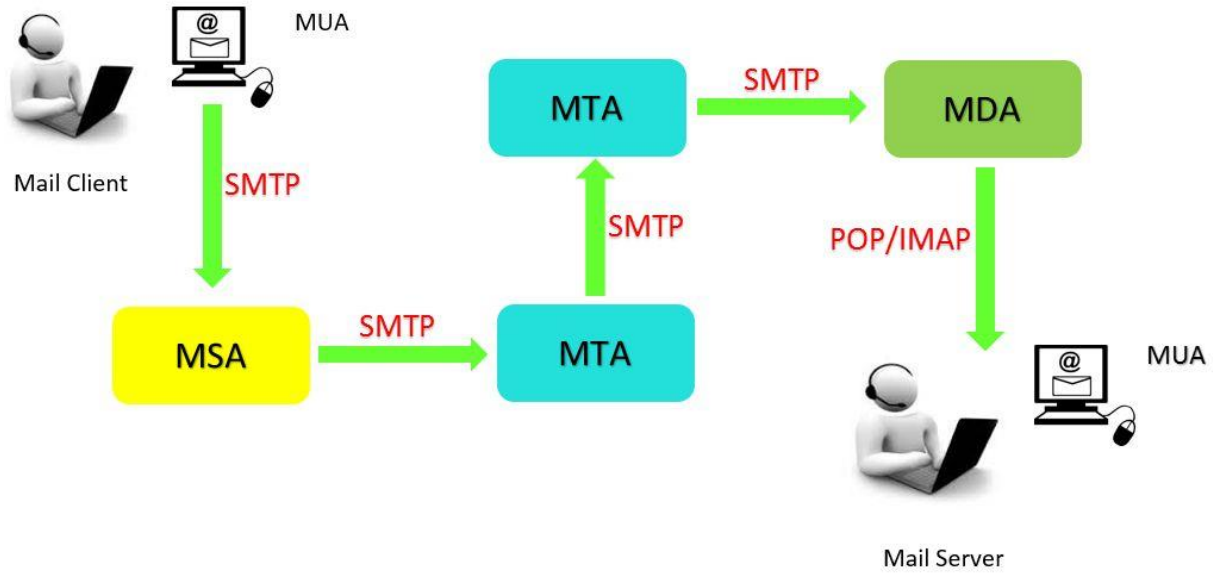
8) Learning Outcomes

9) Conclusion
10) References

1. INTRODUCTION

1.1 What is SMTP?

Simple Mail Transfer Protocol (SMTP), first documented in RFC 821, is Internet's standard host to host mail transport protocol, used in sending and receiving emails. SMTP operates on the port 25 of the TCP/IP. It requires only a reliable ordered data stream channel. An important aspect or SMTP is its capability to send mail across multiple networks, known as "SMTP mail relaying". For receiving mails, client applications usually make use of P OP3 or I MAP. **POP3** stands for Post Office Protocol. Using POP3, an email client is allowed to download a mail from the email server. POP3 is a simple protocol and offers only download feature. It does not support more number of features. Its design is such that it lets the client download all emails available from the server. And then deletes them from server and finally disconnects. POP3 uses Port 110. **IMAP** stands for Internet Message Access Protocol. IMAP has many features similar to POP3. Like POP3, IMAP is a protocol that can be used by the client to download email from an email server. But unlike POP3, IMAP offers many more features. It is designed to let the use keep their emails on the server instead of downloading and deleting it immediately. IMAP needs more CPU resources and larger disk space on the server as compared to POP3. This is so because all emails are stored on the server. IMAP usually uses port 143.



1.2. Motivation (History of SMTP)

Various forms of one to one electronic mailing were in use in 1960. People communicated using mainframe computers. As more computers were interconnected, new developed standards came into picture which allowed users on different systems to mail each other. SMTP grew out of these standards during 1970s. SMTP has its root in two implementations. One, the Mail Box Protocol and other the SNDMSG program. Later on came the FTP implementation and Mail Protocol in 1973.

In early 1980s, SMTP became widely used. S MTP became widely used in the early 1980s. Initially, SMTP servers were typically meant to be internal to an organization. It was used to receive mail from outside for the organization and relay messages from the organization for the outside. Subsequently, SMTP servers (mail transfer agents), started expanding their features to become message submission user or the Mail User Agents. Due to the rapid popularization and expansion of

SMTP, SMTP required to add rules and methods for relaying mail and adding authentication for users to prevent relay of unsolicited emails like spam.

Many people contributed to the core SMTP specifications, among them J on Postel, Eric Allman, Dave Crocker, Ned Freed, Randall Gellens, John Klensin, and Keith Moore.

With SMTP, a process can transfer mail to another process either on the same network or on between different networks using relay or gateway process. S M T P uses an asymmetric request response type protocol, used extensively in 1980s and is still occasionally seen in most mail protocols. It was last updated in 2008 and is known as Extended SMTP (ESMTP) defined by RFC 5321, which is the protocol used widely today.

1.3. Services (What SMTP can do)

Simple Mail Transfer Protocol (SMTP) is based on end-to-end message delivery.

An SMTP client after contacting the SMTP server On port 25, waits for the server to send a 220 READY for mail reply message. On receiving this message, the client sends a HELO command. The server will then send a reply with a "250 Requested mail action Okay" message.

The mail transaction now begins with MAIL command identifying the sender and a FROM field that holds the address to which error must be sent.

After MAIL, sender sends some RCPT commands to identify the recipients of the mail. The Receiver then acknowledges all the RCPT commands individually by either issuing a 250 OK respond or the error message 550- No such user here.

Once the RCPT commands are acknowledged, the sender will send a DATA command to tell the receiving client that the sender is ready to transfer an entire mail message. The receiver sends a reply with message 354 Start mail command. This reply also contains an ending sequence to be used by the sender to terminate the message data.

There are 5 parts in this termination sequence. These 5 characters are: carriage return, line feed, period, carriage return, and line feed.

The client now sends the message line by line, terminating each line with the 5-character sequence. The receiver responds with 250 OK or error message. After the sending is completed, the client can follow any of these actions. Terminate Session: When there are no more messages to be sent by the current Simple Mail Transfer Protocol (SMTP) client, the connection can be ended using QUIT command, which is answered with a 221 Service closing transmission channel reply. Exchange Roles: When there are no more messages to be sent by the Simple Mail Transfer Protocol (SMTP) client, but it is ready to receive message from SMTP server. The SMTP client issues TURN command. Now the SMTP client and server switch their role of sender/receiver. So now the sender (earlier receiver) sends messages by issuing MAIL command.

Send Another Mail: If the Simple Mail Transfer Protocol (SMTP) client (sender) has another message to send, it can issue a new MAIL command.

.

. **1.4. Overview of the Protocols**:

1.4.1. Mail Protocols

- SMTP- Simple Mail Transport Protocol, Used on internet. It's an application layer protocol.
- POP3- Post Office Protocol version 3. Clients use POP3 to access the internet mail server to send/receive mails. It is not a protocol of transport layer.
- IMAP4 -Internet Mail Access Protocol version 4. This protocol is considered as an advanced or replaced version for POP3.
- MIME -Multipurpose Internet Mail Extension. This protocol defines the way through which files are attached to SMTP messages.
- X.400 -International Telecommunication Union standard which defines the transfer protocols used for sending mails between the mail servers.
- MHS -Message Handling Service. This is used for mail on Netware networks.

1.4.2 Directory Services

- Lightweight Directory Access Protocol (LDAP)
- X.500 - This gives an outline of how an organization can share names and objects on a large network. X.500 system is similar to a directory. Its recommendation comes from the International Telegraph and Telephone Consultative Committee (CCITT).

1.4.3. Mail API

The Mail Application Programming Interface (API) allows e-mail feature to be included in other application programs.

● MAPI - Microsoft's Messaging API which is incorporated throughout Microsoft's office products support mail at the application level

● VIM – Vendor-Independent Messaging protocol from Lotus is supported by many vendors exclusive of Microsoft.

● There are three parts in a mail message:

○ Envelope -This section of the mail message holds the recipient and sender addresses using the MAIL and RCPT commands.

○ Headers -Each header is made of a name and followed by a colon with its value.Some headers are Date, Message ID, From, To and Subject.

○ Body -This includes the content of the message that i sent as a 7 bit ASCII code.

● SMTP Commands:

○ HELO -Sent from client with a domain name. Eg:system.company.com

○ MAIL -From <new@system.company.com>

○ RCPT -To <receiver@rcvmachine.rcvorg.org>

○ DATA -The contents of the message are sent via DATA command. First the the headers are sent followed by a blank line, then body of the message is sent. Any line with only "." and no other character implies end of a message.

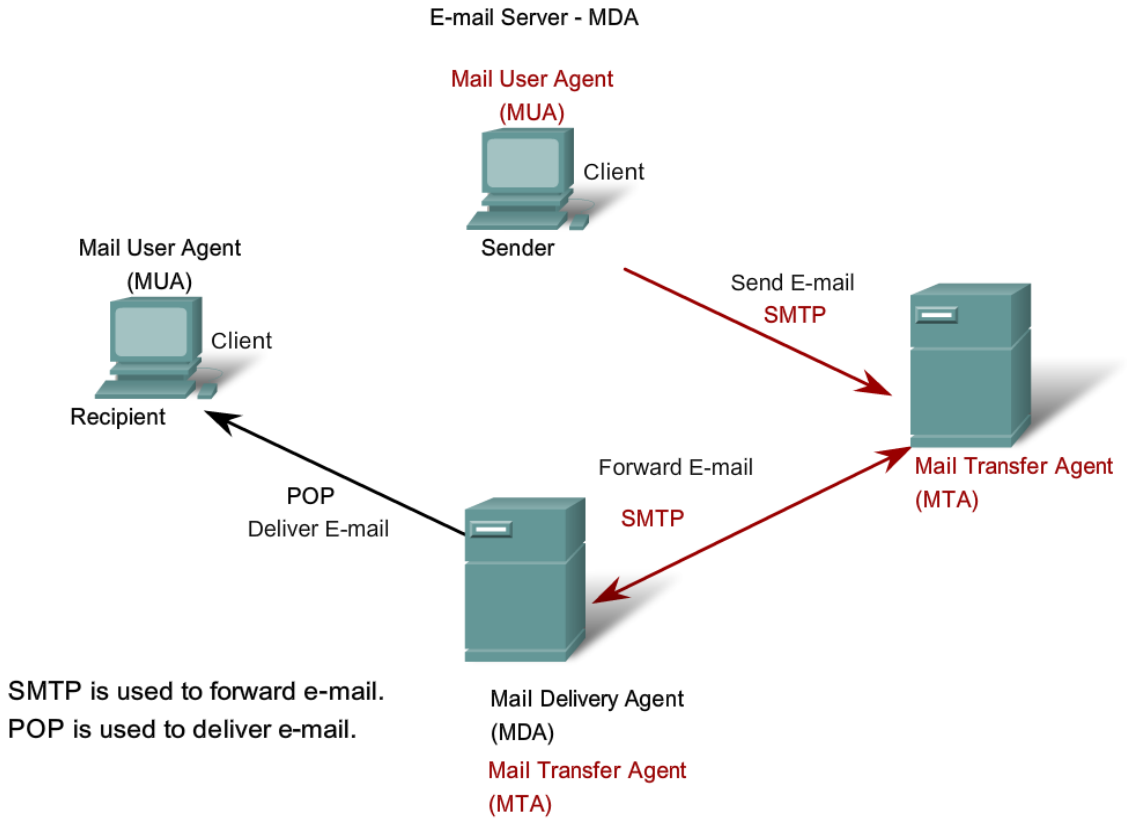
- QUIT

2. INSIGHTS OF SMTP PROTOCOL

2.1 SMTP components

There are four types of programs that we use while sending and receiving mail:

1) **MUA** – Mail Users Agent. This is what the user uses to type an e-mail. This has an editor for support and when the user types the mail it is sent to the MTA.
2) **MTA** – Also known as the Mail Transfer Agent, this program is used for sending mail from sending machine to the receiving machine. We have an MTA program running both on the sending side and the receiving side. Sendmail is characterized under MTA.
3) **LDA** – Local delivery agent is on the receiving side of the machine and receives the mail from the MTA. This program is mostly procmail.
4) **Mail Notifier** – The job of this program is to notify the recipient that the mail has been received. Usually this requires two programs, biff and comsat.



MTA on both the machines, receiving and sending uses the SMTP to pass mail between them. Some other components of the mail service include:

1) **Directory Services** - A list of users on the system. Microsoft gives a Global address list and a Personal address book.
2) **Post Office** – Usually the place where all the messages are stored.

2.2. SMTP Sessions

Each SMTP session has commands by SMTP client and according responses from server. So when the session is opened, all the corresponding sessions are exchanged.

So each session may include transactions ranging from zero to many. Mainly SMTP transaction consists of following command or reply sequences.

- MAIL command is used to establish the return address, a.k.a. Return-Path, reverse-path, bounce address, mfrom, or envelope sender.
- RCPT command, to establish a recipient of this message. This command can be issued multiple times, one for each recipient. These addresses are also part of the envelope.
- DATA to signal the beginning of the message text; the content of the message, as opposed to its envelope. It consists of a message header and a message body separated by an empty line. DATA is actually a group of commands, and the server replies twice: once to the DATA command proper, to acknowledge that it is ready to receive the text, and the second time after the end-of-data sequence, to either accept or reject the entire message.

Besides for DATA, server response can be positive (codes with starting digit 2) Or negative. Negative replies can be permanent (5xx codes) or transient (4xx codes).

A reject is considered as a permanent failure from the SMTP server. Considering this case the SMTP client should send a bounce message. A drop is a positive response followed by message discard rather than delivery.

The initiating host, the SMTP client, can be either an end-user's email client, functionally identified as a mail user agent (MUA), or a relay server's mail transfer agent (MTA), that is an SMTP server acting as an SMTP client, in the relevant session, in order to relay mail. Fully capable SMTP servers maintain queues of messages for retrying message transmissions that resulted in transient failures.

