

CN ASSIGNMENT 2

SOCKET PROGRAMMING

Topic: Encryption and Decryption

Team Details

Name: Naren Chandrashekhar	SRN: PES2UG20CS216
Name: Navtej Reddy	SRN: PES2UG20CS218
Name: Nitin Jayachandran	SRN: PES2UG20CS231
Section: D	

Abstract:

Our project is encryption/decryption using socket programming in Python. Socket programming is about the connection between client and server or many clients for data communication, networking and data transfer. The network formed between the nodes is known as sockets.

In our project, we have used Caesar Cipher to encrypt and decrypt our messages. We have a client and server in our project and the server shows which client is connected for encryption/decryption. The server also shows the messages sent and received from the client. We have set the key value of encryption as 5, which means if we want to encrypt the letter 'a' the output will be 'f'. The same way, a message can be decrypted, where 'f' gives the output 'a'.

Code:

Server.py:

```
client.py  server.py  X
D: > PES > Semester_4 > Computer Networks > Project > Socket_Programming > server.py
1  import socket
2
3  import math
4
5  def encrypt_func(txt, s):
6      result = ""
7
8      #caeser encryption
9      # transverse the plain txt
10     for i in range(len(txt)):
11         char = txt[i]
12         # encrypt_func uppercase characters in plain txt
13
14         if (char.isupper()):
15             result += chr((ord(char) + s - 64) % 26 + 65)
16         # encrypt_func lowercase characters in plain txt
17         else:
18             result += chr((ord(char) + s - 96) % 26 + 97)
19     return result
20
21 #caeser decryption
22 def decrypt_func(msg):
23     key = 5
24     LETTERS = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
25     LETTERS = LETTERS.lower()
26     message = str(msg)
27     translated = ''
28     for symbol in message:
29         if symbol in LETTERS:
30             num = LETTERS.find(symbol)
```

```

31         num = num - key
32         if num < 0:
33             num = num + len(LETTERS)
34         translated = translated + LETTERS[num]
35     else:
36         translated = translated + symbol
37     return translated
38
39 # take the server name and port name
40 host = 'localhost'
41 port = 5000
42
43 # create a socket at server side
44 # using TCP / IP protocol
45 s = socket.socket(socket.AF_INET,
46                  socket.SOCK_STREAM)
47
48 # bind the socket with server
49 # and port number
50 s.bind(('', port))
51
52 # allow maximum 1 connection to
53 # the socket
54 s.listen(1)
55 c, addr = s.accept()
56
57 # display client address
58 print("CONNECTION FROM:", str(addr))
59 while True:
60     # wait till a client accept
61     # connection
62
63     # send message to the client after
64     # encoding into binary string
65     c.send(b"enter e to encrypt and d to decrypt and f to disconnect")
66     while True:
67         msg1 = c.recv(1024)
68         print(msg1)
69         msg_str = msg1.decode()
70         if(msg_str == "f"):
71             print("disconnecting ",str(addr))
72             c.send(b"ack")
73             c.close()
74             exit(0)
75
76         if(msg_str == "e"):
77             c.send(b"enter message to encrypt")
78
79             msg2 = c.recv(1024)
80             msg2_str = msg2.decode()
81             print(msg2_str)
82             print(encrypt_func(msg2_str,4)) #s=4,shift =s+1 =5
83             answer =encrypt_func(msg2_str,4)
84             c.send(answer.encode())
85             break
86
87         if(msg_str == "d"):
88             c.send(b"enter message to decrypt")
89             msg2 = c.recv(1024)
90             msg2_str = msg2.decode()

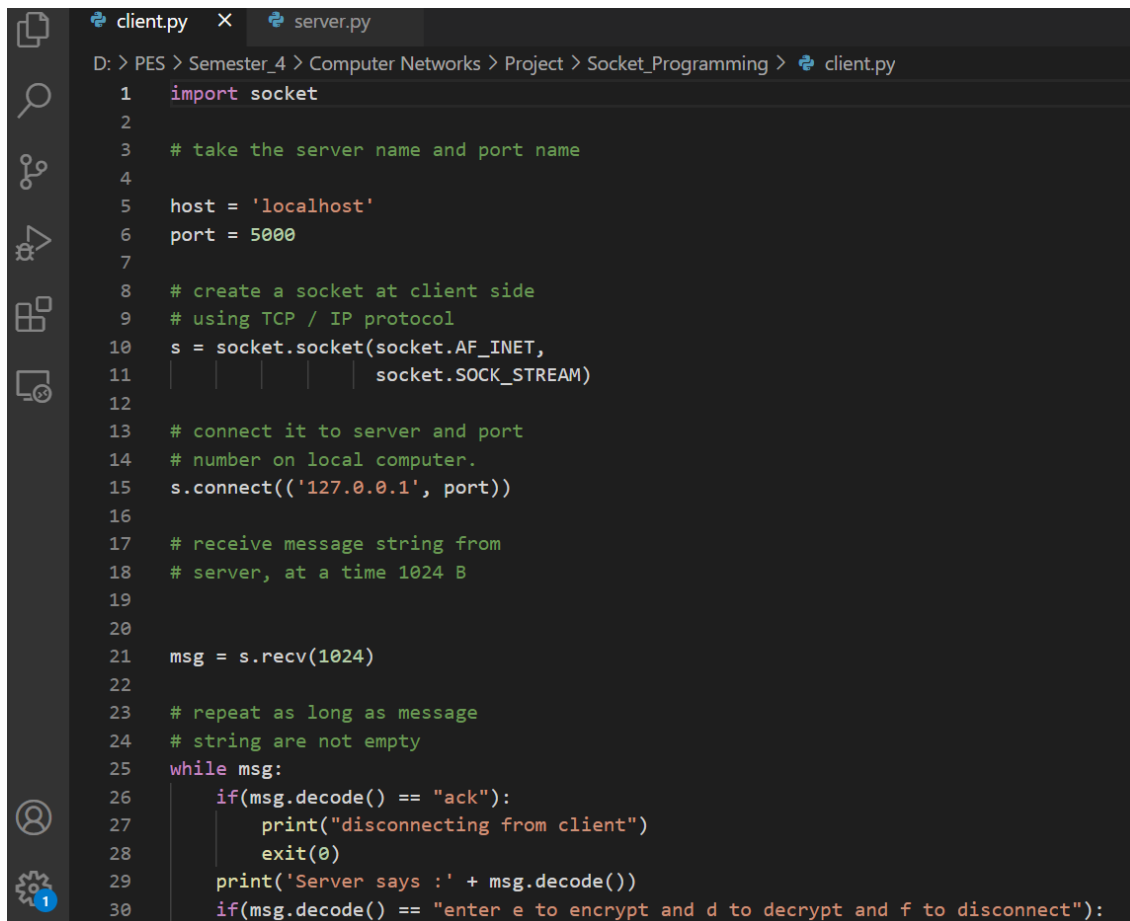
```

```

91         print(msg2_str)
92         print(decrypt_func(msg2_str))
93         answer = decrypt_func(msg2_str)
94
95         c.send(answer.encode())
96         break
97
98     #c.send(b"connection end")
99     #c.close() #added this line and below
100    #break
101    # if int(msg) == 3+4:
102    #     c.send("Correct answer".encode())
103    #     c.close()
104    #     break
105
106    # else:
107    #     c.send("Wrong answer try again.".encode())
108
109
110    # disconnect the server
111    c.close()
112    exit(0)
113

