

컴퓨터네트워크

- 05. Mail -

※ 과제 목표 및 해결 방법

- 과제 목표
 - Python Socket으로 SMTP, POP3, IMAP 구현
 - ※ id, password의 경우 sys argv로 입력받도록 구현
- 해결 방법

1. SMTP

```
from socket import *  
import ssl  
import base64  
import sys
```

[그림1. use_module]

→ 구현에 필요한 socket, ssl, base64, sys 모듈을 import한다.

```
# Variable  
hostname = "smtp.naver.com"  
portNumber = 465  
  
username = sys.argv[1]  
password = sys.argv[2]
```

[그림2. variable]

→ 변수로 사용한 smtp_hostname, portNumber와 실행할 때 인자로 준 아이디와 비밀번호를 저장한다.

```
# Connect  
client_socket = socket(AF_INET, SOCK_STREAM)  
ssl_client_sock = ssl.wrap_socket(client_socket)  
ssl_client_sock.connect((hostname, portNumber))
```

[그림3. connect]

→ socket을 열고 ssl Client를 사용해서 ssl_client_sock으로 naver mail 서버와 연결한다.

```
# First recv  
recv = ssl_client_sock.recv(1024)  
print('S :', recv.decode('utf-8'), '\r\n')
```

[그림4. first recv]

→ 서버와 연결 후 서버로부터 데이터를 받아서 출력한다.

```
# AUTH LOGIN
login_command = 'AUTH LOGIN\r\n'
ssl_client_sock.send(login_command.encode())
recv_1 = ssl_client_sock.recv(1024)
print('C :', login_command)
print('S :', recv_1.decode('utf-8'), '\r\n')
```

[그림5. auth login]

➔ 로그인을 위한 명령어를 서버에게 전송한다. 전송 후 서버에게 받은 데이터를 출력한다.

```
# base64encoded ID
username_encode = base64.b64encode(username.encode())
a = username_encode.decode() + '\r\n'
ssl_client_sock.send(a.encode())
recv_1 = ssl_client_sock.recv(1024)
print('C: ', username_encode, '\r\n')
print('S :', recv_1.decode('utf-8'), '\r\n')

# base64encoded PASSWORD
userpass_encode = base64.b64encode(password.encode())
b = userpass_encode.decode() + '\r\n'
ssl_client_sock.send(b.encode())
recv_1 = ssl_client_sock.recv(1024)
print('C : pass', userpass_encode, '\r\n')
print('S :', recv_1.decode('utf-8'), '\r\n')
```

[그림6. id, password]

➔ ID, PASSWORD를 입력하여 로그인을 한다. 여기서 ID, PASSWORD는 base64를 사용해서 암호화를 한다. 암호화한 ID, PASSWORD에 'WrWn'을 기준으로 구분하기 때문에 붙여서 서버에게 전송해준다. 전송한 데이터에 대해서 서버가 보낸 데이터를 받아서 출력한다.

```
# MAIL FROM : <송신메일주소>
mail_from = 'MAIL FROM: <{0}>\r\n'.format("ssey0921@naver.com")
ssl_client_sock.send(mail_from.encode())
recv_1 = ssl_client_sock.recv(1024)
print('C :', mail_from)
print('S :', recv_1.decode('utf-8'), '\r\n')

# RCPT TO: <수신메일주소>
mail_to = 'RCPT TO: <{0}>\r\n'.format("ssey0921@gmail.com")
ssl_client_sock.send(mail_to.encode())
recv_1 = ssl_client_sock.recv(1024)
print('C :', mail_to)
print('S :', recv_1.decode('utf-8'), '\r\n')
```

[그림7. mail_addr]

➔ 송신 메일 주소와 수신 메일 주소를 입력하여 처리하는 과정이다. 정해진 형식에 맞춰서 네이버 이메일로부터 구글 이메일로 보내는 것으로 입력하였다.

```
# DATA
data = 'DATA\r\n'
ssl_client_sock.send(data.encode())
recv_1 = ssl_client_sock.recv(1024)
print('C : ', data)
print('S : ', recv_1.decode('utf-8'), '\r\n')
```

