

COLLEGE OF ENGINEERING, TRIVANDRUM

EXPERIMENT 16

Network Programming Lab

Author:
Alan Anto

Registration Number :
TVE16CS09

April 24, 2019

Contents

1	Simple Mail Transfer Protocol	2
1.1	Aim	2
1.2	Theory	2
1.2.1	SMTP	2
1.2.2	Working	3
1.3	Program	3
1.3.1	SMTP Server	3
1.3.2	SMTP Client	5
2	Output	5
2.1	SMTP Server	5
2.2	SMTP Client	6
3	Result	6

1 Simple Mail Transfer Protocol

1.1 Aim

Implement Simple Mail Transfer Protocol

1.2 Theory

1.2.1 SMTP

MTP is a push protocol and is used to send the mail whereas POP (post office protocol) or IMAP (internet message access protocol) are used to retrieve those mails at the receiver's side.

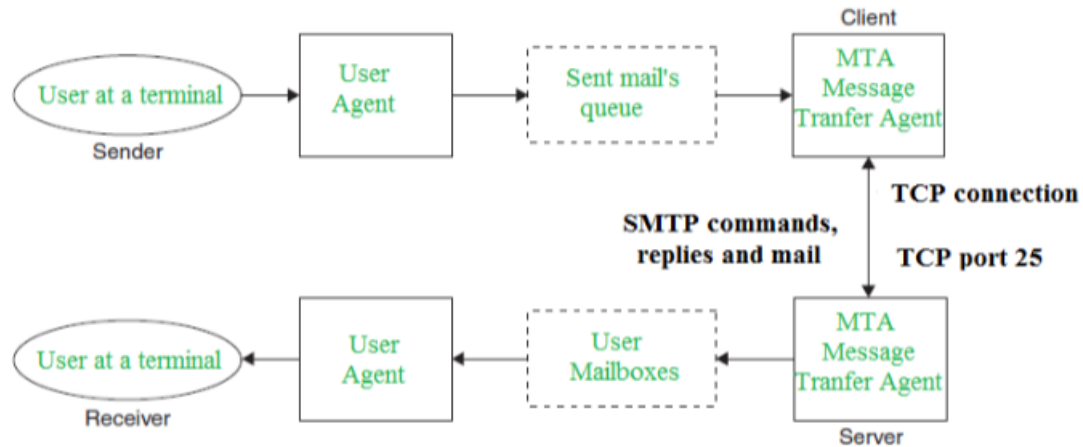
SMTP is an application layer protocol. The client who wants to send the mail opens a TCP connection to the SMTP server and then sends the mail across the connection. The SMTP server is always on listening mode. As soon as it listens for a TCP connection from any client, the SMTP process initiates a connection on that port (25). After successfully establishing the TCP connection the client process sends the mail instantly.

The SMTP model is of two type :

1. End-to- end method
2. Store-and- forward method

The end to end model is used to communicate between different organizations whereas the store and forward method is used within an organization. A SMTP client who wants to send the mail will contact the destination's host SMTP directly in order to send the mail to the destination. The SMTP server will keep the mail to itself until it is successfully copied to the receiver's SMTP.

The client SMTP is the one which initiates the session let us call it as client- SMTP and the server SMTP is the one which responds to the session request and let us call it as receiver-SMTP. The client- SMTP will start the session and the receiver-SMTP will respond to the request.



1.2.2 Working

1. SMTP server is always on a listening mode.
2. Client initiates a TCP connection with the SMTP server.
3. SMTP server listens for a connection and initiates a connection on that port.
4. The connection is established.
5. Client informs the SMTP server that it would like to send a mail.
6. Assuming the server is OK, client sends the mail to its mail server.
7. Client's mail server use DNS to get the IP Address of receiver's mail server.
8. Then, SMTP transfers the mail from sender's mail server to the receiver's mail server.

1.3 Program

1.3.1 SMTP Server

```
import socket
import datetime
s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind(("127.0.0.1",8080))
```

```
x=[" alananto@gmail.com", " hello@gmail.com", " example@gmail.com"]
s.listen(5)
conn, addr=s.accept()
while 1:
    mail=conn.recv(1024)
    tok=mail.split()
    print(mail)
    if tok[0]=='HELO':
        conn.send("250 mail.example.org")
    elif tok[0]=='MAILFROM: ':
        flag1=0
        for i in x: #checking if mail is in the list
            if tok[1]=="<"+i+">":
                conn.send("250 Sender "+tok[1]+" OK")
                flag1=1
                break
        if flag1==0:
            conn.send("421 Service Unavailable")
    elif tok[0]=='RCPTTO: ':
        flag2=0
        for i in x: #checking if mail is in the list
            if tok[1]=="<"+i+">":
                conn.send("250 Recipient "+tok[1]+" OK")
                flag2=1
                break
        if flag2==0:
            conn.send("421 Service Unavailable")
    elif tok[0]=='DATA':
        flag3=0
        if flag1==1 and flag2==1:
            flag3=1
            conn.send("354 Go Ahead, Enter data ending with
<CRLF>.<CRLF>")
            break
        else:
            conn.send("421 Service Unavailable")
```

```
if flag3==1:
    buff=conn.recv(2048)
    print("Message: "+buff)
    conn.send("250 OK; Message Accepted")
mail=conn.recv(1024)
tok=mail.split()
if tok[0]=='QUIT':
    conn.send("221 mail.example.org")
```

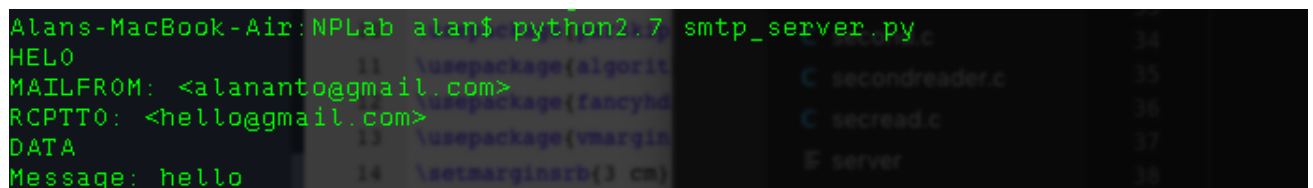
1.3.2 SMTP Client

```
import socket
import sys
s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect(("127.0.0.1", 8080))
print("Waiting to connect...")
while 1:
    comm=raw_input()

    s.send(comm)
    print("Server: "+s.recv(1024))
    if comm=='QUIT':
        s.close()
        break
```

2 Output

2.1 SMTP Server



```
Alans-MacBook-Air:NPLab alan$ python2.7 smtp_server.py
HELO
MAILFROM: <alananto@gmail.com>
RCPTTO: <hello@gmail.com>
DATA
Message: hello
250 OK; Message Accepted
```

2.2 SMTP Client

```
Alan@Alan-MacBook-Air:~$ python2.7 smtp_client.py
Waiting to connect...
HELO mail.example.org
Server: 250 mail.example.org
MAILFROM: <alananto@gmail.com>
Server: 250 Sender <alananto@gmail.com> OK
RCPTTO: <hello@gmail.com>
Server: 250 Recipient <hello@gmail.com> OK
DATA
Server: 354 Go Ahead, Enter data ending with <CRLF>.<CRLF>
hello
Server: 250 OK; Message Accepted
QUIT
Server: 221 mail.example.org
```

3 Result

SMTP server and client was implemented using python and the required output was obtained.