



ACADEMIA DE STUDII ECONOMICE BUCUREȘTI
FACULTATEA DE CIBERNETICĂ, STATISTICĂ ȘI INFORMATICĂ ECONOMICĂ
SPECIALIZAREA INFORMATICĂ ECONOMICĂ

Rețele de calculatoare

Programe de tip client-server în limbajele C și Python

Coordonator științific

Prof. univ. dr. TIMOFTE Carmen Manuela

Student

Coman Claudia Ana-Maria, grupa 1082

MAI 2023 BUCUREȘTI

Cuprins

| | |
|--|----|
| Capitolul 1 | 3 |
| 1.1 Client – Server TCP/IP in C: Serverul criptează mesajul primit de la „n” utilizatori și returnează un mesaj de confirmare. | 4 |
| 1.2 Print screen cu rularea intregului program | 13 |
| 1.3 Portul cu identificarea protocolului | 14 |
| Capitolul 2 | 15 |
| 2.1 Client – Server TCP/IP in Python: Serverul criptează mesajul primit de la un utilizator și returnează un mesaj de confirmare. | 15 |
| 2.3 Programul Client | 18 |
| 2.4 Rularea programului client | 20 |
| 2.5 Rularea intregului program | 20 |
| 2.6 Portul cu identificarea protocolului | 21 |

Capitolul 1

Proiectul implementat constă într-un program de tipul client-server care permite criptarea mesajelor utilizând algoritmul de substituție Cezar.

Programul este implementat atât în limbajul de programare C pentru a cunoaște ceea ce se află în spatele metodelor din Python, cât și pentru o înțelegere mai profundă a conceptelor de programare în rețea, dar și în Python. De asemenea, programul permite conectarea unui singur client la program și trimiterea unui singur mesaj per conexiune.

Funcționalitatea proiectului este următoarea:

1. Serverul:

- Primește conexiune de la client.
- CripTEAZĂ mesajele primite de la client în mod automat, utilizând algoritmul de substituție Cezar.
- Mesajele primite devin criptate în mod automat în server.
- Aceste mesaje sunt afisate în server ca devin criptate.
- Se trimite înapoi către client mesaj de confirmare a criptării mesajului.

2. Clientul:

- Se conectează la serverul specificat prin adresa IP și portul corespunzător.
- Permite utilizatorului să introducă un mesaj.
- Trimite mesajul către server pentru criptare.
- Așteaptă răspunsul de la server.

Algoritmul de criptare utilizat este unul simplu și se bazează pe substituția fiecărei litere din mesajul original cu o altă literă din alfabet, deplasată cu un anumit număr de poziții înainte. În cazul acestui proiect, se utilizează substituția Cezar cu o cheie fixă de 3, adică fiecare literă este înlocuită cu cea care se află la 3 poziții înainte în alfabet.

Proiectul oferă o modalitate simplă de a demonstra și înțelege conceptele de bază ale criptografiei și comunicării socket de tipul TCP/IP client-server utilizând atât limbajul de programare C cât și Python.

1.1 Client – Server TCP/IP in C: Serverul criptează mesajul primit de la „n” utilizatori și returnează un mesaj de confirmare.

Programul conține două secțiuni: prima secțiune cuprinde partea de server, iar a doua cea de client.

Program Server:

```
//includere biblioteci pentru creare socket
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <string.h>
```

```
#include <unistd.h>
```

```
#include <sys/types.h>
```

```
#include <sys/socket.h>
```

```
#include <netinet/in.h>
```

```
// functie pentru criptarea mesajului
```

```
char* encrypt(char* message) {
```

```
    char* encrypted = (char*) malloc(strlen(message) + 1);
```

```
    int i = 0;
```

```
//se parcurge mesajul dar ca parametru si daca litera este majuscula/litera mica, caracterul  
se va inlocui cu litera de pe a treia pozitie
```

```
    while (message[i] != '\0') {
```

```
        if (message[i] >= 'A' && message[i] <= 'Z') {
```

```
            encrypted[i] = 'A' + (message[i] - 'A' + 3) % 26;
```

```

    } else if (message[i] >= 'a' && message[i] <= 'z') {

        encrypted[i] = 'a' + (message[i] - 'a' + 3) % 26;

    } else {

        encrypted[i] = message[i];

    }

    i++;

}

encrypted[i] = '\0';

return encrypted;

}

