EXPERIMENT - XIII
SIMPLE MAIL TRANSFER PROTOCOL

April 6, 2020

ADITHYA D RAJAGOPAL
ROLL NO : 9
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
COLLEGE OF ENGINEERING TRIVANDRUM

AIM

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

THEORY

Simple Mail Transfer Protocol

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. End the data stream with <CRLF>.
- 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.

SOURCE CODE

Server

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<unistd.h>
#include<sys/socket.h>
#include<arpa/inet.h>
#include<fcntl.h>
#define SA struct sockaddr
#define PORT 8080
#define MAX 1024

char users[2][80]={"abc@mail.com","xyz@mail.com"};
char from[80],to[80];

int processMailFrom(char* buff)
{
 int i;
 char s[50];
 if(strncmp("MAIL FROM",buff,9)==0)
 {
  printf("MAIL FROM received...\n");
  for(i=0;i<strlen(buff);i++)
  {
   s[i]=buff[i+11];
   if(s[i]=='>')
   break;
  }
  s[i+1]='\0';
  for(i=0;i<2;i++)
  if(strncmp(users[i],s,strlen(users[i]))==0)
  {
```

```
    strcpy(from,users[i]);
    return 250;
   }
   return 501;
  }
  return 503;
}

int processRcptTo(char* buff)
{
 int i;
 char s[50];
 if(strncmp("RCPT TO",buff,7)==0)
 {
  printf("RCPT TO received...\n");
  for(i=0;i<strlen(buff);i++)
  {
   s[i]=buff[i+9];
   if(s[i]=='>')
   break;
  }
  for(i=0;i<2;i++)
  if(strncmp(users[i],s,strlen(users[i]))==0)
  {
   strcpy(to,users[i]);
   return 250;
  }
  return 501;
 }
 return 503;
}

void handleData(int sockfd)
{
 char buffer[MAX];
```

```
 char msg[80],str[MAX];
 int mail,i,j;
 strcpy(msg,to);
 strcat(msg,".txt");
 mail=open(msg,O_CREAT|O_EXCL|O_WRONLY,0666);
 strcpy(msg,"FROM : ");
 strcat(msg,from);
 strcat(msg,"\n");
 write(mail,msg,sizeof(msg));
 read(sockfd,buffer,sizeof(buffer));
 for(i=0;i<MAX;i++)
 {
  if(buffer[i]=='<')
  {
   char end[10]="<CRLF>\n";
   for(j=0;j<5;j++)
   if(buffer[i+j]!=end[j])
   break;
   if(j==5)
   break;
  }
 }
 int len=i+j+2;
 buffer[len-1]='\n';
 write(mail,buffer,len);
 printf("Mail content successfully written to file\n");
 close(mail);
}

void processMail(int sockfd)
{
 char buffer[MAX],msg[MAX];
 int status=220;
 memset(buffer,' ',MAX);
 sprintf(msg,"%d mail.com Ready\n",status);
```

```
write(sockfd,msg,sizeof(msg));
read(sockfd,buffer,MAX);
if(strncmp("HELO",buffer,4)==0)
{
 printf("HELO received...\n");
 if(strncmp("HELO mail.com",buffer,13)==0)
 status=250;
 else
 status=501;
}
else
{
 status=503;
 sprintf(msg,"%d bad sequence",status);
 write(sockfd,msg,sizeof(msg));
 close(sockfd);
 exit(0);
}
if(status==250)
{
 printf("250 OK\n");
 sprintf(msg,"%d OK",status);
 write(sockfd,msg,sizeof(msg));
}
else if(status==501)
{
 sprintf(msg,"%d wrong domain",status);
 write(sockfd,msg,sizeof(msg));
 close(sockfd);
 exit(0);
}
memset(buffer,' ',MAX);
read(sockfd,buffer,sizeof(buffer));
status=processMailFrom(buffer);
if(status==250)
```

```
{
 printf("250 OK\n");
 sprintf(msg,"%d OK",status);
 write(sockfd,msg,sizeof(msg));
}
else if(status==501)
{
 sprintf(msg,"%d wrong domain",status);
 write(sockfd,msg,sizeof(msg));
 close(sockfd);
 exit(0);
}
else if(status==503)
{
 sprintf(msg,"%d bad sequence",status);
 write(sockfd,msg,sizeof(msg));
 close(sockfd);
 exit(0);
}
memset(buffer,' ',MAX);
read(sockfd,buffer,MAX);
status=processRcptTo(buffer);
if(status==250)
{
 printf("250 OK\n");
 sprintf(msg,"%d OK",status);
 write(sockfd,msg,sizeof(msg));
}
else if(status==501)
{
 sprintf(msg,"%d wrong domain",status);
 write(sockfd,msg,sizeof(msg));
 close(sockfd);
 exit(0);
}
```