2.3 SMTP Message Format

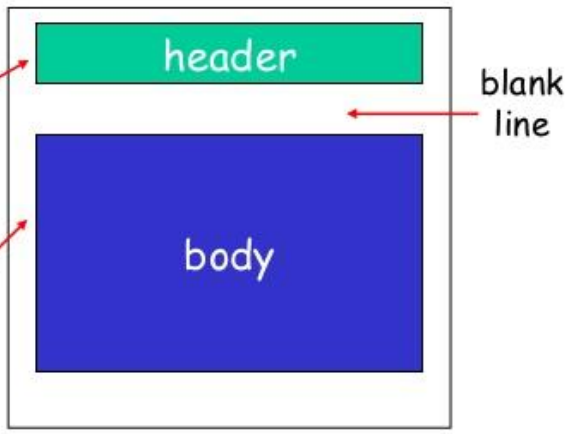
Mail message format

SMTP: protocol for exchanging email msgs

RFC 822: standard for text message format:

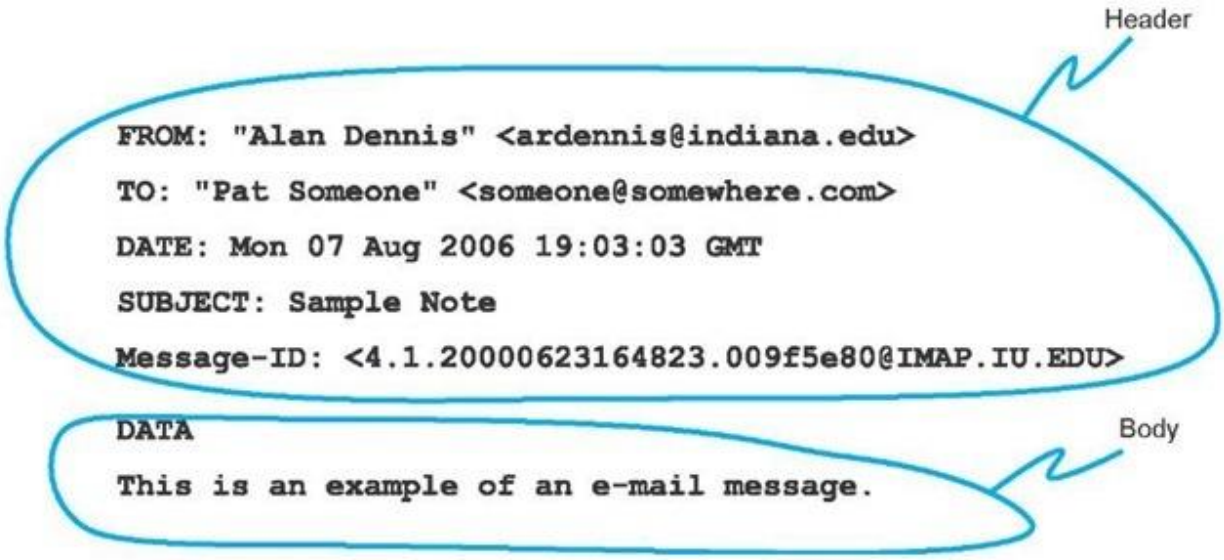
- header lines, e.g.,
  - To:
  - From:
  - Subject:

  different from SMTP commands!
- body
  - the "message", ASCII characters only

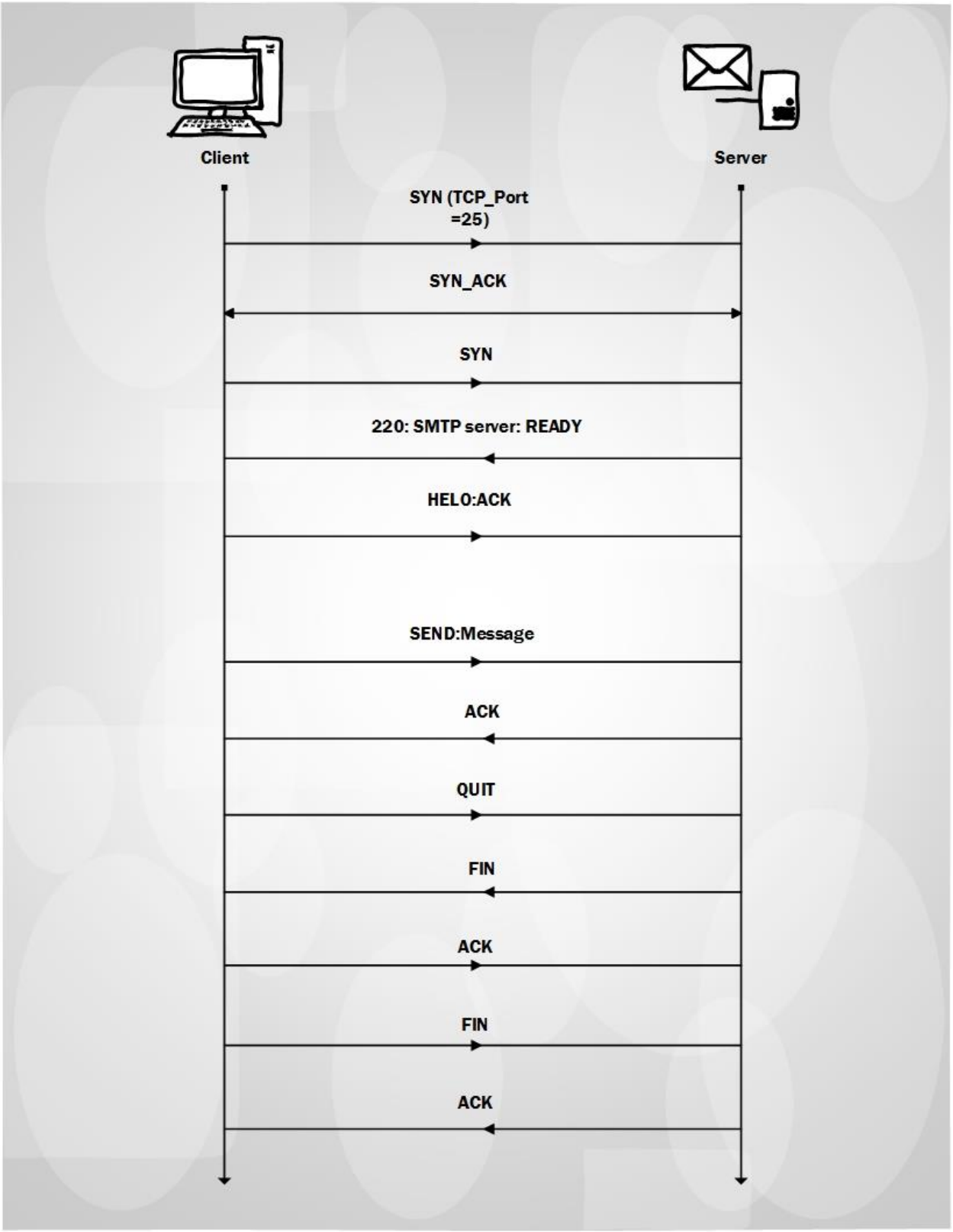


9

Sample SMTP Header and Body



2.4. SMTP Handshake



2.5. SMTP standards

- **RFC 1123** – Requirements for Internet Hosts—Application and Support
- **RFC 1870** – SMTP Service Extension for Message Size Declaration

- **RFC 2505** – Anti-Spam Recommendations for SMTP MTAs
- **RFC 2920** – SMTP Service Extension for Command Pipelining
- **RFC 3030** – SMTP Service Extensions for Transmission of Large and Binary MIME Messages
- **RFC 3207** – SMTP Service Extension for Secure SMTP over Transport Layer Security
- **RFC 3461** – SMTP Service Extension for Delivery Status Notifications
- **RFC 3463** – Enhanced Status Codes for SMTP
- **RFC 3464** – An Extensible Message Format for Delivery Status Notifications
- **RFC 3798** – Message Disposition Notification
- **RFC 3834** – Recommendations for Automatic Responses to Electronic Mail
- **RFC 4952** – Overview and Framework for Internationalized Email
- **RFC 4954** – SMTP Service Extension for Authentication
- **RFC 5068** – Email Submission Operations: Access and Accountability Requirements
- **RFC 5248** - A Registry for SMTP Enhanced Mail System Status Codes
- **RFC 5321** – The Simple Mail Transfer Protocol
- **RFC 5322** – Internet Message Format
- **RFC 5504** – Downgrading Mechanism for Email Address Internationalization
- **RFC 6409** – Message Submission for Mail
- **RFC 6522** – Report Content Type for the Reporting of Mail System Administrative Messages

  RFC 6531 – SMTP Extension for Internationalized Email Addresses

2.6. Quality of Service (QoS) in SMTP

All SMTP messages carry same Differentiated service value. It is observed that there is no extra priority give to the SMTP packets.

In the project we have observed that all the SMTP messages contains DSF as 0x00 and DSCP code as 0x00 Default in the IPV4 section which can be considered as default code. This gives a hint that SMTP does not use any kind of special priorities in transferring the packets.

Following Wireshark capture depicts the DSF field for SMTP packet

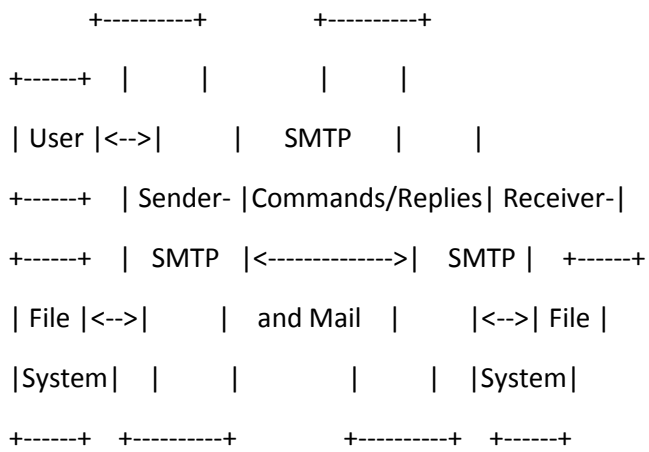
```
406 18… 173.194.202.108  192.168.59.133   SMTP     209 S: 250 smtp.gmail.com at your service, [150.65.2…
407 18… 192.168.59.133   173.194.202.108  SMTP      64 C: STARTTLS
409 18… 173.194.202.108  192.168.59.133   SMTP      84 S: 220 2.0.0 Ready to start TLS
```

```
Frame 409: 84 bytes on wire (672 bits), 84 bytes captured (672 bits)
Ethernet II, Src: Vmware_e0:57:59 (00:50:56:e0:57:59), Dst: Vmware_e8:b0:72 (00:0c:29:e8:b0:72)
Internet Protocol Version 4, Src: 173.194.202.108, Dst: 192.168.59.133
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes
▾ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    0000 00.. = Differentiated Services Codepoint: Default (0)
    .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
  Total Length: 70
  Identification: 0x6d1c (27932)
▸ Flags: 0x00
  Fragment offset: 0
```

3 SMTP MODELLING

3.1 Description:

The SMTP design is based on the following model of communication: as the result of a user mail request, the sender-SMTP establishes a two-way transmission channel to a receiver-SMTP. The receiver-SMTP may be either the ultimate destination or an intermediate. SMTP commands are generated by the sender-SMTP and sent to the receiver-SMTP. SMTP replies are sent from the receiver-SMTP to the sender-SMTP in response to the commands.



Sender-SMTP Receiver-SMTP

Model for SMTP Use

Once the transmission channel is established, the SMTP-sender sends a MAIL command indicating the sender of the mail. If the SMTP-receiver can accept mail it responds with an OK reply. The SMTP-sender then sends a RCPT command identifying a recipient of the mail. If the SMTP-receiver can accept mail for that recipient it responds with an OK reply; if not, it responds with a reply rejecting that recipient (but not the whole mail transaction). The SMTP-sender and SMTP-receiver may negotiate several recipients. When the recipients have been negotiated the SMTP-sender sends the mail data, terminating with a special sequence. If the SMTP-receiver successfully processes the mail data it responds with an OK reply. The dialog is purposely lock-step, one-at-a-time.

The RFC 281 standard articulates on the client-server protocol . The client SMTP will initiates the session while the server replies back to the session request.

A SMTP process is capable of transferring mails on same network or to other through a gateway or relay. Users interact using user agent (UA). Some of the popular UAs are as follows,

UNIX: Berkeley Mail, ELM, MH, Pine and Mutt.

Windows: Microsoft Outlook/Outlook Express and Netscape/Mozilla Communicator.

The exchange of mail through TCP is completed by an MTA (Mail Transfer Agent). The most common MTA we observe is Microsoft Exchange and sendmail in Linux.

The MTA uses mail boxes to deliver the mail from which a user can access the content.

The RFC 821 specifies standard for communication between two MTAs through TCP. The RFC 822 provides format for messages to be transmitted.

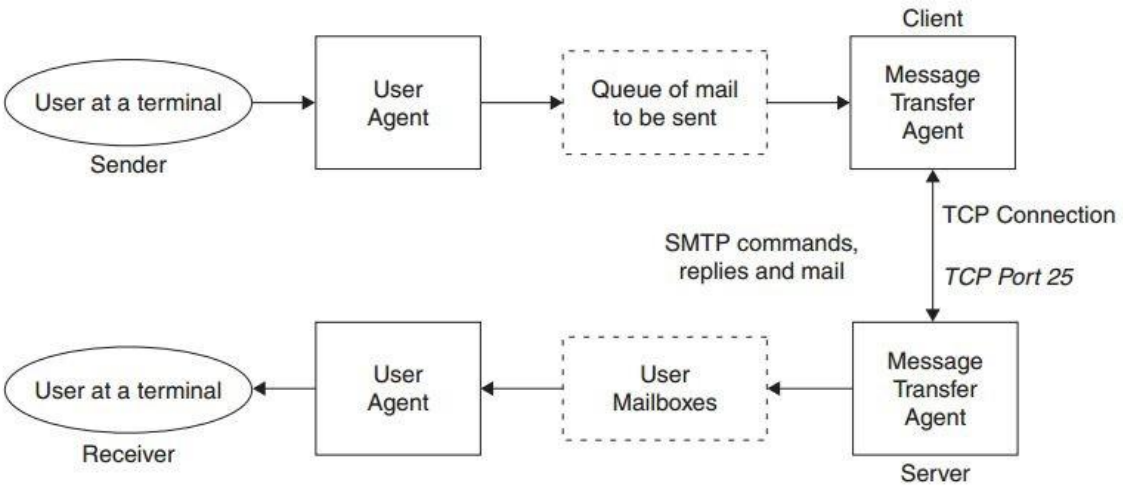


Figure 1: The basic simple mail transfer protocol (SMTP) model.