```

```

int main(int argc, char *argv[]) {

```

```

    //descriptor de fisier pt socket ul serverului, descriptor de fisier pentru socket ul clientului
    conectat, portul

```

```

    int sockfd, newsockfd, portno;

```

```

    //dimensiunea socketului

```

```

    socklen_t clilen;

```

```

    char buffer[256];

```

```

//structura ce contine adresa si portul serverului

struct sockaddr_in serv_addr, cli_addr;

int n;

// cream socket-ul pentru asteptarea conexiunilor de la clienti

sockfd = socket(AF_INET, SOCK_STREAM, 0);

if (sockfd < 0) {

    perror("ERROR opening socket");

    exit(1);

}

// initializam structura pentru adresa serverului

bzero((char *) &serv_addr, sizeof(serv_addr));

portno = 1234;

serv_addr.sin_family = AF_INET;

serv_addr.sin_addr.s_addr = INADDR_ANY;

serv_addr.sin_port = htons(portno);

// asociem adresa serverului cu socket-ul

if (bind(sockfd, (struct sockaddr *) &serv_addr, sizeof(serv_addr)) < 0) {

    perror("ERROR on binding");

    exit(1);

}

```

```

// asteptam conexiuni de la clienti

listen(sockfd, 5);

clilen = sizeof(cli_addr);

while (1) {

    // acceptam o conexiune de la un client

    newsockfd = accept(sockfd, (struct sockaddr *) &cli_addr, &clilen);

    if (newsockfd < 0) {

        perror("ERROR on accept");

        exit(1);

    }

    // citim mesajul de la client si il criptam

    bzero(buffer, 256);

    n = read(newsockfd, buffer, 255);

    if (n < 0) {

        perror("ERROR reading from socket");

        exit(1);

    }

    //afisam mesajul primit

    printf("Received message: %s\n", buffer);