[그림8. data]

→ data를 보내기 시작한다는 의미로 DATA의 명령어를 입력하는 것 같다는 생각했다.

```
# SUBJECT, FROM, TO...
subject = "SUBJECT: {0}\r\n".format("AAAAAAAAAAAA")
_from = "FROM: {0}\r\n".format("ssey0921@naver.com")
_to = "TO: {0}\r\n".format("ssey0921@gmail.com")
_content = "SMTP TEST!\r\n"
_send = ".\r\n"
ssl_client_sock.send(subject.encode())
ssl_client_sock.send(_from.encode())
ssl_client_sock.send(_to.encode())
ssl_client_sock.send(_content.encode())
ssl_client_sock.send(_send.encode())
recv_1 = ssl_client_sock.recv(1024)
print('C : ', subject)
print('C : ', _from)
print('C : ', _to)
print('C : ', _content)
print('C : ', _send)
print('S : ', recv_1.decode('utf-8'), '\r\n')
```

[그림9. subject, from, to, content]

→ 메일의 제목과 FROM, TO, CONTENT, DOT을 처리할 수 있도록 변수로 저장해준다. 저장한 변수들은 인코딩하여 서버에게 전송한다. 마지막 "."이 들어가야 메일 작성을 마쳤다는 것을 알 수 있다.

```
# QUIT
quitcommand = 'QUIT\r\n'
ssl_client_sock.send(quitcommand.encode())
print('C : ', quitcommand)
ssl_client_sock.close()
```

[그림10. quit]

→ QUIT를 입력하여 메일을 전송하고 마치는 것으로 알 수 있다. 전송을 마치면 Socket을 닫는다.

※ 실행 결과 - (SMTP)

```
leebyeongman@leebyeongman-VirtualBox:~/CN_201402391/05$ python3 SMTP.py ssey0921
S : 220 smtp.naver.com ESMTP uPuI8IRbSdCVoHHIAe2HtQ - nsmtp
C : EHL0 naver.com
S : 220 smtp.naver.com ESMTP uPuI8IRbSdCVoHHIAe2HtQ - nsmtp
C : AUTH LOGIN
S : 220 smtp.naver.com ESMTP uPuI8IRbSdCVoHHIAe2HtQ - nsmtp
C: b'c3NleTA5MjE='
S : 220 smtp.naver.com ESMTP uPuI8IRbSdCVoHHIAe2HtQ - nsmtp
C : pass
S : 220 smtp.naver.com ESMTP uPuI8IRbSdCVoHHIAe2HtQ - nsmtp
C : MAIL FROM: <ssey0921@naver.com>
S : 220 smtp.naver.com ESMTP uPuI8IRbSdCVoHHIAe2HtQ - nsmtp
C : RCPT TO: <ssey0921@gmail.com>
S : 220 smtp.naver.com ESMTP uPuI8IRbSdCVoHHIAe2HtQ - nsmtp
C : DATA
S : 220 smtp.naver.com ESMTP uPuI8IRbSdCVoHHIAe2HtQ - nsmtp
C : SUBJECT: AAAAAAAAAA
C : FROM: ssey0921@naver.com
C : TO: ssey0921@gmail.com
C : SMTP TEST!
C : .
S : 220 smtp.naver.com ESMTP uPuI8IRbSdCVoHHIAe2HtQ - nsmtp
C : QUIT
```



2. POP3

```
from socket import *  
import ssl  
import base64  
import sys
```

[그림1. use_module]

→ 구현에 필요한 socket, ssl, base64, sys 모듈을 import한다.

```
# Variable  
hostname = "pop.naver.com"  
portNumber = 995  
  
username = sys.argv[1]  
password = sys.argv[2]
```