```
else if(status==503)
{
 sprintf(msg,"%d bad sequence",status);
 write(sockfd,msg,sizeof(msg));
 close(sockfd);
 exit(0);
}
memset(buffer,' ',MAX);
read(sockfd,buffer,MAX);
if(strncmp("DATA",buffer,4)==0)
{
 printf("DATA command received...\n");
 sprintf(msg,"354 Start mail input; end with <CRLF>\n");
 write(sockfd,msg,sizeof(msg));
 handleData(sockfd);
}
else
{
 sprintf(msg,"%d bad sequence",status);
 write(sockfd,msg,sizeof(msg));
 close(sockfd);
 exit(0);
}
memset(buffer,' ',MAX);
read(sockfd,buffer,MAX);
if(strncmp("QUIT",buffer,4)==0)
{
 status=221;
 printf("QUIT command received...\n");
 sprintf(msg,"%d service closing",status);
 write(sockfd,msg,sizeof(msg));
}
else
{
 sprintf(msg,"%d bad sequence",status);
```

```
   write(sockfd,msg,sizeof(msg));
   close(sockfd);
   exit(0);
  }
  close(sockfd);
}

void main()
{
  int sockfd,connfd,bindfd,len;
  struct sockaddr_in servaddr,cliaddr;
  sockfd=socket(AF_INET,SOCK_STREAM,0);
  if(sockfd<0)
  {
   printf("Socket creation failed...\n");
   exit(0);
  }
  else
  printf("Socket created successfully...\n");
  memset(&servaddr,0,sizeof(servaddr));
  servaddr.sin_family=AF_INET;
  servaddr.sin_addr.s_addr=htonl(INADDR_ANY);
  servaddr.sin_port=htons(PORT);
  bindfd=bind(sockfd,(SA*)&servaddr,sizeof(servaddr));
  if(bindfd==0)
  printf("Socket binded successfully...\n");
  else
  {
   printf("Socket binding failed...\n");
   exit(0);
  }
  if(listen(sockfd,5)<0)
  {
   printf("Socket listen failed...\n");
   exit(0);
```

```
 }
 else
 printf("Socket listening on port %d\n",PORT);
 len=sizeof(cliaddr);
 connfd=accept(sockfd,(SA*)&cliaddr,&len);
 if(connfd<0)
 {
  printf("Connection failed...\n");
  exit(0);
 }
 else
 printf("Connection successful...\n");
 processMail(connfd);
 close(sockfd);
}
```

Client

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<unistd.h>
#include<sys/socket.h>
#include<arpa/inet.h>
#include<fcntl.h>
#define MAX 1024
#define PORT 8080
#define SA struct sockaddr

void sendMail(char* from,char* to,char* body)
{
 char buffer[MAX],msg[MAX];
 int sockfd,connfd;
 struct sockaddr_in servaddr;
 memset(buffer,' ',MAX);
 char data[MAX],response[MAX],command[MAX];
 sockfd=socket(AF_INET,SOCK_STREAM,0);
 if(sockfd<0)
 {
  printf("Socket creation failed...\n");
  exit(0);
 }
 else
 printf("Socket created successfully...\n");
 servaddr.sin_family=AF_INET;
 servaddr.sin_port=htons(PORT);
 servaddr.sin_addr.s_addr=inet_addr("127.0.0.1");
 connfd=connect(sockfd,(SA*)&servaddr,sizeof(servaddr));
 if(connfd<0)
 {
  printf("Conection to the server failed...\n");
```