```

```
//int-un char* ne vom tine mesajul criptat dupa ce apelam functia creata

char* encrypted_message = encrypt(buffer);

//afisam mesajul criptat

printf("Encrypted message: %s\n", encrypted_message);


// trimitem confirmarea criptarii catre client

n = write(newsockfd, "Mesajul a fost criptat cu succes\n", 34);

if (n < 0) {

    perror("ERROR writing to socket");

    exit(1);

}

// inchidem conexiunea cu clientul

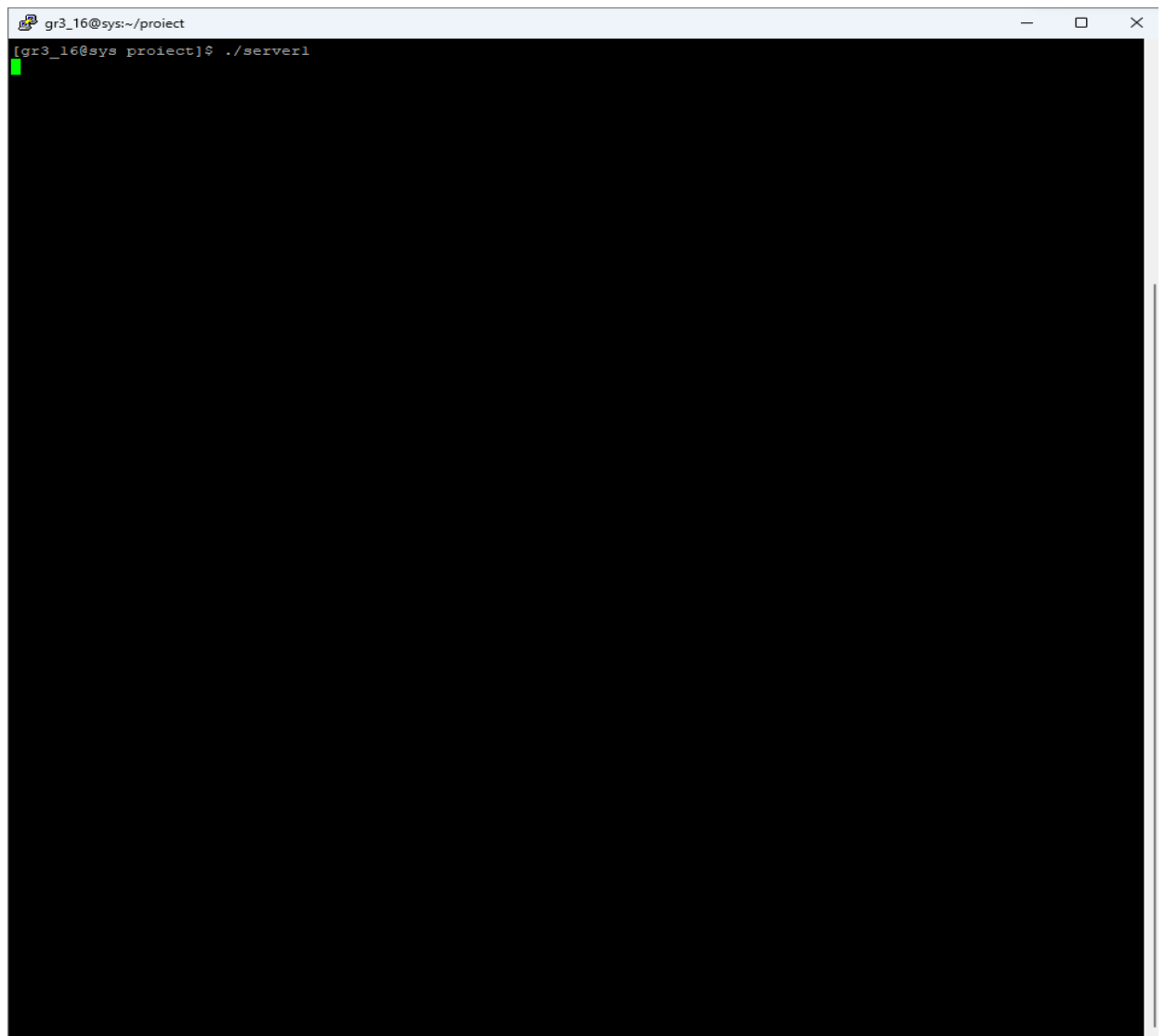
close(newsockfd);

}

close(sockfd);

}
```


Rularea programului server:



A terminal window with a light blue title bar. The title bar contains the text 'gr3_16@sys:~/proiect' on the left and standard window control buttons (minimize, maximize, close) on the right. The terminal area has a black background. The first line of text is '[gr3_16@sys:~/proiect]\$./server1'. A green cursor is positioned at the end of this line.

```
gr3_16@sys:~/proiect
[gr3_16@sys:~/proiect]$ ./server1
```

Program client:

```
#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/socket.h>

#include <arpa/inet.h>

#include <unistd.h>


#define MAX_MESSAGE_SIZE 1024


int main(int argc, char *argv[]) {

    if (argc != 3) {

        fprintf(stderr, "Usage: %s <server_ip_address> <port>\n", argv[0]);

        return 1;

    }


    char *server_ip = argv[1];

    int server_port = atoi(argv[2]);


    // crearea socket-ului clientului

    int client_socket = socket(AF_INET, SOCK_STREAM, 0);

    if (client_socket == -1) {

        perror("Failed to create socket");

        return 1;

    }
```

```

// pregatim informatiile pentru conectarea la server

struct sockaddr_in server_address;

memset(&server_address, 0, sizeof(server_address));

server_address.sin_family = AF_INET;

server_address.sin_addr.s_addr = inet_addr(server_ip);

server_address.sin_port = htons(server_port);


// conectarea la server

if (connect(client_socket, (struct sockaddr*) &server_address, sizeof(server_address))
< 0) {

    perror("Failed to connect to server");

    return 1;

}


// citirea mesajului de la tastatură

char message[MAX_MESSAGE_SIZE];

printf("Introduceți mesajul pentru a fi criptat: ");

fgets(message, MAX_MESSAGE_SIZE, stdin);

message[strcspn(message, "\n")] = 0; // eliminarea caracterului newline (\n)


// trimiterea mesajului catre server

if (send(client_socket, message, strlen(message), 0) < 0) {

    perror("Failed to send message to server");

    return 1;

}