[그림2. variable]

→ 변수로 사용한 pop3_hostname, portNumber와 실행할 때 인자로 준 아이디와 비밀번호를 저장한다.

```
# Connect  
client_socket = socket(AF_INET, SOCK_STREAM)  
ssl_client_sock = ssl.wrap_socket(client_socket)  
ssl_client_sock.connect((hostname, portNumber))
```

[그림3. connect]

→ socket을 열고 ssl Client를 사용해서 ssl_client_sock으로 naver mail 서버와 연결한다.

```
# First recv  
recv = ssl_client_sock.recv(1024)  
print('S : ', recv.decode('utf-8'), '\r\n')
```

[그림4. first recv]

→ 서버와 연결 후 서버로부터 데이터를 받아서 출력한다.

```
# Input ID  
username_send = 'user ' + username + '\r\n'  
ssl_client_sock.send(username_send.encode())  
recv_1 = ssl_client_sock.recv(1024)  
print('C : ', 'user ' + username, '\r\n')  
print('S : ', recv_1.decode('utf-8'), '\r\n')
```

[그림5. input id]

→ 유저의 이메일 주소를 입력받은 것을 서버에게 전송하고 데이터를 서버로부터 받은 것을 출력한다.

```
# Input PASSWORD
userpass_send = 'pass ' + password + '\r\n'
ssl_client_sock.send(userpass_send.encode())
recv_1 = ssl_client_sock.recv(1024)
print('C : ', 'pass ' + password, '\r\n')
print('S : ', recv_1.decode('utf-8'), '\r\n')
```

[그림6. input password]

➔ 유저의 비밀번호를 입력받은 것을 서버에게 전송하고 데이터를 서버로부터 받은 것을 출력한다.

```
while True:
    _input = input('C : ')
    _input = _input + '\r\n'
    ssl_client_sock.send(_input.encode())
    recv_1 = ssl_client_sock.recv(4096)
    print('S : ', recv_1.decode('utf-8'), '\r\n')
```

[그림7. input command]

➔ 반복문을 실행하면서 명령어를 입력받는다.