```

Client.py:



```

client.py X server.py
D: > PES > Semester_4 > Computer Networks > Project > Socket_Programming > client.py
1  import socket
2
3  # take the server name and port name
4
5  host = 'localhost'
6  port = 5000
7
8  # create a socket at client side
9  # using TCP / IP protocol
10 s = socket.socket(socket.AF_INET,
11                   socket.SOCK_STREAM)
12
13 # connect it to server and port
14 # number on local computer.
15 s.connect(('127.0.0.1', port))
16
17 # receive message string from
18 # server, at a time 1024 B
19
20
21 msg = s.recv(1024)
22
23 # repeat as long as message
24 # string are not empty
25 while msg:
26     if(msg.decode() == "ack"):
27         print("disconnecting from client")
28         exit(0)
29     print('Server says : ' + msg.decode())
30     if(msg.decode() == "enter e to encrypt and d to decrypt and f to disconnect"):

```

```
31     ans = str(input("Your answer : "))
32     s.send(ans.encode())
33     if(msg.decode() == "enter message to encrypt"):
34         ans = str(input("message to encrypt : "))
35         s.send(ans.encode())
36     if(msg.decode() == "enter message to decrypt"):
37         ans = str(input("message to decrypt : "))
38         s.send(ans.encode())
39     msg = s.recv(1024)
40
41
42 # disconnect the client
43 s.close()
```

Output Screenshots:

Running Server file on terminal:

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19043.1586]
(c) Microsoft Corporation. All rights reserved.