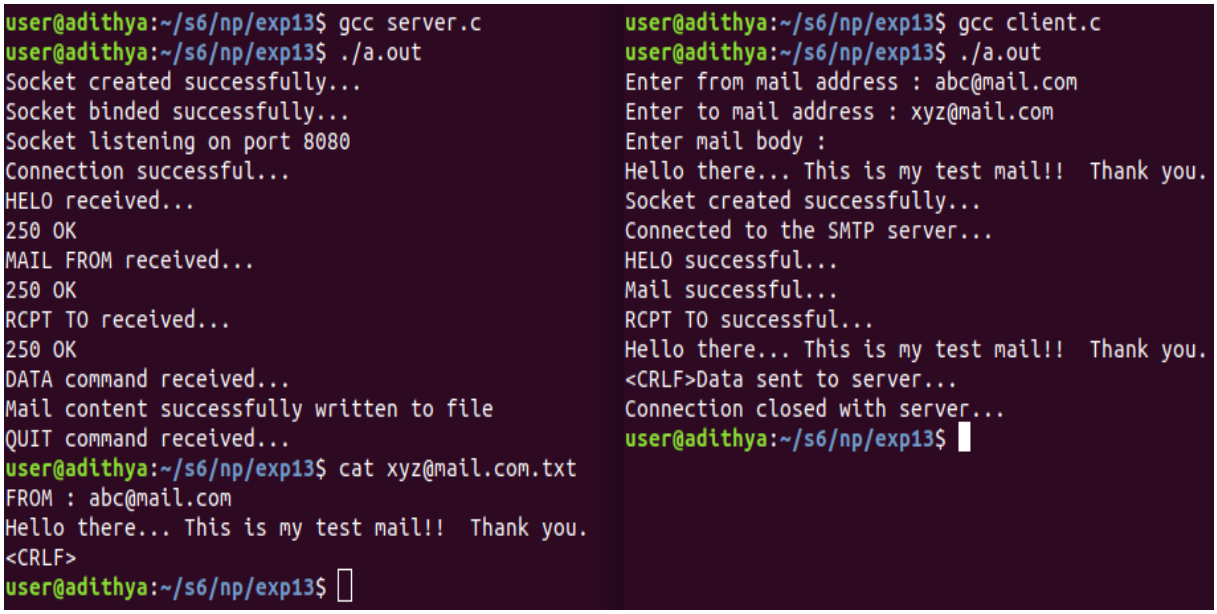
```
  exit(0);
 }
 read(sockfd,buffer,MAX);
 if(strncmp("220",buffer,3)==0)
 printf("Connected to the SMTP server...\n");
 else
 {
  printf("Unable to connect to the server...\n");
  exit(0);
 }
 sprintf(msg,"HELO mail.com");
 write(sockfd,msg,sizeof(msg));
memset(buffer,' ',MAX);
 read(sockfd,buffer,MAX);
 if(strncmp("250",buffer,3)==0)
 printf("HELO successful...\n");
 else if(strncmp("503",buffer,3)==0)
 {
  printf("Bad sequence of commands...\n");
  exit(0);
 }
 else if(strncmp("501",buffer,3)==0)
 {
  printf("Invalid domain...\n");
  exit(0);
 }
 strcpy(command,"MAIL FROM:<");
 strcat(command,from);
 strcat(command,">");
 write(sockfd,command,sizeof(command));
memset(buffer,' ',MAX);
 read(sockfd,buffer,MAX);
 if(strncmp("250",buffer,3)==0)
 printf("Mail successful...\n");
 else if(strncmp("503",buffer,3)==0)
```

```
{
 printf("Bad sequence of commands...\n");
 exit(0);
}
else if(strncmp("501",buffer,3)==0)
{
 printf("Check sender...\n");
 exit(0);
}
strcpy(command,"RCPT TO:<");
strcat(command,to);
strcat(command,">");
write(sockfd,command,sizeof(command));
memset(buffer,' ',MAX);
read(sockfd,buffer,MAX);
if(strncmp("250",buffer,3)==0)
printf("RCPT TO successful...\n");
else if(strncmp("503",buffer,3)==0)
{
 printf("Bad sequence of commands...\n");
 exit(0);
}
else if(strncmp("501",buffer,3)==0)
{
 printf("Check reciever...\n");
 exit(0);
}
strcpy(command,"DATA\n");
write(sockfd,command,sizeof(command));
memset(buffer,' ',MAX);
read(sockfd,buffer,MAX);
if(strncmp("354",buffer,3)==0)
{
 strcat(body,"<CRLF>");
 write(sockfd,body,strlen(body));
```

```
    printf("%s",body);
    printf("Data sent to server...\n");
  }
  strcpy(command,"QUIT");
  write(sockfd,command,sizeof(command));
  memset(buffer,' ',MAX);
  read(sockfd,buffer,MAX);
  if(strncmp("221",buffer,3)==0)
  printf("Connection closed with server...\n");
  close(sockfd);
}

void main()
{
  char from[80],to[80],data[MAX];
  printf("Enter from mail address : ");
  fgets(from,80,stdin);
  printf("Enter to mail address : ");
  fgets(to,80,stdin);
  printf("Enter mail body : \n");
  fgets(data,MAX,stdin);
  sendMail(from,to,data);
}
```

OUTPUT


user@adithya:~/s6/np/exp13$ gcc server.c
user@adithya:~/s6/np/exp13$ ./a.out
Socket created successfully...
Socket binded successfully...
Socket listening on port 8080
Connection successful...
HELO received...
250 OK
MAIL FROM received...
250 OK
RCPT TO received...
250 OK
DATA command received...
Mail content successfully written to file
QUIT command received...
user@adithya:~/s6/np/exp13$ cat xyz@mail.com.txt
FROM : abc@mail.com
Hello there... This is my test mail!! Thank you.
<CRLF>
user@adithya:~/s6/np/exp13$
user@adithya:~/s6/np/exp13$ gcc client.c
user@adithya:~/s6/np/exp13$ ./a.out
Enter from mail address : abc@mail.com
Enter to mail address : xyz@mail.com
Enter mail body :
Hello there... This is my test mail!! Thank you.
Socket created successfully...
Connected to the SMTP server...
HELO successful...
Mail successful...
RCPT TO successful...
Hello there... This is my test mail!! Thank you.
<CRLF>Data sent to server...
Connection closed with server...
user@adithya:~/s6/np/exp13$

RESULT

Simulated simple mail transfer protocol using UDP in C.