※ 실행 결과 - (POP3)

```
leebyeongman@leebyeongman-VirtualBox:~/CN_201402391/05$ python3 POP3.py ssey0921@naver.com qudaks0921
S : +OK Naver Popper starting

C : user ssey0921@naver.com

S : +OK Password required for ssey0921@naver.com.

C : pass [REDACTED]

S : +OK ssey0921@naver.com has 55 visible messages in 18848598 octets.
```

```
C : list
S : +OK 55 visible messages (18848598 octets)
1 1802
2 27617
3 8440
4 8442
5 6505
6 9660
7 46069
8 7716
9 32495
10 72239
11 13435
12 305335
13 52301
14 9905
15 110729
16 10000
```

```
17 24070
18 24070
19 24070
20 24070
21 24070
22 24070
23 24070
24 24070
25 24070
26 24070
27 24070
28 24070
29 24070
30 24070
31 24070
32 24070
33 24070
34 24070
35 24070
36 24070
37 24070
38 24070
39 24070
40 24070
41 24070
42 24070
43 24070
44 24070
45 24070
46 24070
47 24070
48 24070
49 15242
50 6355
51 22075
52 5558076
53 5558111
54 139715
55 3012
```

```
C : retr 1
S : +OK 1802 octets
DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/simple; d=naver.com; s=s20171208;
t=1538724584; bh=s2mh2YJopieo1DhUsZ4Y4lBZspxoXr7NC5iwTTAjb4=;
h=Message-ID:Date:From:To:Subject;
b=FNiPJ8rjHcevhNT4xfZoP09uxlKPuz8no1HRWiHyfhTrjq0iu6lY1bKjRg7/bbKWw
4tFqdH2mX9UGsaGEukc+fLFK0lKve4SefY/6FUUow8ifWaeVzQrH7YEJwEBRWJfIFb
8e0HNYHPeMwGUCsbJoqJrWcABpJAjVVH+XEkmBw5BbQgweqdhByfD+khv07T8MotJ
/1JfT3fCPkIuNr8Hjovdz+p0jzce+4Kn6qqGSscIRRCH5kbv7iLW0iiAZ0jNZhh66u
T9BF5v3aznMQYfaIQAKzEisMP7QKXu9Q7zh8lfQ1YUAW3I0mPWctFCrtigWeTEfDhq
T+HJg8a58JguA==
X-Session-ID: o707zHifQUSyPviQCu7aLw
MIME-Version: 1.0
Message-ID: <e3a4lead885ef151285fb2cd4cb29dd@cweb30.nm.nhnsystem.com>
Date: Fri, 05 Oct 2018 16:29:43 +0900
From: =?utf-8?B?7Zwc7KeE7JiB?=<hjiny2002@naver.com>
Importance: normal
To: =?utf-8?B?670R66eM66as?=<ssey0921@naver.com>
Subject: =?utf-8?B?SVU=?=
X-Originating-IP: 168.188.130.239
Content-Type: multipart/alternative;
boundary="-----Boundary-WM=_7f1c2f3f9700.1538724584071"

-----Boundary-WM=_7f1c2f3f9700.1538724584071
Content-Type: text/plain;
charset="utf-8"
Content-Transfer-Encoding: base64

SVU=

-----Boundary-WM=_7f1c2f3f9700.1538724584071
Content-Type: text/html;
charset="utf-8"
Content-Transfer-Encoding: base64

PGh0bWw+PGh1YW0+PHN0eWxlPnB7bWYyZ2luLXRvcDowcHq7bWYyZ2luLWJvdHRvbTowcHq7fTwv
c3R5bGU+PC9oZWFKPjxib2R5Pk1VPC9ib2R5PjwvaHRtdD48dGF1bGUgc3R5bGU9J2Rpc3BsYXk6
bm9uZSc+PHRyPjx0ZD48aWlnIHNYZz0iaHR0cHM6Ly9tYWlsLm5hdmVybWVubS9yZWFKUmVjZWlw
dC9ub3Rpb2NkP2ltZz1Fd1laRDZsRzE0MnFweG1zS29nbU14dHdwNkp2YUEyOXA2TXdGeHVaYUFR
NE1vaW9wQTNvTW9vbHB6M2dNWCUyQjBNb0ttNzRsUjc0bGNXTkZsYlgzMfMdbG9XcmRRYVhGcXBC
dm1heFV3dHpsQyYyQjRrWjc0RlRXdCUzRCUzRC5naWYiIGJvcmlkZj01MCIvPjwvdGQ+PC90cj48
L3RhYmxlPg==

-----Boundary-WM=_7f1c2f3f9700.1538724584071--
```


3. IMAP

```
from socket import *  
import ssl  
import sys
```

[그림1. use_module]

→ 구현에 필요한 socket, ssl, base64, sys 모듈을 import한다.

```
# Commuicate  
def transport():  
    _input = input('C : ')  
    _input = _input + '\r\n'  
    ssl_client_sock.send(_input.encode())  
    recv_1 = ssl_client_sock.recv(4096)  
    print('S : ', recv_1.decode('utf-8'), '\r\n')
```

[그림2. transport]

→ 변수로 사용한 imap_hostname, portNumber와 실행할 때 인자로 준 아이디와 비밀번호를 저장한다.

```
# Variable  
hostname = 'imap.naver.com'  
portNumber = 993  
  
username = sys.argv[1]  
password = sys.argv[2]
```

[그림3. variable]

→ 변수로 사용한 imap_hostname, portNumber와 실행할 때 인자로 준 아이디와 비밀번호를 저장한다.

```
# Connect  
client_socket = socket(AF_INET, SOCK_STREAM)  
ssl_client_sock = ssl.wrap_socket(client_socket)  
ssl_client_sock.connect((hostname, portNumber))
```