It can be observed that both the server have a user agent and MTA. The following describes ways of communication between a sender and receiver.

- Direct delivery
- Indirect delivery

Indirect delivery, the sender UA prepares the message, pack message into an envelope and MTA transfers the message across the network on TCP-port 25.

In indirect communication, relaying is involved. We see that in addition to one MTA at the sender site and receiver site, there are many MTAs involved in delivery of the message.

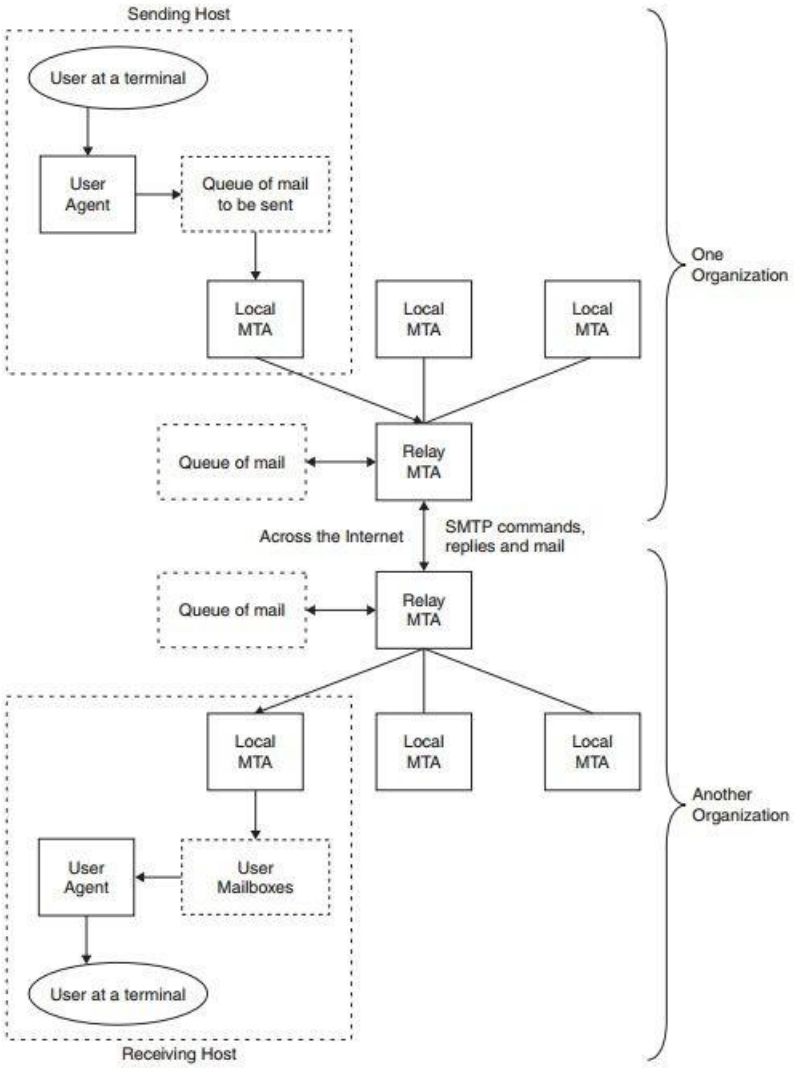


Figure 2: The simple mail transfer protocol (SMTP) model with relay mail transfer agents.

Format: It consists of following fields

- To: It consists primary recipient
- Cc: secondary recipient
- Bcc: blind copies for other recipients
- From: creator of the message
- Sender: email address of the sender
- Received: trace of transfer agent in case of relay being used
- Return-Path: accessed to trace back the sender

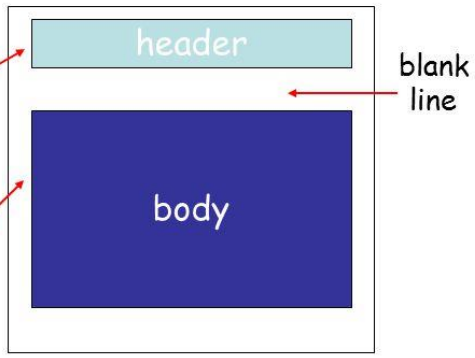
Mail message format

SMTP: protocol for exchanging email msgs

RFC 822: standard for text message format:

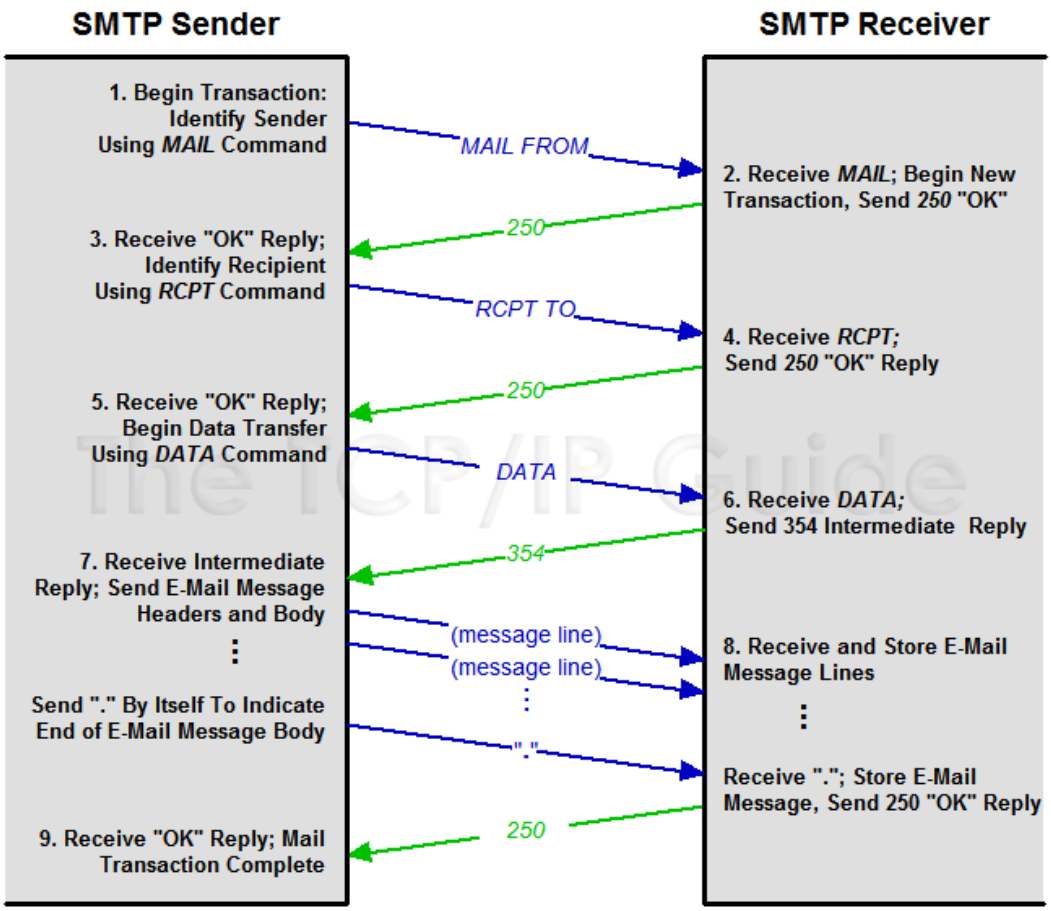
- header lines, e.g.,
  - To:
  - From:
  - Subject:

  different from SMTP commands!
- body
  - the "message", ASCII characters only



25

3.2. Working:



As we have already described SMTP works on end to end message delivery. For this purpose the sender and receiver utilizes port 25 for communication. A client reaches server thought port 25. When client requests server to start communication, server replies with status code 220 indicating ready for mail. The client sends HELO command for the reply from server which will be “250 Requested mail action okay”.

And then after initial setup actual mail communication starts with the MAIL command from the ender which include FROM field.

Later, the sender will send a series of RCPT (Recipient) commands identifying recipients of the mail message. The receiver will acknowledge each of RCPT command by sending back 250 OK. In case of any errors it denotes with the 550 error message.

Using DATA command the sender informs receiver that data or body is ready to be transferred. Receiver the respond with the status code 354 along with an ending sequence that sender should be using to terminate the message data. This sequence generally consists of 5 characters characters: carriage return, line feed, period, carriage return and line feed.

Now the client is ready and starts sending the data line by line, ending with the character sequence line sent by the receiver, upon receiving the receiver will acknowledge with a 250 OK message, or an appropriate error message if anything went wrong.

Upon completion of the transaction the client may opt for any of the following scenarios for maintaining the state of the connection

Terminate Session: the connection can be closed using QUIT command by the client and the server will respond with a status of 221 indicating acceptance for closing transmission channel.

Exchange Roles: the role of the server and client can be interchanged in case nor client is expecting any reply message for its mail sent. it issues the TURN command to the server for the same. Based on the acceptance the SMTP server will switch the roles and sending mail with a new transaction using MAIL command.

Send Another Mail: If the Simple Mail Transfer Protocol (SMTP) client has another message to send, it can issue a new MAIL command.

To summarize there are 5 SMTP commands utilized in successful transfer of any mail using SMTP

- HELO
- MAIL
- RCPT
- DATA
- QUIT

3.3 SMTP COMMANDS:

A SMTP client establish communication with an SMTP server by using SMTP commands. The following is the core list of SMTP commands supported by all SMTP servers:

- **HELO (HELLO):**

  The client sends this command to the SMTP server to identify itself and initiate the SMTP conversation. The domain name or IP address of the SMTP client will be sent together with the command HELO (e.g “HELO client.example.com”).

- **MAIL FROM:**

  Specifies the e-mail address of the sender. The command also tell the SMTP server that a new mail transaction will be starting and requires the server to reset all its state tables and buffers etc. This command will usually be sent as the first command after the identifying and login process. If the server accepts the client request then the server will reply with a 250 OK reply code.

- **RCPT TO (RECIPIENT TOO):**

  Specifies the e-mail address of the recipient. It can be repeated multiple no of times for a given e-mail message in order to deliver a single message to multiple e-mail recipients.

- **DATA**

  The DATA message starts the transfer of message contents. After the DATA command has been to the server, it will respond with the 354 reply code. After this the message contents can be transferred. After all the contents of the message is sent a single dot (“.”) must be sent in a line by itself. If the message is accepted for delivery, the SMTP server will respond with a 250 reply code.

- **RSET**

  If the RSET command will be sent to the server the current mail transaction will be aborted. The connection will not be closed but all the information of the sender, recipients and e-mail data will be removed and buffers are cleared.

- **VRFY**

  This command asks the server to verify whether the specified user name or mailbox is valid. If the user name is enquired then the full name of the user and the mailbox are returned.
- **NOOP**

  The NOOP command does nothing other than making the receiver to send an OK REPLY. It usually used to verify that the server is still connected to client.
- **QUIT**

  The QUIT command asks the server to close the connection. If the connection can be closed then the server replies with the 221 numerical code and the session is closed.

3.4. Status Codes:

The following table depicts all the possible status codes (or commands) exchanged between a server and client for proper communication

SMTP Status Codes	Description
211	Help reply - system status
214	Help message
220	Service ready
221	Closing connection
250	Requested action okay
251	User not local - forwarding to <path>
354	Start mail input

421	Service not available
450	Action not taken - mailbox busy
451	Action aborted - local error
452	Action not taken - insufficient storage
500	Command unrecognized or syntax error
501	Syntax error in parameters or arguments
502	Command not supported
503	Bad sequence of commands (given out of order)
504	Command parameter not supported

550	Action not taken - mailbox unavailable
551	Not a local user
552	Aborted: Exceeded storage allocation
553	Action not taken - mailbox name not allowed
554	Transaction failed

3.5. Extended SMTP (ESMTP) Commands

If the clients need any arguments to support proper communication, extended communication is utilized. ESMTP extends the SMTP protocol, meaning that Exchange Server and other products that support the ESMTP protocol can perform tasks that wouldn't be possible with basic SMTP, such as email message delivery notification. But both the sending host and the receiving host must support the ESMTP protocol for these extended capabilities to be realized. If the receiving host doesn't support ESMTP, the session will revert to standard SMTP.

If the client initiates the SMTP communication with the EHLO command instead of HELO command then some additional SMTP commands are made to available to the client. These additional SMTP commands are referred to as ESMTP. Every server has its own list of extended commands. Following are list of few extended commands utilized

- **EHLO**

  EHLO command is same as HELO command used for initialization but it tells the server that client wants to use the Extended SMTP protocol. EHLO can be used even if client does not need any ESMTP commands.

- **AUTH**

The AUTH command is used to authenticate the client. Using AUTH command client sends the username and password to the e-mail server. AUTH can be combined with different keywords as PLAIN, LOGIC and CRAMMD5 to use different logins depending on level of security.

- **STARTTTLS (Start Transport Layer Security)**

The communication between client and servers using the SMTP protocol normally uses plain text. Considering the relays used in indirect deliveries, messages are often transferred through more routers which are not trusted by the server and client. There is a possibility of altering the message in between.

So to enhance the security, we use the encrypted TLS (Transport Layer Security) connection between the client and e-mail server. TLS command can be used to encrypt the whole e-mail message. But this doesn't ensure that the message will stay encrypted along the path.

TLS is most useful when a login credentials are to be transferred. Using the STARTTLS command together along with the AUTH command is most secure way to authenticate the server.

- **SIZE**

The SIZE command is used for two purposes. Firstly, the SMTP server can inform the client the maximum message size and secondly, the client can inform the server the size of e-mail message which will be sent. The client should not send the message whose size is Larger than the maximum size given by the server. The size is always specified in Bytes.

- **HELP**

This command make the server to respond with the useful information requested by the client.

4. SMTP IMPLEMENTATION

4.1: Capturing SMTP packets by creating local smtp server(sendmail):

Fake/Dummy SMTP is an SMTP mail transfer establishment procedure where the dummy establishment will be used as a testing environment in order to avoid testing on the actual scenario of mail transfer in SMTP.

