Simple Mail Transfer Protocol

Aim

Simulation of SMTP(Simple Mail transfer Protocol) using UDP.

Theory

SMTP is part of the application layer of the TCP/IP protocol. Using a process called ”store and forward,” SMTP moves your email on and across networks. SMTP makes use of a set of commands to transfer information via the client and server. Some of the important ones include:

- HELO: The HELO command is used to initiate an SMTP session.
- MAIL FROM: command is used primarily to send email addresses, it needs a way to alert the recipient host to who is sending the inbound message.
- RCPT TO: command tells the receiving host the email address of the message recipient
- DATA: When the sending host transmits the DATA command, it tells the receiving host that a stream of data will follow.
- QUIT: QUIT command is used to terminate an SMTP session. Based on similar information regarding SMTP, the basic intent here is to simulate how SMTP works to move your mail on and across networks.

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.

Code

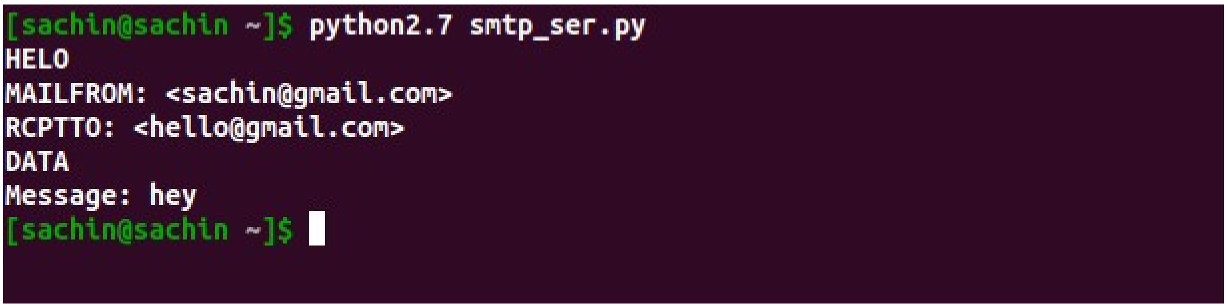
smtp_ser.py

```
import socket
import datetime
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.bind(("127.0.0.1",5001))
x=["sachin@gmail.com","hello@gmail.com","okayyy@server.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.com")
        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")
```

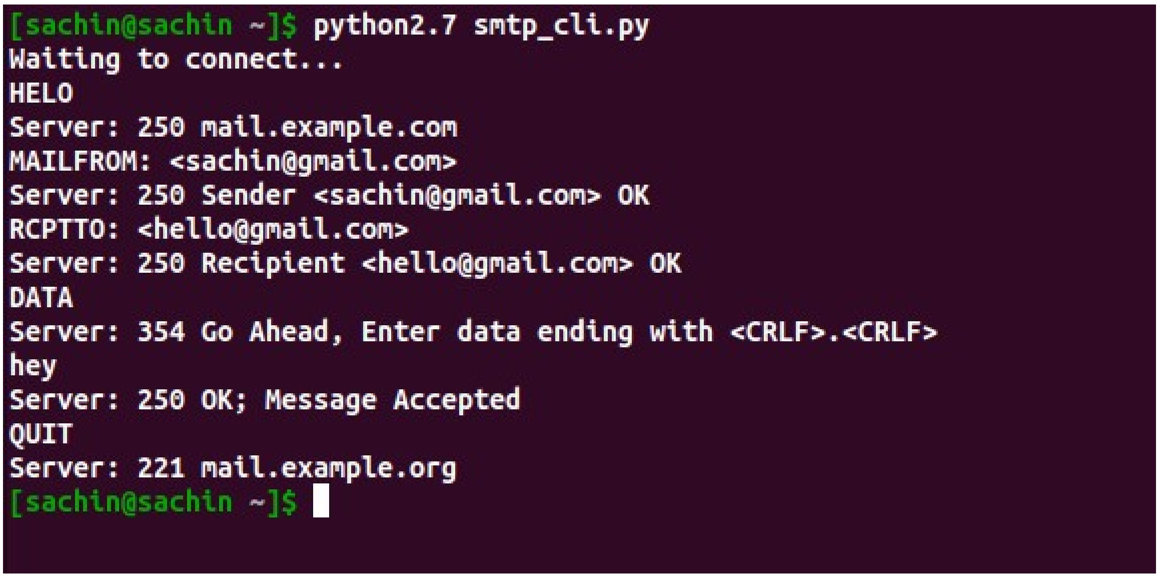
smtp_cli.py

```
import socket
import sys
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.connect(("127.0.0.1",5001))
print("Waiting to connect...")
while 1:
        comm=raw_input()
        s.send(comm)
        print("Server: "+s.recv(1024))
        if comm=='QUIT':
                s.close()
                break
```

Output



```
[sachin@sachin ~]$ python2.7 smtp_ser.py
HELO
MAILFROM: <sachin@gmail.com>
RCPTTO: <hello@gmail.com>
DATA
Message: hey
[sachin@sachin ~]$ 
```



```
[sachin@sachin ~]$ python2.7 smtp_cli.py
Waiting to connect...
HELO
Server: 250 mail.example.com
MAILFROM: <sachin@gmail.com>
Server: 250 Sender <sachin@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>
hey
Server: 250 OK; Message Accepted
QUIT
Server: 221 mail.example.org
[sachin@sachin ~]$ 
```

Result

Simulated SMTP(Simple Mail transfer Protocol) using UDP with python.