```

```
// primirea răspunsului de la server

char response[MAX_MESSAGE_SIZE];

if (recv(client_socket, response, MAX_MESSAGE_SIZE, 0) < 0) {

    perror("Failed to receive response from server");

    return 1;

}

printf("%s\n", response);

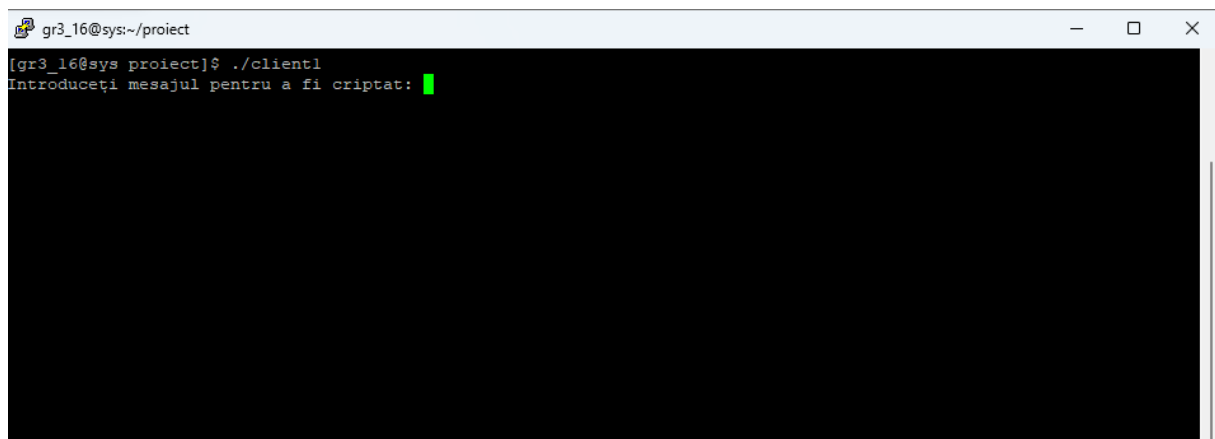
// închiderea socket-ului clientului

close(client_socket);

return 0;

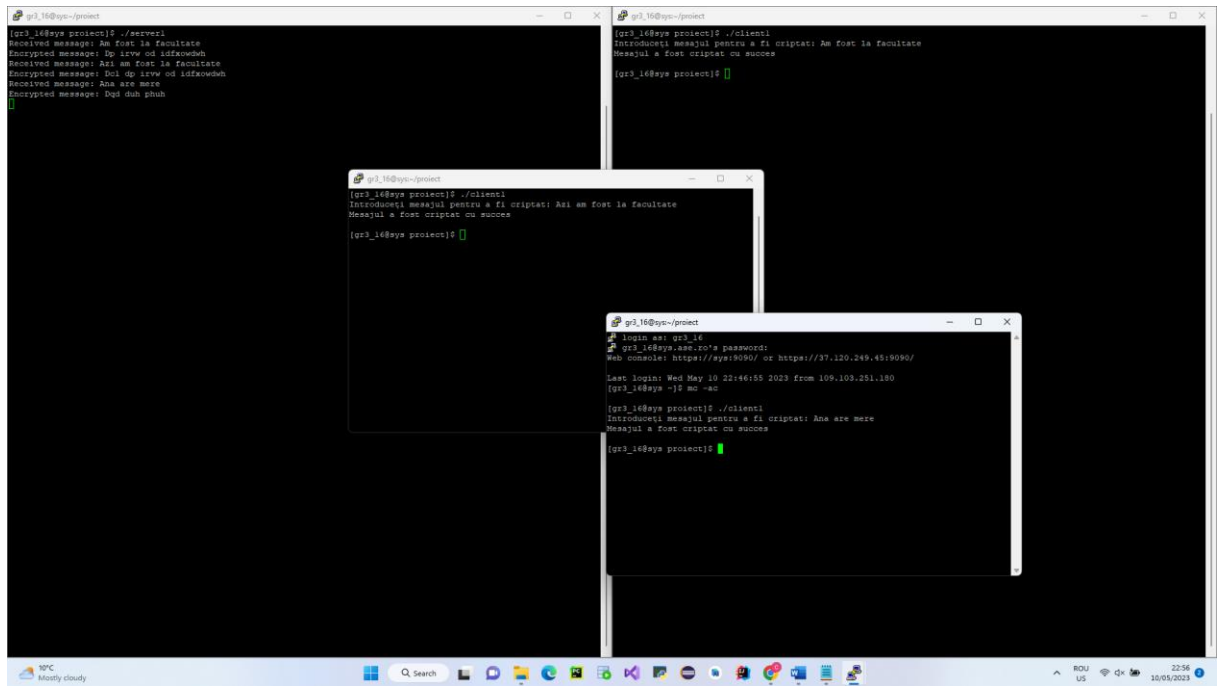
}
```