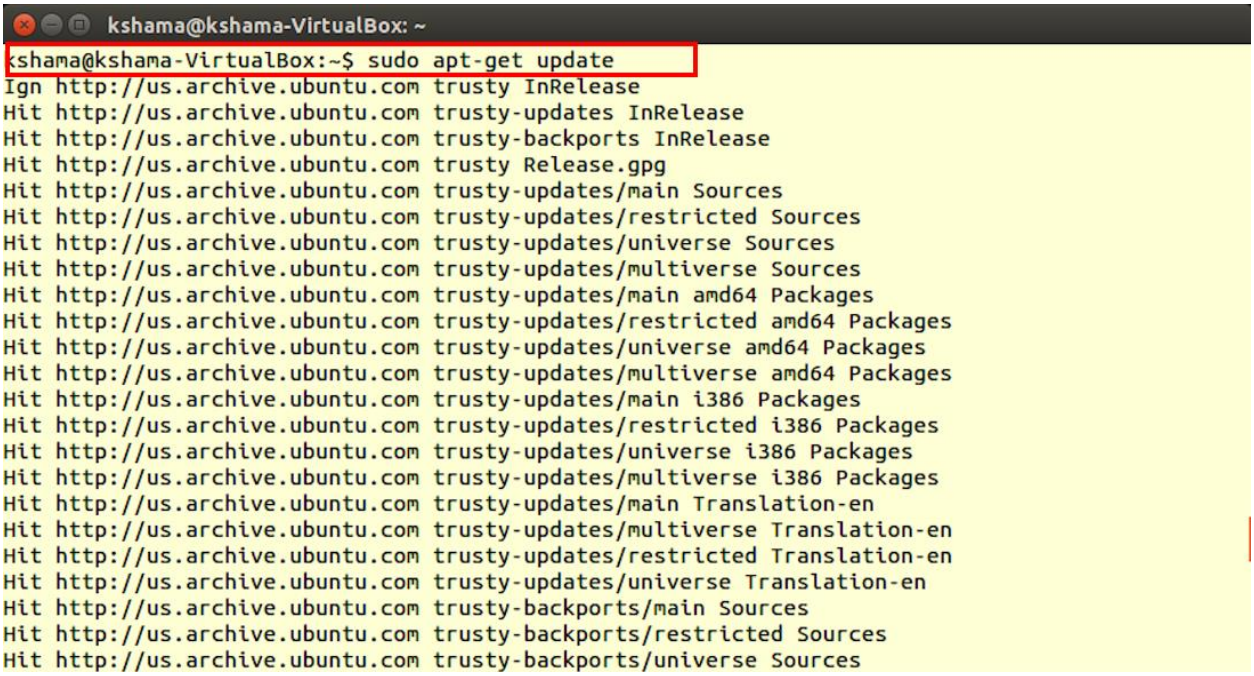
1. Setting up environment

a. First we need to install sendmail repository on Ubuntu.

How to install sendmail repository

Before proceeding with the installation we need to update the ubuntu repository. So, from a terminal session type the command line and execute by return key.

$sudo apt-get update



Then install sendmail repository by typing below command in terminal.

$sudo apt-get install sendmail

```
kshama@kshama-VirtualBox: ~
kshama@kshama-VirtualBox:~$ sudo apt-get install sendmail
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  procmail sensible-mda
Suggested packages:
  sendmail-doc rmail
The following NEW packages will be installed:
  procmail sendmail sensible-mda
0 upgraded, 3 newly installed, 0 to remove and 370 not upgraded.
Need to get 150 kB of archives.
After this operation, 847 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us.archive.ubuntu.com/ubuntu/ trusty-updates/main procmail amd64 3.22-21ubuntu0.1 [
135 kB]
Get:2 http://us.archive.ubuntu.com/ubuntu/ trusty/universe sensible-mda amd64 8.14.4-4.1ubuntu1
[8,246 B]
Get:3 http://us.archive.ubuntu.com/ubuntu/ trusty/universe sendmail all 8.14.4-4.1ubuntu1 [6,248
 B]
Fetched 150 kB in 0s (225 kB/s)
Selecting previously unselected package procmail.
(Reading database ... 183240 files and directories currently installed.)
Preparing to unpack .../procmail_3.22-21ubuntu0.1_amd64.deb ...
```

b. Configuring sendmail to start localhost smtp server.

Next, we edit the 'sendmail.mc' to configure 'sendmail.conf' ".mc" file give an easy access to edit the configuration file, because the .config is complex to exit.

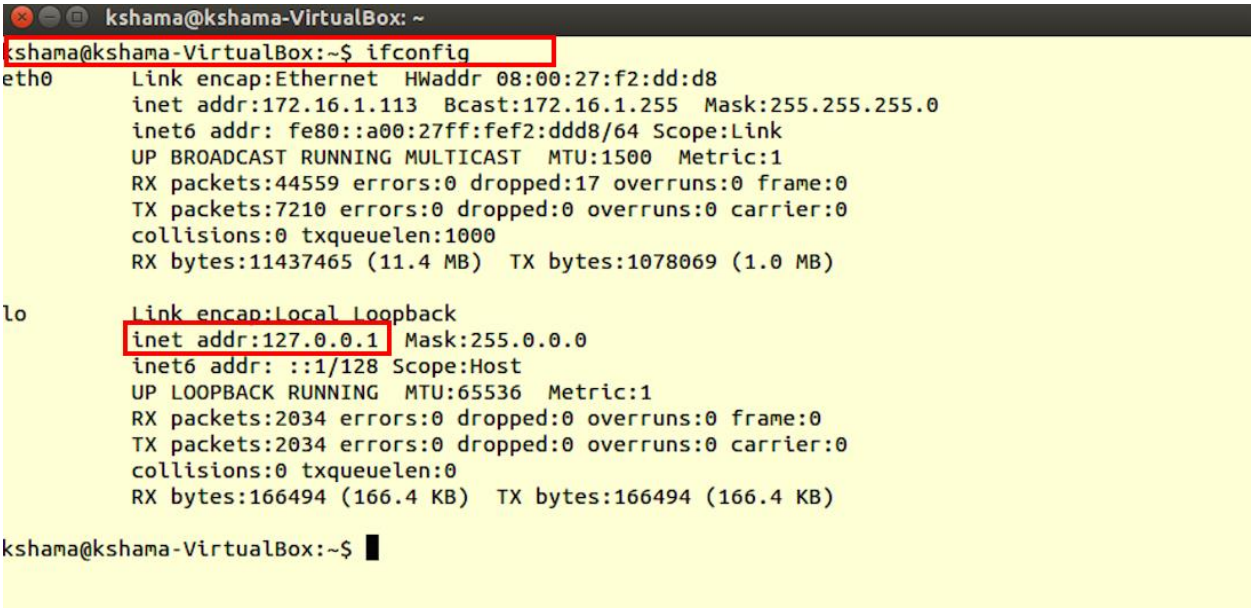
To edit sendmail.mc used below command:

$sudo gedit /etc/mail/sendmail.mc

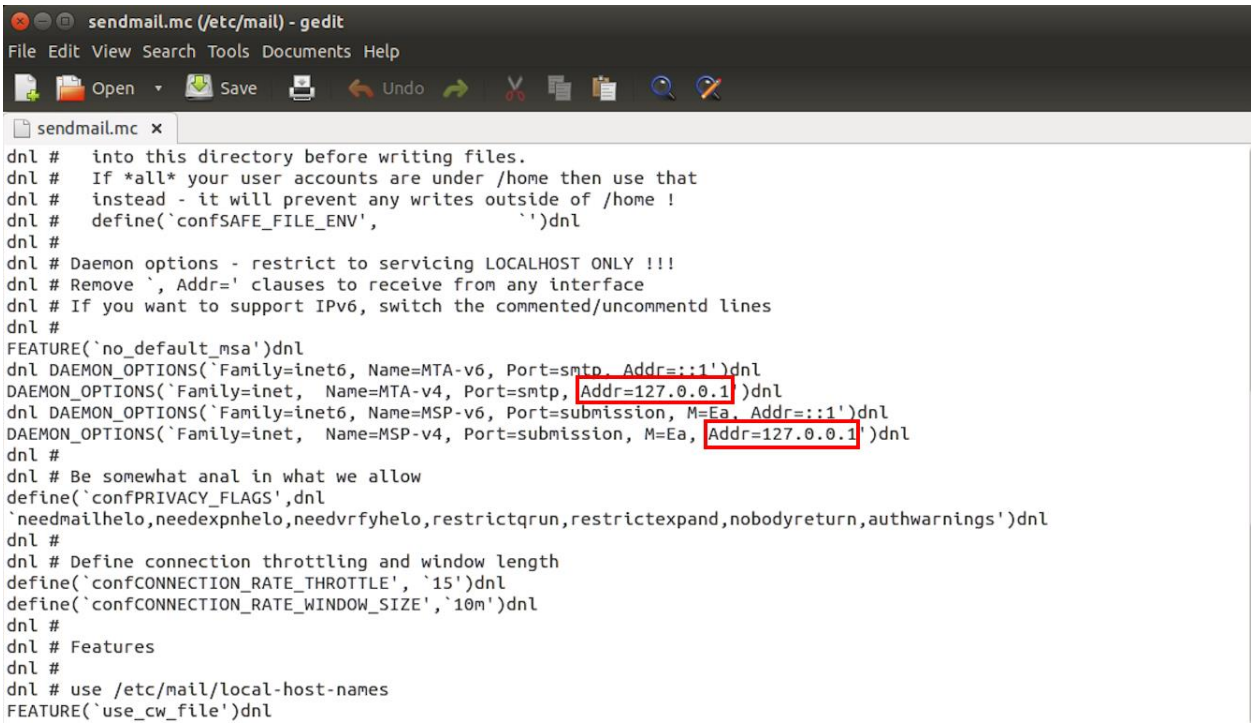
```
kshama@kshama-VirtualBox: ~
 B]
Fetched 150 kB in 0s (225 kB/s)
Selecting previously unselected package procmail.
(Reading database ... 183240 files and directories currently installed.)
Preparing to unpack .../procmail_3.22-21ubuntu0.1_amd64.deb ...
Unpacking procmail (3.22-21ubuntu0.1) ...
Selecting previously unselected package sensible-mda.
Preparing to unpack .../sensible-mda_8.14.4-4.1ubuntu1_amd64.deb ...
Unpacking sensible-mda (8.14.4-4.1ubuntu1) ...
Selecting previously unselected package sendmail.
Preparing to unpack .../sendmail_8.14.4-4.1ubuntu1_all.deb ...
Unpacking sendmail (8.14.4-4.1ubuntu1) ...
Processing triggers for man-db (2.6.7.1-1ubuntu1) ...
Setting up procmail (3.22-21ubuntu0.1) ...
Setting up sensible-mda (8.14.4-4.1ubuntu1) ...
Setting up sendmail (8.14.4-4.1ubuntu1) ...
kshama@kshama-VirtualBox:~$
kshama@kshama-VirtualBox:~$
kshama@kshama-VirtualBox:~$
kshama@kshama-VirtualBox:~$ sudo gedit /etc/mail/sendmail.mc
kshama@kshama-VirtualBox:~$
kshama@kshama-VirtualBox:~$
kshama@kshama-VirtualBox:~$
kshama@kshama-VirtualBox:~$
```

which opens up, the below window.

Note: any editor can be used like vim, gedit etc, I used gedit to edit in root mode. Before editing check the localhost address in the $ifconfig window

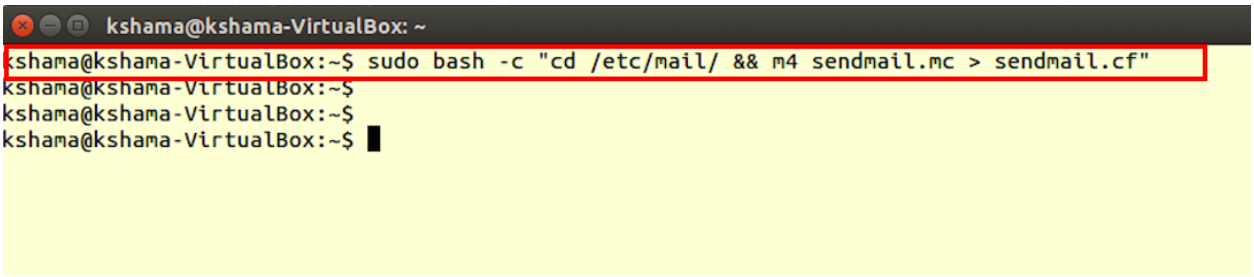


use the localhost address to edit in the .mc file as below,



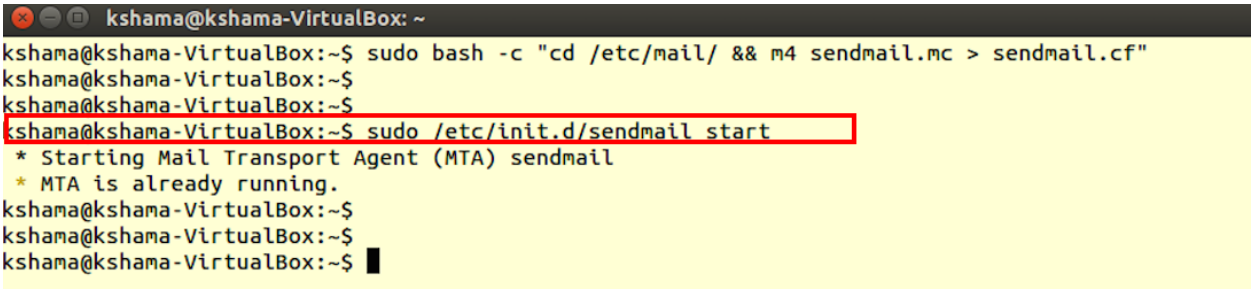
After editing save and exit from editor, then we send the configuration to ‘sendmail.config’ by using below command

$sudo bash -c "cd /etc/mail/ && m4 sendmail.mc > sendmail.cf"



Next, we start the sendmail server using below command

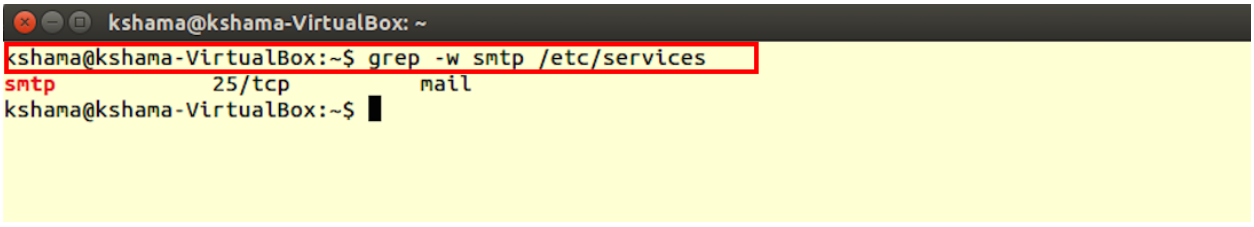
$sudo /etc/init.d/sendmail start

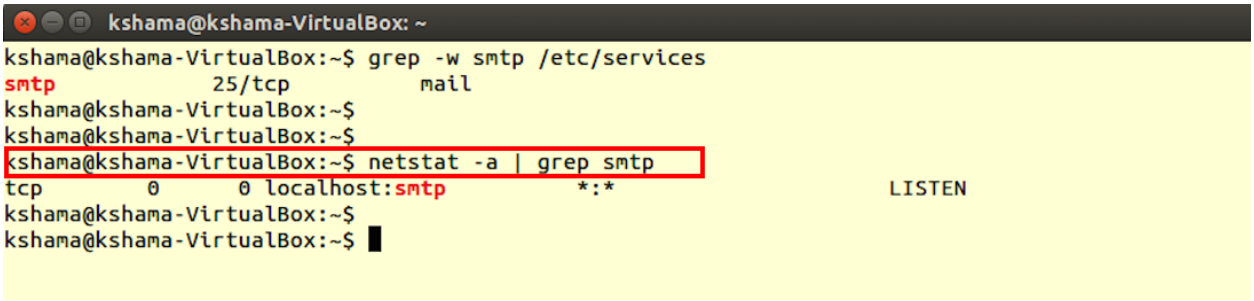


c. Checking if smtp server is running.