[그림4. connect]

→ socket을 열고 ssl Client를 사용해서 ssl_client_sock으로 naver mail 서버와 연결한다.

```
# First recv  
recv = ssl_client_sock.recv(1024)  
print('S : ', recv.decode('utf-8'), '\r\n')
```

[그림5. first recv]

→ 서버와 연결 후 서버로부터 데이터를 받아서 출력한다

```
# LOGIN
login = 'a LOGIN ' + username + ' ' + password
login = login + '\r\n'
ssl_client_sock.send(login.encode())
recv_1 = ssl_client_sock.recv(1024)
print('S : ', recv_1.decode('utf-8'), '\r\n')
```

[그림6. login]

→ 로그인하기 위해서 인자로 주었던 값을 login변수에 저장해서 서버에게 전송해서 데이터를 받아서 출력해준다.

```
# INPUT
while True:
    transport()
```

[그림7. input]

→ 위에 선언한 함수를 실행해서 입력한 명령어에 대해서 서버에서 받은 데이터를 출력해준다.

※ 실행 결과 - (IMAP)

- login -

```
leebyeongman@leebyeongman-VirtualBox:~/CN_201402391/05$ python3 IMAP.py phantom0921@naver.com dlqudaks12
S : * OK [CAPABILITY IMAP4rev1 LITERAL+ SASL-IR ID AUTH=PLAIN] Naver Imap Server ready.

C : a LOGIN phantom0921@naver.com [REDACTED]

S : a OK [CAPABILITY COMPRESS=DEFLATE IMAP4rev1 LITERAL+ SASL-IR ID UNSELECT CHILDREN UIDPLUS LIST-EXTEND
```

- a list "" "*" -

```
C : a list "" "*"
S : * LIST (\HasNoChildren) "/" "&sLSsjMT0u1THfNV0-"
* LIST (\HasNoChildren) "/" "&wqTT0LpUx3zVaA-"
* LIST (\HasNoChildren) "/" "Deleted Messages"
* LIST (\HasNoChildren) "/" "Drafts"
* LIST (\HasNoChildren) "/" "INBOX"
* LIST (\HasNoChildren) "/" "Sent Messages"
a OK List completed.
```

- a select inbox -

```
C : a select inbox
S : * FLAGS (\Answered \Flagged \Deleted \Seen \Draft)
* OK [PERMANENTFLAGS (\Answered \Flagged \Deleted \Seen \Draft \*)] Flags permitted.
* 2 EXISTS
* 0 RECENT
* OK [UNSEEN 1] First unseen.
* OK [UIDVALIDITY 0] UIDs valid
* OK [UIDNEXT 678] Predicted next UID
* OK [NOMODSEQ] No permanent modsequences
a OK [READ-WRITE] Select completed.
```

- 1 fetch 1:* flags -

```
C : 1 fetch 1:* flags
S : * 1 FETCH (FLAGS ())
* 2 FETCH (FLAGS ())
1 OK Fetch completed.
```