Rularea clientului:

A terminal window with a title bar showing 'gr3_16@sys:~/proiect'. The terminal content shows the command '[gr3_16@sys proiect]\$./client1' being executed, followed by the prompt 'Introduceți mesajul pentru a fi criptat:'. A green cursor is visible at the end of the prompt. The rest of the terminal area is black.

```
gr3_16@sys:~/proiect
[gr3_16@sys proiect]$ ./client1
Introduceți mesajul pentru a fi criptat: 
```

1. 2 Print screen cu rularea intregului program



```
gr3_16@sys-project: [gr3_16@sys-project]$ ./server1
Received message: Am fost la facultate
Encrypted message: Ip irvw od idfkowdbh
Received message: Azi am fost la facultate
Encrypted message: Dqi do irvw od idfkowdbh
Received message: Ana are mere
Encrypted message: Dqi duh phuh

gr3_16@sys-project: [gr3_16@sys-project]$ ./client1
Introducati mesajul pentru a fi criptat: Am fost la facultate
Mesajul a fost criptat cu succes

gr3_16@sys-project: [gr3_16@sys-project]$

gr3_16@sys-project: [gr3_16@sys-project]$ ./client1
Introducati mesajul pentru a fi criptat: Azi am fost la facultate
Mesajul a fost criptat cu succes

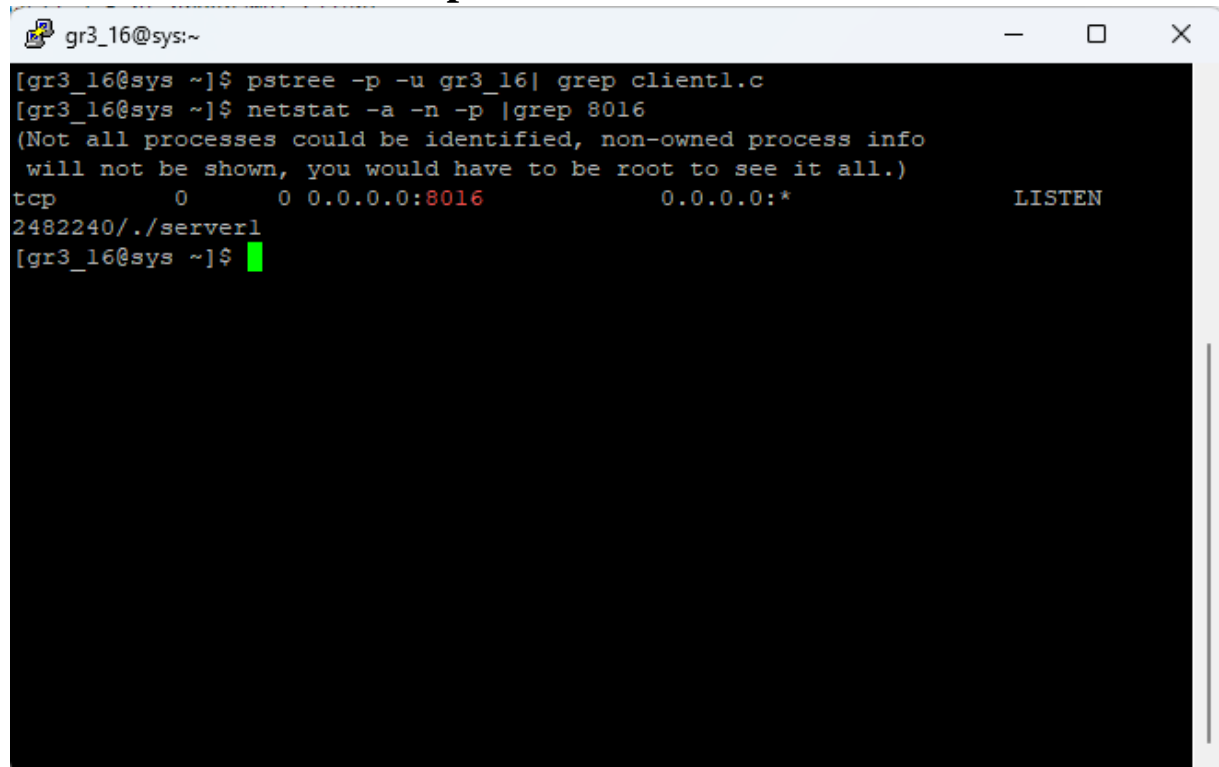
gr3_16@sys-project: [gr3_16@sys-project]$

gr3_16@sys-project: [gr3_16@sys-project]$ ./client1
Introducati mesajul pentru a fi criptat: Ana are mere
Mesajul a fost criptat cu succes

gr3_16@sys-project: [gr3_16@sys-project]$
```