Now, check if the smtp server is up and running,

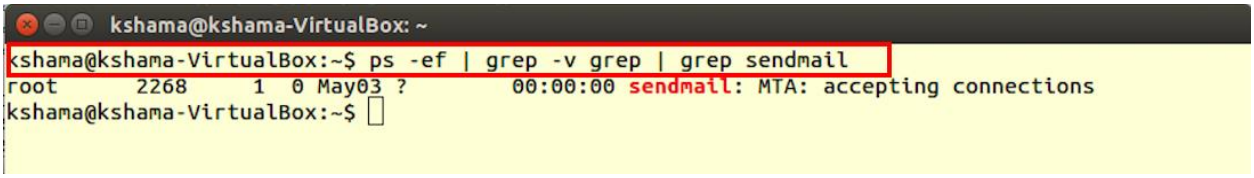
$grep -w smtp /etc/services





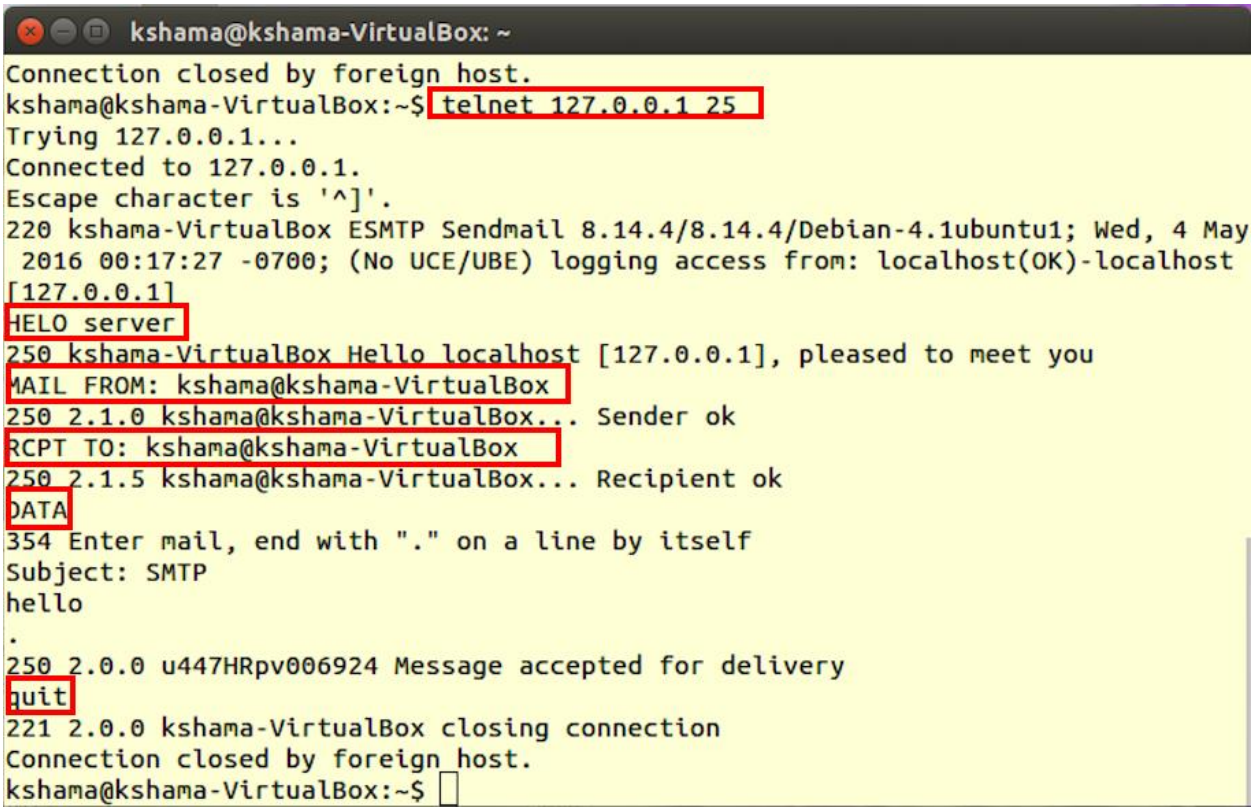
If the sendmail configuration works fine then, we can check that smtp port is “accepting connections”

$ps -ef | grep -v grep | grep sendmail



Now, the environment is setup. 2. Sending mail using telnet. Before connecting to smtp localhost server, start a wireshark capture process in root mode. To send a mail, we have used ‘TELNET’ and port number 25(smtp) $telnet 127.0.0.1 25

Now, we type in the mail attributes, HELO, MAIL FROM:, RCPT TO:, DATA and QUIT.



Now the mail has been sent.

3. Check incoming mail.

To check received mail, we used ‘mail’ repository.

To install ‘mail’ use: $sudo apt-get install mailutils

```
kshama@kshama-VirtualBox: ~
kshama@kshama-VirtualBox:~$ sudo apt-get install mailutils
[sudo] password for kshama:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  libgsasl7 libkyotocabinet16 libmailutils4 libntlm0 mailutils-common
Suggested packages:
  mailutils-mh mailutils-doc
The following NEW packages will be installed:
  libgsasl7 libkyotocabinet16 libmailutils4 libntlm0 mailutils
  mailutils-common
0 upgraded, 6 newly installed, 0 to remove and 370 not upgraded.
Need to get 1,288 kB of archives.
After this operation, 7,253 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us.archive.ubuntu.com/ubuntu/ trusty/universe libkyotocabinet16 amd
64 1.2.76-4 [288 kB]
Get:2 http://us.archive.ubuntu.com/ubuntu/ trusty/universe libntlm0 amd64 1.4-1
[15.3 kB]
Get:3 http://us.archive.ubuntu.com/ubuntu/ trusty/universe libgsasl7 amd64 1.8.0
-2ubuntu2 [117 kB]
Get:4 http://us.archive.ubuntu.com/ubuntu/ trusty/universe mailutils-common all
1:2.99.98-1.1 [245 kB]
```

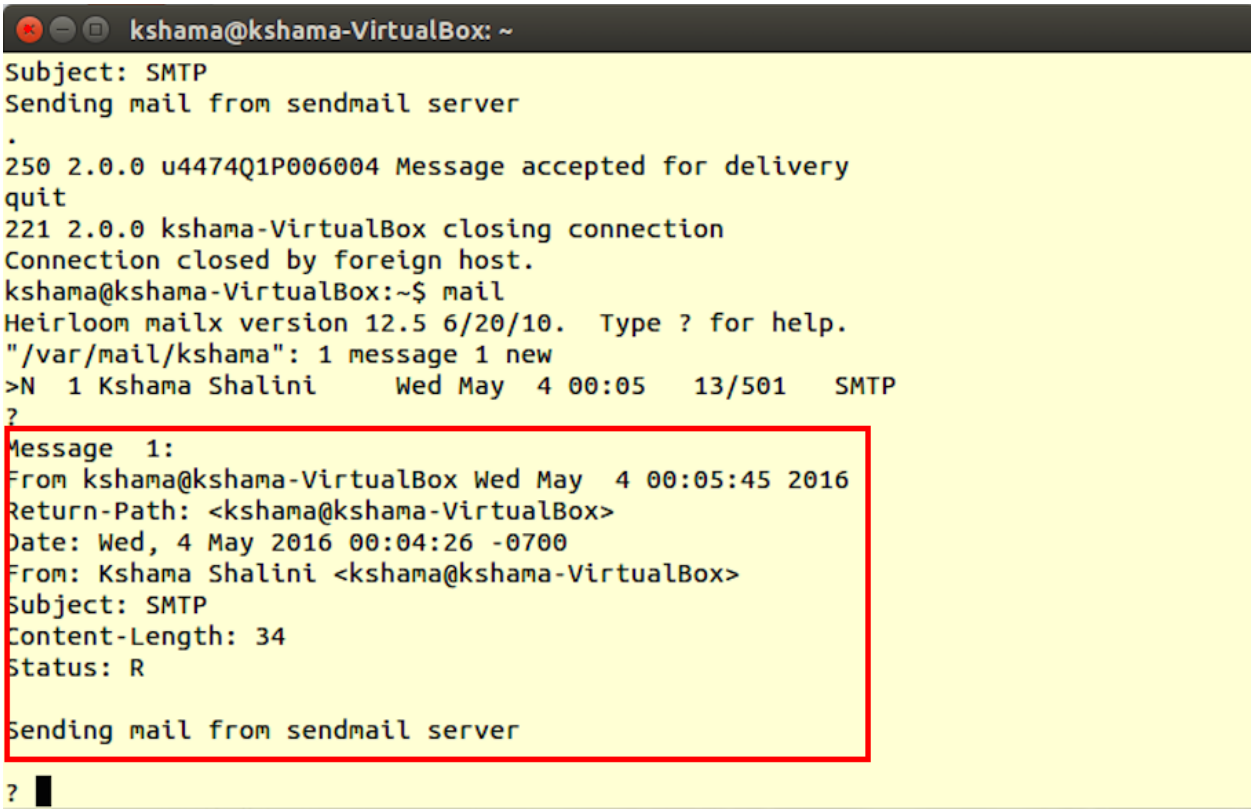
and then type $mail

```
kshama@kshama-VirtualBox: ~
kshama@kshama-VirtualBox:~$ mail
Heirloom mailx version 12.5 6/20/10.  Type ? for help.
"/var/mail/kshama": 4 messages 4 new
>N  1 Kshama Shalini     Wed May  4 00:05   13/501   SMTP
 N  2 Kshama Shalini     Wed May  4 00:12   13/477
 N  3 Kshama Shalini     Wed May  4 00:15   13/484   SMTP Mail
 N  4 Kshama Shalini     Wed May  4 00:18   13/472   SMTP
?
Message  1:
From kshama@kshama-VirtualBox Wed May  4 00:05:45 2016
Return-Path: <kshama@kshama-VirtualBox>
Date: Wed, 4 May 2016 00:04:26 -0700
From: Kshama Shalini <kshama@kshama-VirtualBox>
Subject: SMTP
Content-Length: 34
Status: R

Sending mail from sendmail server

?
Message  2:
From kshama@kshama-VirtualBox Wed May  4 00:12:02 2016
Return-Path: <kshama@kshama-VirtualBox>
Date: Wed, 4 May 2016 00:10:46 -0700
```

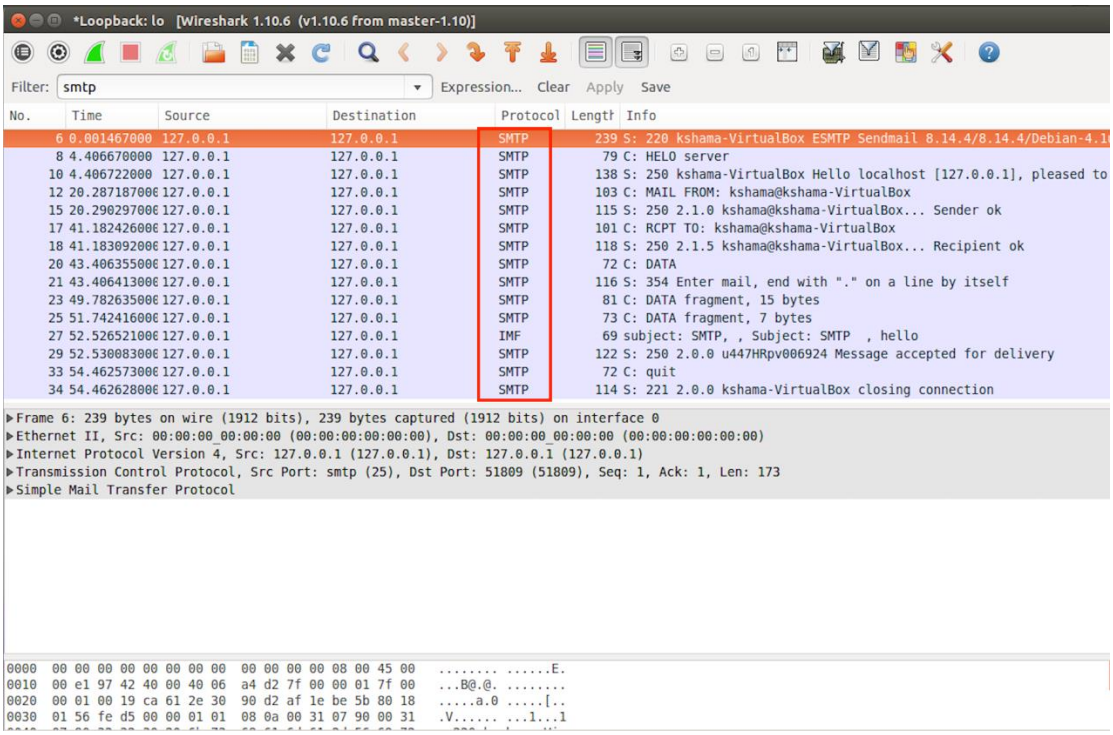
Mail is received, to read the mail press enter, or use the number beside it to open the specific mail.