- a fetch 1 body[] -

```
C : a fetch 1 body[]
S : * 1 FETCH (BODY[] {4113})
Received-SPF: pass (mx.naver.com: domain of ssey0921@gmail.com designates 209.85.166.177 as permitted sender)
client-ip=209.85.166.177; x-iptype=white;
Authentication-Results: mx.naver.com;
spf=pass (mx.naver.com: domain of ssey0921@gmail.com designates 209.85.166.177 as permitted sender) smtp.mailfrom=ssey0921@gmail.com;
dkim=pass header.i=@gmail.com;
dmarc=pass (p=NONE) header.from=gmail.com
X-Naver-ESV: +XFYpB3G1H+YbXFCbHmRKAewKAULjY
X-Session-IP: 209.85.166.177
Received: from mail-it1-f177.google.com (mail-it1-f177.google.com [209.85.166.177])
by crcvmail12.nm.naver.com with SMTP id +2w+A4eVRb+3JFDaieYE1w
for <phanton0921@naver.com>;
Thu, 11 Oct 2018 05:10:29 -0800
Received: by mail-it1-f177.google.com with SMTP id 176-v6so11437852ita.3
for <phanton0921@naver.com>; Wed, 10 Oct 2018 22:10:28 -0700 (PDT)
DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/relaxed;
d=gmail.com; s=20161025;
h=mime-version:references:in-reply-to:from:d
Content-Type: text/plain; charset="UTF-8"
Content-Transfer-Encoding: quoted-printable
```

```
----- Forwarded message -----
From: =EC=9D=B4=EB=B3=91=EB=A7=8C <ssey0921@gmail.com>
Date: 2018=EB=85=84 10=EC=9B=94 10=EC=9D=BC (=EC=88=98) =EC=98=A4=ED=98=84 1=
12:45
Subject: TEST1
To: <phanton0921@naver.com>
```

IMAP TEST!!!

```
--0000000000003c0d270577ecfd53
Content-Type: text/html; charset="UTF-8"
Content-Transfer-Encoding: quoted-printable
```

```
<div dir=3D"ltr"><br><br><div class=3D"gmail_quote"><div dir=3D"ltr">-----=
---- Forwarded message -----<br>From: <strong class=3D"gmail_sendername=
" dir=3D"auto">=EC=9D=B4=EB=B3=91=EB=A7=8C</strong> <span dir=3D"ltr">&lt;=
a href=3D"mailto:ssey0921@gmail.com">ssey0921@gmail.com</a>&gt;</span><br>D=
ate: 2018=EB=85=84 10=EC=9B=94 10=EC=9D=BC (=EC=88=98) =EC=98=A4=ED=98=84 1=
2:45<br>Subject: TEST1<br>To: &lt; a href=3D"mailto:phanton0921@naver.com"=
>phanton0921@naver.com</a>&gt;<br></div><br><div dir=3D"ltr">IMAP TEST!=
!!</div>
</div><
C :
S : /div>
```

```
--0000000000003c0d270577ecfd53--
)
```

```
C :
S : a OK Fetch completed.
```

- a store +flags \Deleted -

```
S : * 1 FETCH (FLAGS (\Deleted))
* 1 EXPUNGE
* 1 EXISTS
a OK Store completed.
```

- a expunge -

```
C : a expunge
S : a OK Expunge completed.
```

4. 심화과제 – SMTP 메일 보내기 기능

```
from socket import *
import os
import ssl
import base64
import sys
```

[그림1. use_module]

→ 구현에 필요한 socket, os, ssl, base64, sys 모듈을 import한다.

```
# variable
ip_addr = '127.0.0.1'
port_num = 2345

# connect
server_sock = socket(AF_INET, SOCK_STREAM)
server_sock.bind((ip_addr, port_num))
print("Server socket open...")
print("Listening...")
server_sock.listen(1)
clnt_sock, addr = server_sock.accept()
```

[그림2. HTTP socket connect]

→ ip_addr, port_num을 설정하고, HTTP연결을 위해서 socket을 연다. 클라이언트와의 접속을 기다린다. clnt_sock은 server_sock과 accpet한다.

```
# recive
data = clnt_sock.recv(5000)
de_data = data.decode()
parsing_data = de_data.split(' ')

is_file = 'HTTP/1.1 200 OK\r\n\r\n'.encode()
none_file = 'HTTP/1.1 404 Not Found\r\n\r\n'.encode()

file_name = parsing_data[1].split('/')[1]

if parsing_data[0] == "GET":
    if os.path.isfile(file_name):
        f = open(file_name, 'rb')
        line = f.read()
        is_file += line
        f.close()
        clnt_sock.send(is_file)
    else:
        clnt_sock.send(none_file)

elif parsing_data[0] == "POST":
    print("Next Time")

clnt_sock.close()
```