D:\PES\Semester_4\Computer Networks\Project\Socket_Programming>python server.py
CONNECTION FROM: ('127.0.0.1', 59769)
b'e'
naren
sfwjs
b'e'
sfwjs
xkbox
b'd'
sfwjs
naren
b'f'
disconnecting ('127.0.0.1', 59769)

D:\PES\Semester_4\Computer Networks\Project\Socket_Programming>
```

Running Client file on terminal:

```
C:\Windows\System32\cmd.exe

Microsoft Windows [Version 10.0.19043.1586]
(c) Microsoft Corporation. All rights reserved.

D:\PES\Semester_4\Computer Networks\Project\Socket_Programming>python client.py
Server says :enter e to encrypt and d to decrypt and f to disconnect
Your answer : e
Server says :enter message to encrypt
message to encrypt : naren
Server says :sfwjs
Server says :enter e to encrypt and d to decrypt and f to disconnect
Your answer : e
Server says :enter message to encrypt
message to encrypt : sfwjs
Server says :xkbox
Server says :enter e to encrypt and d to decrypt and f to disconnect
Your answer : d
Server says :enter message to decrypt
message to decrypt : sfwjs
Server says :naren
Server says :enter e to encrypt and d to decrypt and f to disconnect
Your answer : f
disconnecting from client

D:\PES\Semester_4\Computer Networks\Project\Socket_Programming>
```

Wireshark Output: Following the TCP stream on port 5000

Wireshark packet capture showing a TCP stream on port 5000. The packet list shows several TCP packets between 127.0.0.1 and 127.0.0.1. The packet details pane shows the selected packet (No. 4) with its structure: Ethernet II, Internet Protocol Version 4, and Transmission Control Protocol. The packet bytes pane shows the raw data of the selected packet.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.0.1	TCP	69	57070 → 5000 [PSH, ACK] Seq=1 Ack=1 Win=512 Len=1 TSval=807
2	0.000052134	127.0.0.1	127.0.0.1	TCP	68	5000 → 57070 [ACK] Seq=1 Ack=2 Win=512 Len=0 TSval=807
3	0.000200113	127.0.0.1	127.0.0.1	TCP	92	5000 → 57070 [PSH, ACK] Seq=1 Ack=2 Win=512 Len=24 TSval=807
4	0.000210656	127.0.0.1	127.0.0.1	TCP	68	57070 → 5000 [ACK] Seq=2 Ack=25 Win=512 Len=0 TSval=807
10	5.229862495	127.0.0.1	127.0.0.1	TCP	73	57070 → 5000 [PSH, ACK] Seq=2 Ack=25 Win=512 Len=5 TSval=807
11	5.229890089	127.0.0.1	127.0.0.1	TCP	68	5000 → 57070 [ACK] Seq=25 Ack=7 Win=512 Len=0 TSval=807
12	5.230169494	127.0.0.1	127.0.0.1	TCP	73	5000 → 57070 [PSH, ACK] Seq=25 Ack=7 Win=512 Len=5 TSval=807
13	5.230178236	127.0.0.1	127.0.0.1	TCP	68	57070 → 5000 [ACK] Seq=7 Ack=30 Win=512 Len=0 TSval=807
14	5.230209739	127.0.0.1	127.0.0.1	TCP	123	5000 → 57070 [PSH, ACK] Seq=30 Ack=7 Win=512 Len=55 TSval=807
15	5.230214280	127.0.0.1	127.0.0.1	TCP	68	57070 → 5000 [ACK] Seq=7 Ack=85 Win=512 Len=0 TSval=807

Wireshark · Follow TCP Stream (tcp.stream eq 0) · any

```
enter message to encryptnarensfwjsenter e to encrypt and d to decrypt and f to disconnect
```