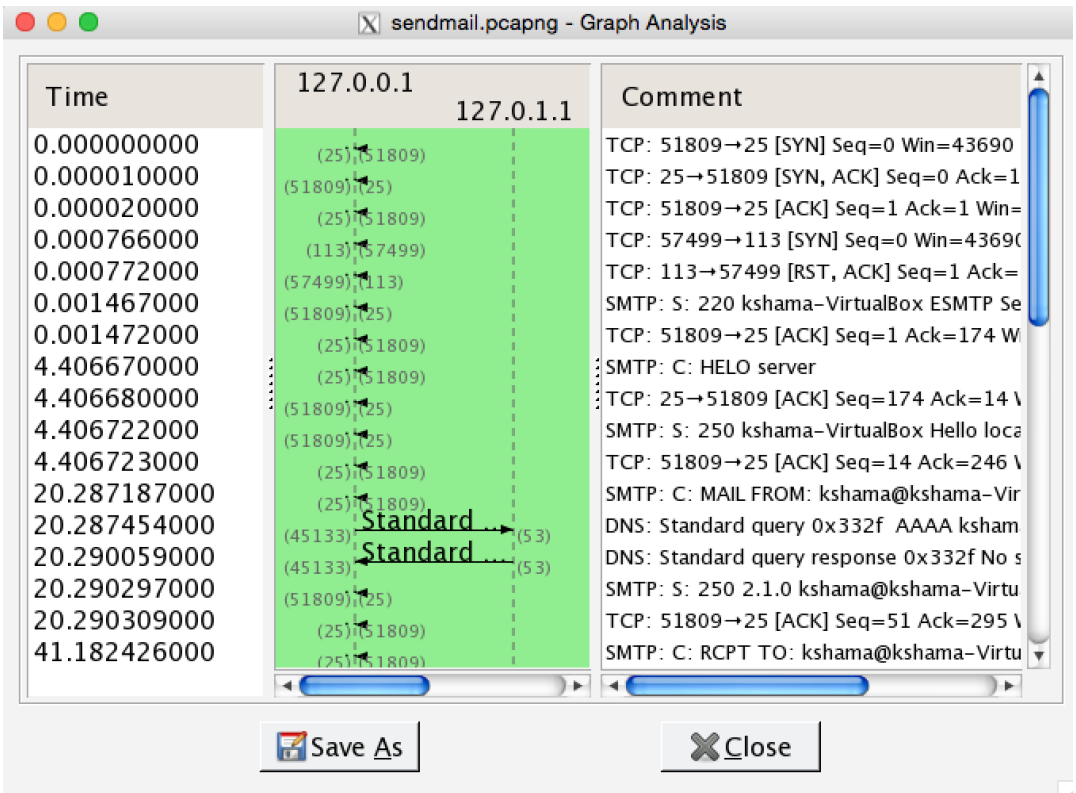
So now we have created a smtp server and sent a mail using local server and checked the mail received. Next, if we want to restart, stop or start the mail server we can use the below commands, $sudo /etc/init.d/sendmail restart

$sudo /etc/init.d/sendmail stop $sudo /etc/init.d/sendmail start

d. Now we have the wireshark capture as below



Graphical analysis



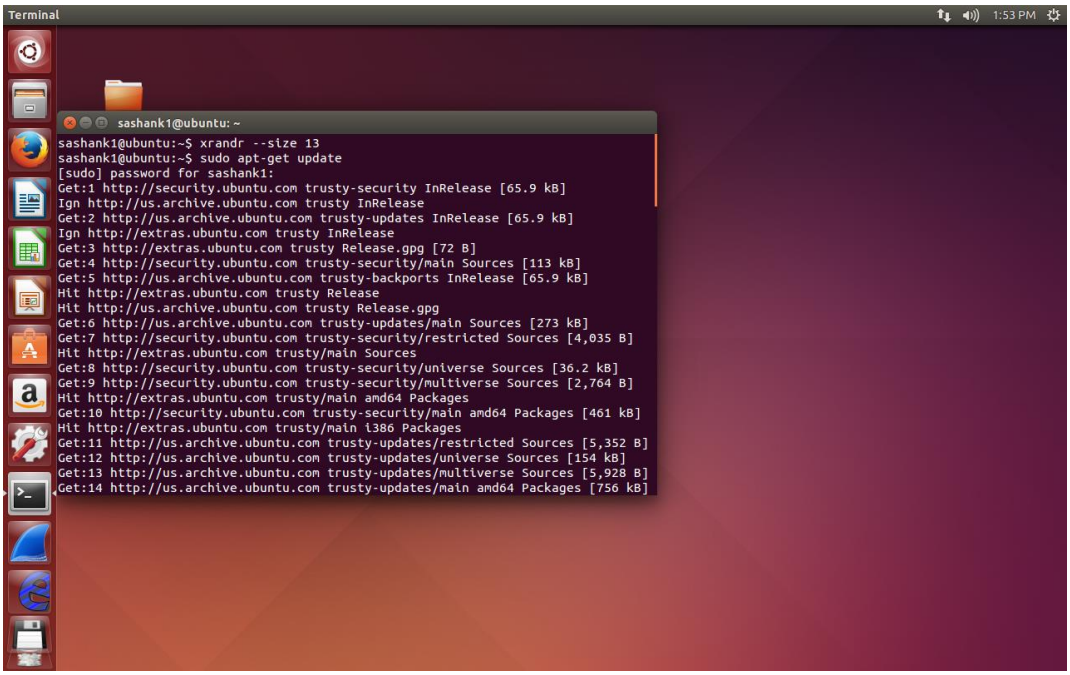
4.2. Sending mail through SSMTP

1. Environment Setup

a. Install SSMTP repository on Ubuntu

It is preferred to update to latest version of Ubuntu before installing the SSMTP. The following command update the Ubuntu repository.

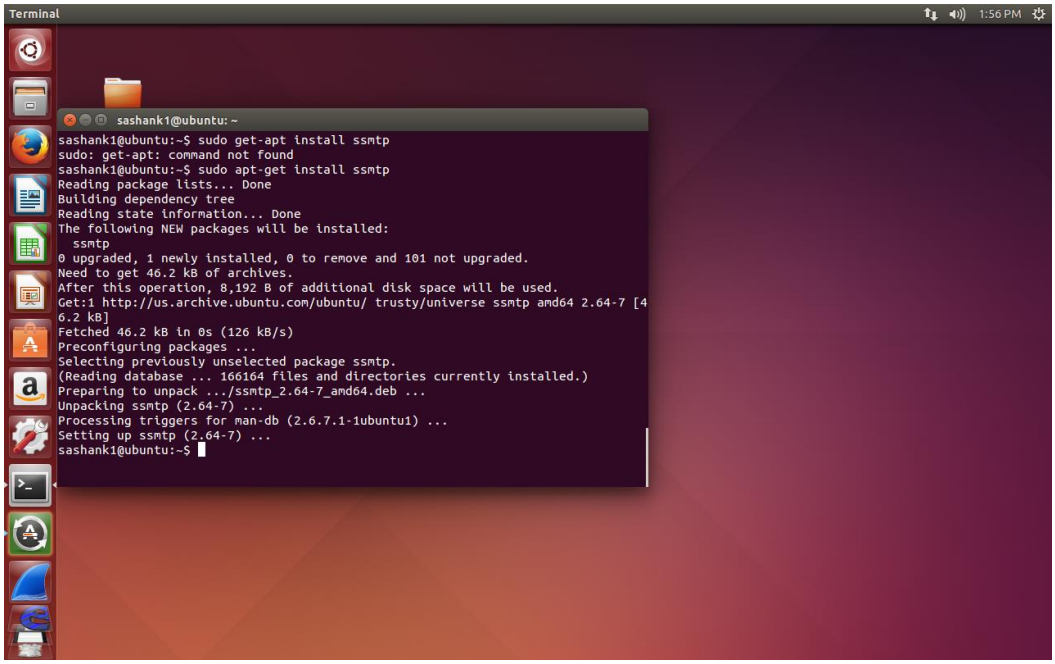
$sudo apt-get update



The above terminal indicates successful update on Ubuntu.

Now install SSMTP on update Ubuntu with the following command in terminal.

$sudo apt-get install ssmtp



b. **Configure SSMTP**

Domain utilized in sending email can be set through the configuration file. The configuration file can be found on following directory

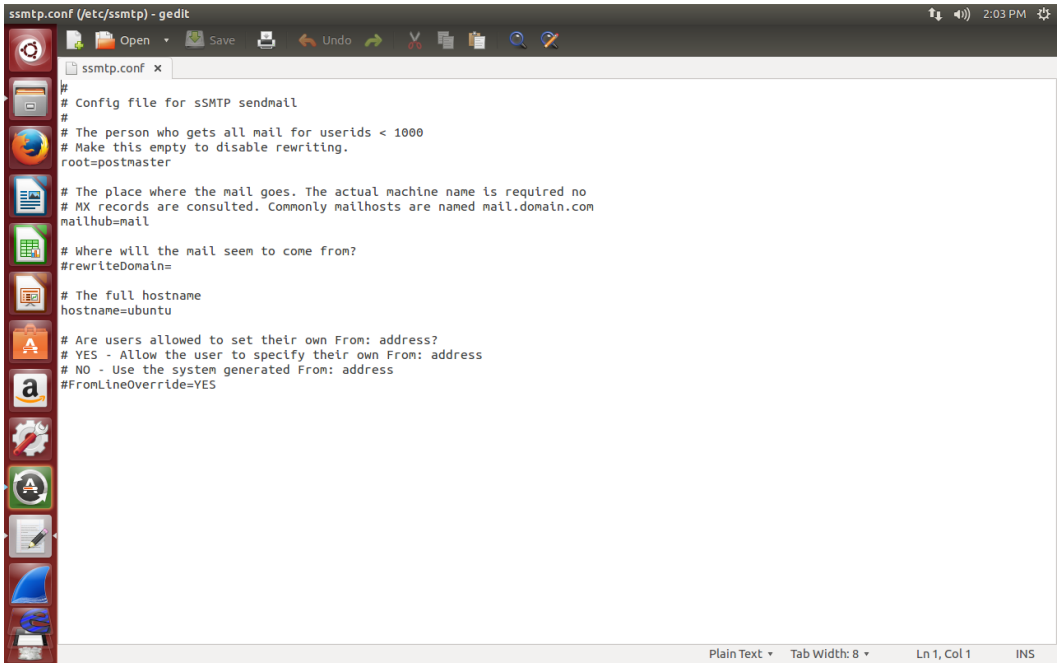
"/etc/ssmtp/ssmtp.conf"

Let us try taking Gmail as domain and send mail through the Gmail server using SSMTP. So configure the SSMTP open the config file using following command

$sudo gedit /etc/ssmtp/ssmtp.conf

Generally the config file is write protected so the entire directory os SSMTP should be changed to read and write permissions. CHMOD command is used to change the file permission settings

The following is screen shot for default SSMTP config



The following changes should be done on to config file and save it.

****

Config file for sSMTP sendmail

#

The person who gets all mail for userids < 1000

Make this empty to disable rewriting.

#root=postmaster
root=ConfigEmailAddress@
gmail.com

The place where the mail goes. The actual machine name is required no

MX records are consulted. Commonly mailhosts are named mail.domain.com

#mailhub=mail
mailhub=smtp.gmail.
com:587

AuthUser= ConfigEmailAddress @gmail.com

AuthPass=
CinfigEmailAdd
ressPasswordU
s
UseSTARTTLS=Y
ES

Where will the mail seem to come from?

#rewriteDomain=

rewriteDomain=gmail.com

The full hostname

#hostname=MyMediaServer.
home hostname=

ConfigEmailAddress
@gmail.com

Are users allowed to set their own From: address?

YES - Allow the user to specify their own From: address

NO - Use the system generated From: address

FromLineOverride=YES

****

c. Sending mail by using SSMTP

As we are using google, there are few privacy or accessibility settings involved to use gmail account for transferring the mail. Please follow the following link into your google email account:

https://www.google.com/settings/security/lesssecureapps

And set "Access for less secure apps" to ON. This gives access for external apps that is used to send mails by using your SSMTP repository.