[그림3. file read]

→ url에 주소와 포트 파일 이름을 적은 것을 서버가 받아서 처리를 한다. 입력한 파일이 존재하면 웹 브라우저에

보여준다. 그 후 해당 socket을 닫아준다.

```
# socket re_open
server_sock.listen(1)
clnt_sock, addr = server_sock.accept()
```

[그림4. socket re open]

→ 소켓을 다시 열어서 웹 브라우저에서 보내는 데이터를 받을 수 있도록 한다.

```
# variable
data = ''
recv_data = ''

# receive
data = clnt_sock.recv(5000)
de_data = data.decode()
parsing_data = de_data.split(' ')
```

[그림5. variable, receive]

→ data, recv_data 변수를 선언하고 웹 브라우저가 보낸 쿼리 데이터를 읽어서 파싱한다.

```
# data setting
url = de_data.split(' ')
query = url[1].split('?')
parse_data = query[1].split('&')

_id = parse_data[0].split('=')[1]
_pass = parse_data[1].split('=')[1]

_from = parse_data[2].split('=')[1]
_from = _from.replace('%40', '@')

_to = parse_data[3].split('=')[1]
_to = _to.replace('%40', '@')

_subject = parse_data[4].split('=')[1]
_content = parse_data[5].split('=')[1]
```

[그림6. data setting]

→ url을 파싱하고 쿼리로 받은 문자열을 각 변수로 파싱해서 할당한다. "@"는 %40으로 변환되었기 때문에 replace함수로 다시 "@"로 변환하여 저장했다.

※ 실행 결과 - (심화과제)

CN_mail_service x +

127.0.0.1:2345/mail_service.html

MAIL SERVICE

ID :

PASS :

FROM :

TO :

SUBJECT :

CONTENT :

Submit

[보내기 전]

CN_mail_service x +

127.0.0.1:2345/mail_service.htm

MAIL SERVICE

ID :

PASS :

FROM :

TO :

SUBJECT :

CONTENT :

Submit

[보내기]

```
leebyeongman@leebyeongman-VirtualBox:~/CN_201402391/05$ python3 smtp_mail_service.py
Server socket open...
Listening...
S : b'220 smtp.naver.com ESMTP FTBUIMCIQb6HhOG0SwPD2A - nsmtp\r\n'
C : EHLO naver.com

S : b'250-smtp.naver.com Pleased to meet you\r\n250-SIZE 20971520\r\n250-8BITMIME\r\n250-
C : AUTH LOGIN

S : b'334 VXNlcm5hbWU6\r\n'
ssey0921
C : b'c3NleTA5MjE='

S : b'334 UGRZC3dVcmQ0\r\n'
C : pass b'cXVkyWtzMDkyMQ=='

S : b'235 2.7.0 Accepted FTBUIMCIQb6HhOG0SwPD2A - nsmtp\r\n'
C : MAIL FROM: <ssey0921@naver.com>

S : b'250 2.1.0 OK FTBUIMCIQb6HhOG0SwPD2A - nsmtp\r\n'
C : RCPT TO: <ssey0921@gmail.com>

S : b'250 2.1.5 OK FTBUIMCIQb6HhOG0SwPD2A - nsmtp\r\n'
C : DATA

S : b'354 Go ahead FTBUIMCIQb6HhOG0SwPD2A - nsmtp\r\n'
C : SUBJECT: TEST

C : FROM: ssey0921@naver.com
C : TO: ssey0921@gmail.com
C : Network+good
C : .

S : b'250 2.0.0 OK FTBUIMCIQb6HhOG0SwPD2A - nsmtp\r\n'
C : QUIT

Send Message back to client
```

[보내기 후]

ssey0921@naver.com
나에게 ▾

영어 ▾ > 한국어 ▾ 메일 번역

Network+good

[보낸 결과]