Now we can use following commands to transfer mail to any user using configured Gmail account on SSMTP using following command on terminal

$ssmtp ConfigEmail@gmail.com

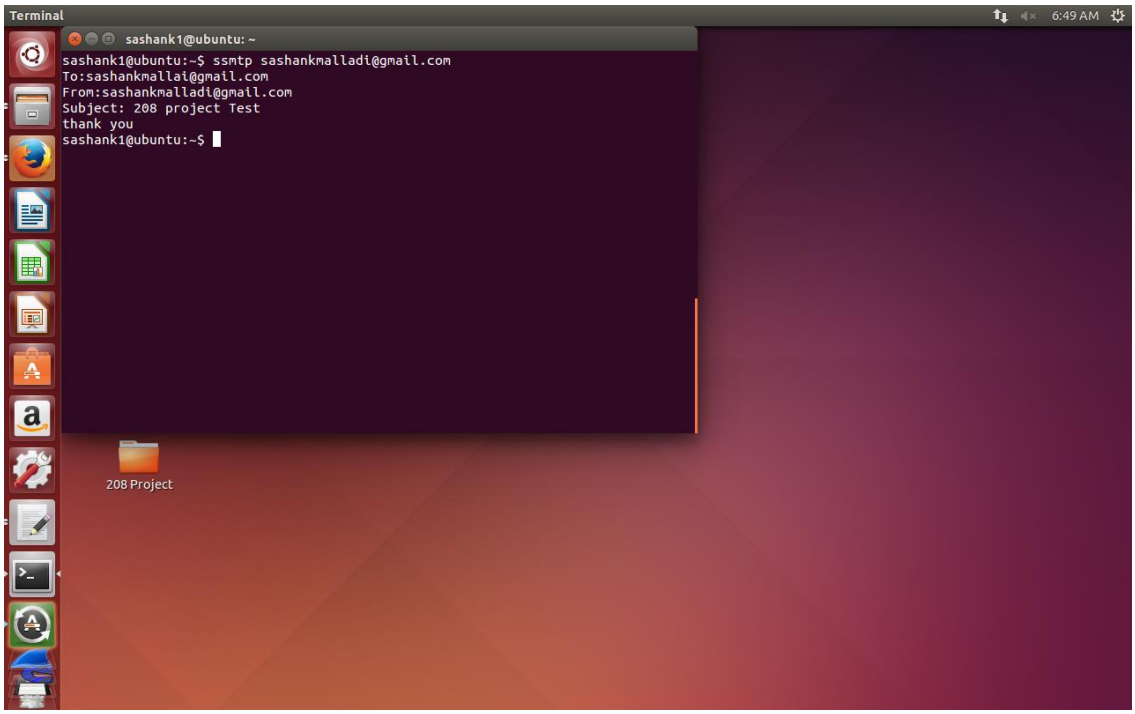
Following the above command, we should provide email format which consists of TO address, from address, Subject and Message Body. The following shows an example

To: recipient @ gmail.com

From: sender@gmail.com

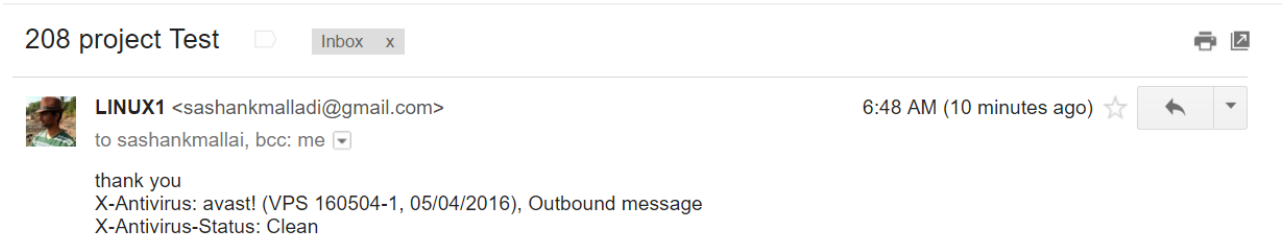
Subject: test email

Hello World!



After entering required details click on cntrl+d for execution of the command. This will use the SSTP config file and google account to route the message to the recever. After a few seconds ssmtp will send the mail and exits. From the above screen an email is sent from sashankmalladi@gmail.com to sashankmalladi@gmail.com

The following screenshot shows the email received



The above screenshot shows the mail received according to the parameters mentioned in email format.

Wireshark could be run on background to capture the packets.

4.3. Wireshark for SSMTP:

Open the wire shark with the admin permissions using following command

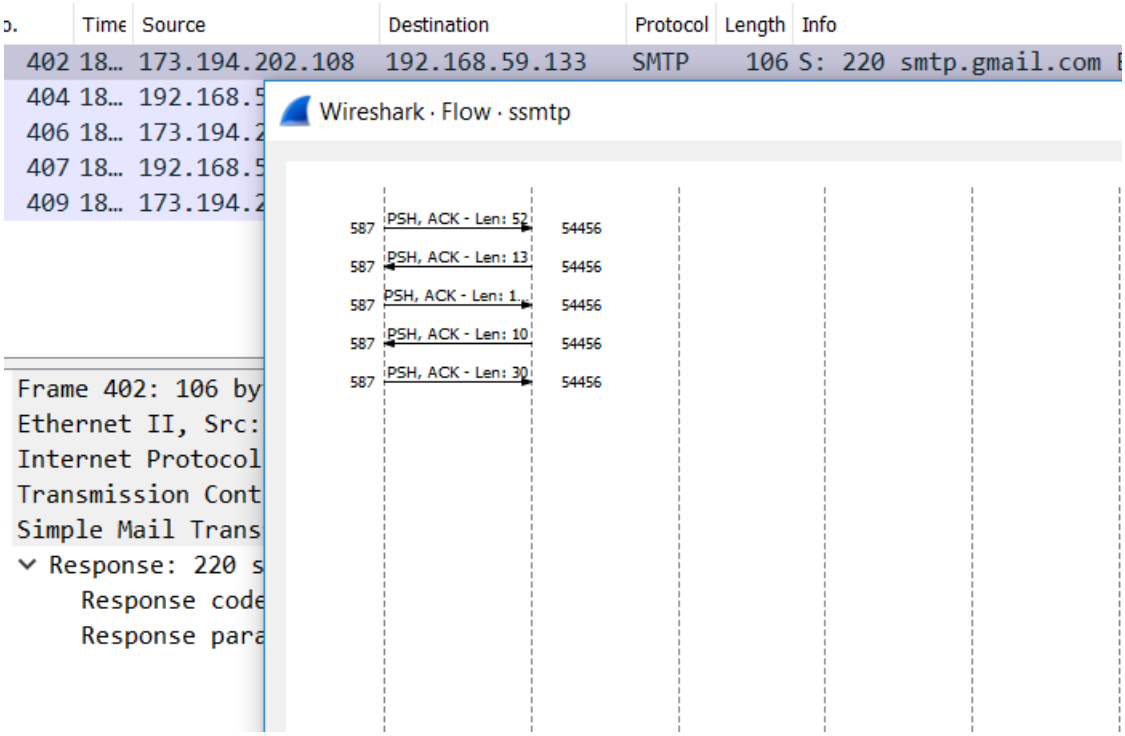
$sudo wireshark

Select the interface being utilized. In my case it is eth0 for Wi-Fi. Run the Wireshark with SMTP as filter and send mail using above mentioned process. Following captures are found on Wireshark with above mentioned steps

No.	Time	Source	Destination	Protocol	Length	Info
402	18…	173.194.202.108	192.168.59.133	SMTP	106	S: 220 smtp.gmail.com ESMTP i6sm1332662pfc.65 - gsmtp
404	18…	192.168.59.133	173.194.202.108	SMTP	67	C: EHLO ubuntu
406	18…	173.194.202.108	192.168.59.133	SMTP	209	S: 250 smtp.gmail.com at your service, [130.65.254.18] | 250 SIZE 35882577 | 250 8BITMIME | 250 STARTTLS | 250 ENHANCEDSTATUSCODES | 250 PIPELINING | 250
407	18…	192.168.59.133	173.194.202.108	SMTP	64	C: STARTTLS
409	18…	173.194.202.108	192.168.59.133	SMTP	84	S: 220 2.0.0 Ready to start TLS

```
> Frame 406: 209 bytes on wire (1672 bits), 209 bytes captured (1672 bits)
> Ethernet II, Src: Vmware_e0:57:59 (00:50:56:e0:57:59), Dst: Vmware_e8:b0:72 (00:0c:29:e8:b0:72)
> Internet Protocol Version 4, Src: 173.194.202.108, Dst: 192.168.59.133
> Transmission Control Protocol, Src Port: 587 (587), Dst Port: 54456 (54456), Seq: 53, Ack: 14, Len: 155
v Simple Mail Transfer Protocol
  v Response: 250-smtp.gmail.com at your service, [130.65.254.18]\r\n
      Response code: Requested mail action okay, completed (250)
      Response parameter: smtp.gmail.com at your service, [130.65.254.18]
  v Response: 250-SIZE 35882577\r\n
      Response code: Requested mail action okay, completed (250)
      Response parameter: SIZE 35882577
  v Response: 250-8BITMIME\r\n
      Response code: Requested mail action okay, completed (250)
      Response parameter: 8BITMIME
  v Response: 250-STARTTLS\r\n
      Response code: Requested mail action okay, completed (250)
      Response parameter: STARTTLS
  v Response: 250-ENHANCEDSTATUSCODES\r\n
      Response code: Requested mail action okay, completed (250)
      Response parameter: ENHANCEDSTATUSCODES
  v Response: 250-PIPELINING\r\n
      Response code: Requested mail action okay, completed (250)
      Response parameter: PIPELINING
  v Response: 250 SMTPUTF8\r\n
      Response code: Requested mail action okay, completed (250)
      Response parameter: SMTPUTF8
```

The following shows the flow graph for the SMTP packets captured which are displayed above



5. EXTENDED IMPLEMENTATION OF SMTP:

5.1. Need of it.

Minimum implementation criteria which is required to make the SMTP in working condition can be found from the following supporting commands,

EHLO MAIL DATA NOOP VRFY HELO RCPT

RSET QUIT

Any system that includes the SMTP server support mail relaying / delivery must support a reserved mailbox postmaster as the case insensitive local name. The requirement to accept mail for the postmaster imply that RCPT command that specify the mailbox for the postmaster at any of domains for which SMTP servers provide mail services, as well as special cases of "RCPT TO:<Postmaster>", will be supported.

Hence, by seeing all the aspects, SMTP systems are expected to make every reasonable effort to accept mail direction to the Postmaster from any of the other system on an Internet.

Transparent view of SMTP

To allow all the user composed text to become transmitted transparent, the following procedures are being incurred:

· Before sending the line of mail text, a SMTP client check a first character of line. If it is the period, one additional period will be inserted at beginning of line.

· When the line of mail text has been received by SMTP server, it checks a line. If line is composed of the single period, it is to be treated as an end of the mail indicators. If a first character is a period and there are others character on line, the first character will be deleted.

5.2. Sizes and the Timeouts pertaining to SMTP:

Size limits and the corresponding Minimum:

There are been several objects that have been required minimum or maximum sizes. Every implementation must have been able to receive object of at least these of the sizes. Extensions to the

SMTP may involve the use of characters that occupy more than the single octet each. This section hence specifies lengths in the octets where a absolute lengths, rather than character counts, have been.

Local part of it:

The maximum total length of the user name / other local-part is 64 octet.

Domain:

Maximum total length of the domain name / number is 255 octet.

Path:

Maximum total length of a reverse-path / forward-path is 256 octet (including the punctuation and the element separators).

Command Line:

Maximum total length of the command line including a command word and <CRLF> is 512 octet. SMTP extensions may be used to the increase limit.

Reply Line:

Maximum total length of the reply line including a reply code and a <CRLF> is 512 octet. More information may be actually conveyed through a multiple-line replies.

Text Line:

Maximum total length of the text line including a<CRLF> is 1000 octet (not counting the leading dot duplicated provided for the transparency). This number may be increase by an use of SMTP Service Extension.

Message Content:

Maximum total length of the message content (including any message header section as well as an message body) must be at least 64K octet.

Receiving buffer:

Minimum total number of recipients that MUST be buffered is the 100 recipient. The general principle that a relaying SMTP server would not, and the delivery SMTP servers should not, perform validation test on the message header fields suggest that message should not be rejected based on a total number of recipients shown in an header fields.

Treatment When Limit is Exceeded:

Errors due to the exceeding these limit may be well reported by using an reply code. Some example of reply codes are:

500 implies Line long./

501 implies Path long/

452 implies many recipients /

552 implies much mail data.

5.3 Timeouts in the SMTP implementation

An SMTP client MUST provide the timeout mechanism. It would use per command timeouts rather than the somehow trying to catch the time the entire mail transaction. Timeouts would be easily reconfigurable, probably without recompiling an SMTP code. Based on extensive experience with the busy mail relay hosts, an minimum per-command timeout values shall be as following:

Initial 220 Message being the timeout of 5 Minutes

An SMTP client processes need to be distinguished between a failed TCP connection and an delay in the receiving initial 220 greet message. So Many SMTP server accepts the TCP connection but delay delivery of an 220 message until the system loads permits more mail to become processed.

MAIL Command being timeout of 5 Minutes

RCPT Command being timeout of 5 Minutes

The longer timeout is required if the processing of mailing lists and the aliases is not being deferred until after an message was being accepted.

DATA Initiation having a timeout of 2 Minutes.

This is when while awaiting an 354 Start Input taken as reply to an data command.

Data Block having a timeout of 3 Minutes.

This is while awaiting an completion of each of the TCP SEND calling transmitting the chunk of the data.

DATA Termination having a timeout of 10 Minutes.

This is while awaiting an 250 OK reply. When a receiver get the final period terminating an message data, it typically perform the processing to deliver a message to the user mailbox.

Server Timeout having a timeout of 5 Minutes.

An SMTP servers should have the timeout of at least 5 minutes while is awaiting the next line command from a sender.

Strategies for Retry operation:

An exact structure would vary depending on an needs of a users on host and a number and the size of mailing lists supporting by the host. We described several optimization that were been proved helpful, particularly for the mailer support high traffic level. Any queuing strategies must included timeout on all activity on the per-command base. A queuing strategy would not send any error messages in the response to an error messages under any of the circumstances.

Strategy involving Sending Operation:

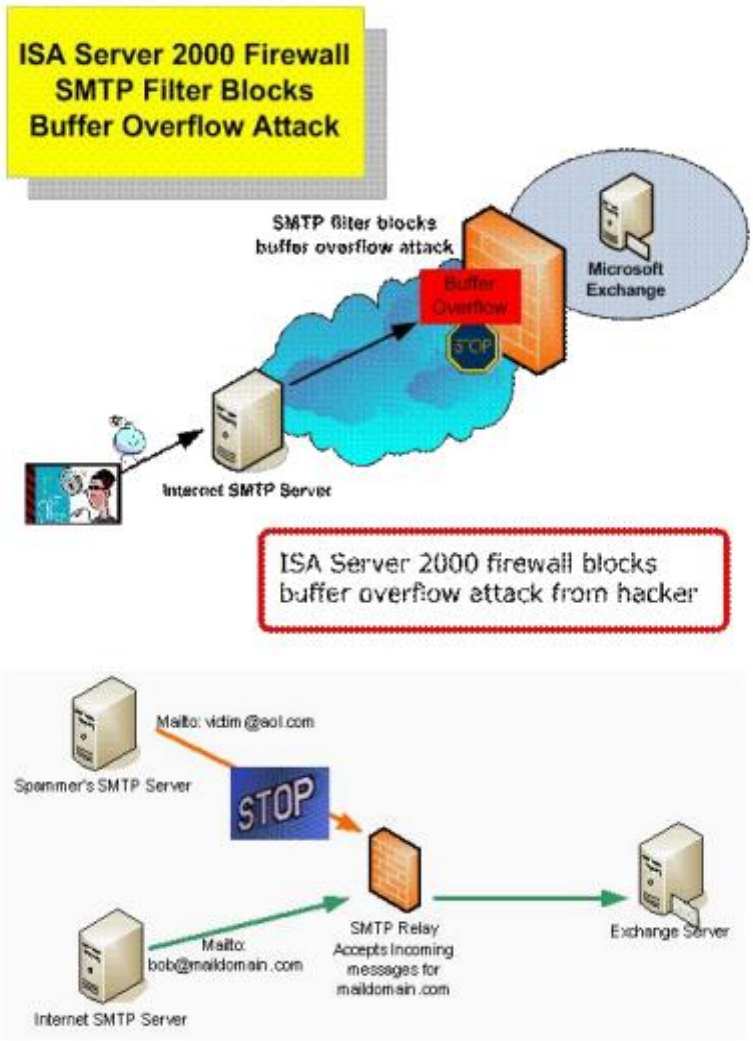
The general model for the SMTP client is the one / more process that can periodically attempts to transmit an outgoing mails. In the typical system, a program that composes of an message has some methods for requesting immediate attentions for the new pieces of outgoing mails, while mails that cannot become transmitted immediately must have been queued and the periodically retried by a sender. The mail queue entry will include not only an message itself but also an envelope information.

Strategy with Receiving Scenario:

The SMTP server should attempts to keep the pending listen on a SMTP port (specified by IANA as from port 25) at all the times. Here requires an support of the multiple incoming TCP connection for an SMTP. Some limits may be imposed, but server that cannot handle more than one of the SMTP transaction at a time is not in the conformance with an intent of this specifications.

5.4 Security Issues involved in SMTP

The process of retrieving an e-mail from the servers and managing data communication through the Internet are being vulnerable to various kinds of different attacks. (CVE)Common Vulnerabilities and Exposures organization provided the list of standardized names for SMTP vulnerabilities and other information security exposures. The vulnerability problems could be grouped into a several general high-risk as various categories being:: Buffer overflow; host shell gaining attacks, bounced from piping attacks and redirection attacks through from the firewall. The medium to high risk categories are to be included by the denial-of service attack. Low--to--medium-risk that are included as categories included as mail--queue manipulation--attacks, crashing antivirus—software--attack, debug--mode--leak category, and mail relaying on remote SMTP server way. Most SMTP specific vulnerabilities have been occur from misapplied else the unapplied patches related to the sendmail installations else the misconfigured sendmail daemon in SMTP server. ISP restricts accesses to outgoing mail server to the provide better service to the customers, as well to prevent the spam from to be sent through the mail server. There are many several methods for the establish restrictions that can result in denying users access to outgoing mail server. Originally from the scratch, e-mail server didn't verify claimed sender identities and will simply passes the mailing on with the whatever returns address have been specified. Bulk mailers definitely will be taken advantage for sending huge volume of the mail with bogus returns with the addresses. This results in the deep slow down in server. To fix this problem, this origins of the spam email shall be identified. A email messages typically transport through the set of the SMTP server (including sender's and receiver's server) before reaching destination hosts. Along this, messages gets stamped by intermediate middle SMTP servers. The stamp releases track information that could be identify in mail header. Mismatch between the various IP address and the main domain name in the very header could unveil real source of the spam mail. The real domain names that will correspond to an indicated IP addresses could be found out by executing the reverse DNS lookup scenario. Modern mail programs have been incorporated this kind of functionality, which generates the received header line that has included the identity of this attacker for protection and very security purposes, companies may have configure the SMTP servers and well other email services systems in such a manner that any of mail coming from RBL blacklisted mail servers will be automatically rejected.



The attacks in the SMTP system and the corresponding intrusion action taken by the attacker are discussed in the following table.

Stack-based buffer overflow in SMTP services support.	It allows the remote attackers to execute arbitrary code via the long RCPT TO arguments.
SMTP proxy in Symantec Enterprise Firewall which includes firewall's physical interface name and the address in an SMTP exchange when a NAT translation is being made to an address other than firewall.	It allows the remote attackers to determine a certain firewall configurational information explicitly.
SMTP service in Microsoft Windows 2000, Windows XP Professional, and Exchange 2000 to the cause a DoS via the command with the malformed data transfer request.	An attacker may disrupt the SMTP service and, when depending on a system configuration, potentially IIS and when other Internet services as well .

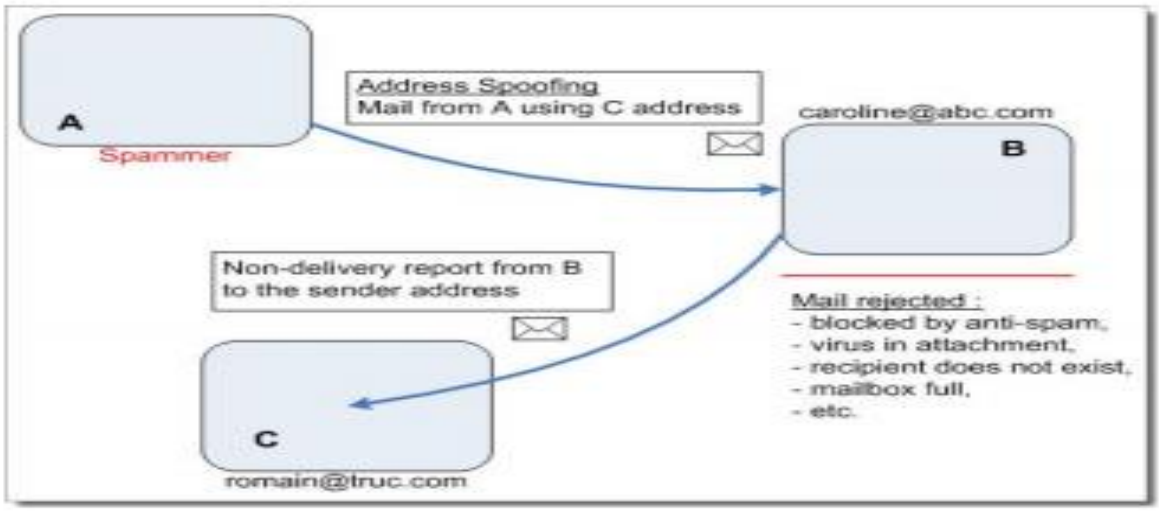
SMTP service in the Microsoft Windows 2000 as well as the Internet Mail Connector in an Exchange Server doesn't properly handle a responses to the NTLM authentication.	It allow remote attackers to perform the mail relaying via a SMTP AUTH command by using null session credentials.
Vuln local buffer overflows on sendmail will cause a session to terminate.	It allows the remote attackers to cause an DoS (memory exhaustion) with generating a large number of SMTP error, which forces an SMTP session log to grow too much large.
Vulnerability in the SMTP proxy with the WatchGuard Firebox	A remote attacker may well bypass firewall filtering via the base64 MIME encoded -mail attachment whose boundary names end in two -- dashes.
Formatting string vulnerability in the Exim (v. 3.22-10 in the Red Hat	It allows the remote attacker to execute arbitrary code via format strings in the SMTP mail headers.
The authentication error on the remote SMTP server	An attacker may be exploit this flaw as to use the SMTP server as an spam relay.

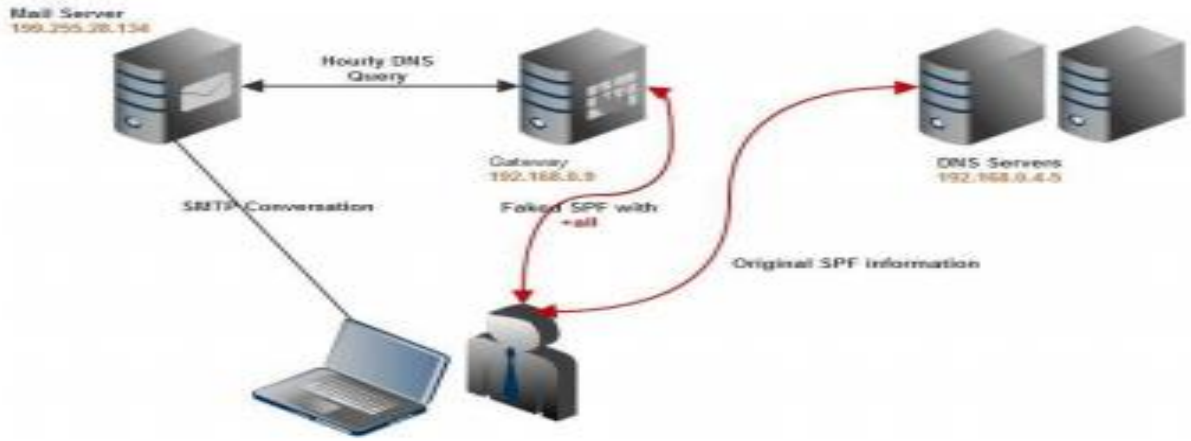
.	
Lotus Domino SMTP server is the vulnerable to the DoS (central processing unit consumption) attack by forging the e-mail message.	It allows a remote attacker to cause a DoS: a server enters a mail loop.
The Lotus Domino SMTP is very vulnerable to buffer overflow when supplied a too long ENVID variable.	An attacker may use the flaw to prevent Domino service from its working properly, / to execute arbitrary code on a host.
The mailguard feature in Cisco Secure PIX Firewall does not properly restrict access to SMTP commands	It allows the remote attackers to execute an restricted commands by sending the DATA command before sending an restricted command.
An remote mate SMTP server crashes when it is in issued an HELO command with argument longer than the 1,200 characters.	Van attacker may will shut down an SMTP server
Buffer overflow on the IT House mail servers	Remote attackers may well execute arbitrary

	commands via the long RCPT To the mail command.
Buffer overflow in remote Lotus SMTP server the server is to issued the long argument to MAIL FROM the command.	An attacker may prevent a host from its acting as the mail host and may execute arbitrary code on a system.
mail.local in a remote Sendmail server does not properly identify the .\n string, which indicates the message-text end.	A remote attacker may cause the DoS corrupt mailboxes via the message line that has 2047 characters long enough.
Super Mail Transfer Package as later be called MsgCore, has the memory leak.	Remote attackers may cause the DoS by repeating multiple HELO MAIL FROM, RCPT TO, as well as DATA commands in same session.
An remote Sendmail SMTP server didn't complain when issued with the command	A attacker may send mails that will be bounced with to a program that allow to execute arbitrar command on a host.

The remote Sendmail SMTP server seems to pipe mail sent to decode alias to the program.	An attacker can use this decode flaw to overwrite arbitrary files on the remote server.
Multiple integer overflow.	An attacker may have execute arbitrary code on the host by sending an specially crafted ASN.1 encoded packet with the improper lengths
Exim MTA heap overflow.	An attacker may gain a shell on this host.
Exchange remote buffer overflows, SMTP service is an vulnerable to the flaw.	An attacker may complete crashed Exchange 5.5 and executed arbitrary code on a Exchange 2000 value.
Remote Sendmail servers belong have overflow on the remote buffer.	An attacker may gain many root privileges which are costly.
Remote Postfix daemon multiple Vulnerabilities.	An attacker may remotely disable it, / use it as the DoS agent against the arbitrary hosts

LMail SMTP server experience various Overflow.	A cracker well execute arbitrary commands on the host or to disable it remote value.
Endmail server have buffer overflow due to the type conversion.	An attacker might gain remotely root privilege.
Remote header buffers overflows on the Sendmail servers.	A remote attacker might gain root privilege.





6. SMTP IMPORTANCE

The most basic function or the main function of the SMTP is to push the emails. It means that it cannot pull them from the server for which we need to Post Office Protocol

The outgoing mails server protocol help the servers communicate with one other and facilitate the delivery of email message.

SMTP usually functions in two ways:

1) In the first case, it will verify the configuration of the computer from where the email will be sent and will grant it access.
2) In the second case, it will send out the message which will then follow a successful delivery of that email. If the email fails to reach its destination it will be returned to the sender or will bounce back.

For sending images and other file attachments we use an enhanced version of the SMTP called the Extended SMTP or ESMTP.

7. SMTP DEDUCTION POINTS FROM IMPLEMENTATION

1) The SMTP deduction gives us a fixed and correct path from the end user (sender) to the authentic user which will be receiving the email.
2) We see that how easily the information and messages travel within the communication platforms

3) One of the most important things that we learnt was that how a real mail transfer can be compared with a dummy communication setup and the ways in which the setup can be improvised based on the real SMTP scenario setup.
4) We realized that SMTP provides an option for the dedicated server where important messages are handled and how the email can be modified.
5) Also, we came to know how fast can the mail be transferred from one point to another using the protocol.

8. LEARNING OUTCOMES

The main part of the report is spent on explaining the SMTP and the SSMTP. We see how the email is send from one user(sender) to another(Receiver). We divide the process into various parts, sendmail being one of them. We use a simple telnet command to then filter out the SMTP protocol using Wireshark. We try to use a real life email exchange and use a simulator to show the intricacies of the SMTP process by comparing the two. Also, we see how the email can be modified. Lastly, we tried to learn about the security issues that may face us when we receive mail and also how to check for spam using wireshark.

9. CONCLUSION

From the above report we can easily conclude that the sole purpose of SMTP is not just sending mails from client programs. In some cases, the backup and the synchronization programs have the permission to send complete profile to an email address. Utility is given to backup data and synchronize files to an email server like Gmail and Hotmail. Also, we saw the complete process and the programs inside the SMTP protocol the email goes through before it is delivered. And by studying the SMTP, we can better explain how the emails are sent on the internet servers.

10. REFRENCES

1) http://www.omnisecu.com/tcpip/smtp-simple-mail-transfer-protocol-client-server-communication.php

2) http://www.ietf.org/rfc/rfc1870.txt

3) https://www.hmailserver.com/documentation/latest/?page=whatis_pop3imapsmtp

4) http://freesoft.org/CIE/Topics/94.htm

5) http://www.comptechdoc.org/independent/networking/guide/netmail.html