In acest print screen apar 3 conexiuni de client ce trimit mesaje diferite catre acelasi server, serverul la randul lui le cripteaza utilizand algoritmul de substitutie Cezar, ulterior se trimite inapoi catre client un mesaj de confirmare a faptului ca mesajul transmis este criptat cu succes.

1.3 Portul cu identificarea protocolului



```
gr3_16@sys:~  
[gr3_16@sys ~]$ pstree -p -u gr3_16 | grep client1.c  
[gr3_16@sys ~]$ netstat -a -n -p | grep 8016  
(Not all processes could be identified, non-owned process info  
will not be shown, you would have to be root to see it all.)  
tcp        0      0 0.0.0.0:8016          0.0.0.0:*            LISTEN  
2482240/./server1  
[gr3_16@sys ~]$
```

Capitolul 2

2.1 Client – Server TCP/IP in Python: Serverul criptează mesajul primit de la un utilizator și returnează un mesaj de confirmare.

Programul permite utilizatorului sa trimita mai multe mesaje catre server, serverul le cripteaza insa pe fiecare in parte.

Programul conține două secțiuni: prima secțiune cuprinde partea de server, iar a doua cea de client.

Program Server:

```
import socket

def encrypt(message):

    # algoritm de criptare simplu (substituție)

    encrypted = ""

    #parcugem fiecare caracter din mesaj

    for char in message:

        #verificam daca caracterul este o litera

        if char.isalpha():

            #convertim caracterul in cod ASCII

            char_code = ord(char)
```

#Verificăm dacă caracterul este o literă majusculă.

if char.isupper():

#Aplicăm algoritmul de substituție pentru literele majuscule, utilizând formula de deplasare cu 3 poziții și asigurându-ne că rămânem în intervalul ASCII pentru literele majuscule (65-90).

char_code = (char_code + 3 - 65) % 26 + 65

else:

#Aplicăm algoritmul de substituție pentru literele minuscule, utilizând formula de deplasare cu 3 poziții și asigurându-ne că rămânem în intervalul ASCII pentru literele minuscule (97-122).

char_code = (char_code + 3 - 97) % 26 + 97

#Convertim codul ASCII înapoi în caracter și îl adăugăm la șirul criptat.

encrypted += chr(char_code)

else:

encrypted += char

return encrypted

HOST = '127.0.0.1'

PORT = 65432


```
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
```

```
#Legăm socket-ul la adresa IP și portul specificate.
```

```
s.bind((HOST, PORT))
```

```
s.listen()
```

```
conn, addr = s.accept()
```

```
with conn:
```

```
    print('Connected by', addr)
```

```
    while True:
```

```
        data = conn.recv(1024)
```

```
        if not data:
```

```
            break
```

```
        message = data.decode()
```

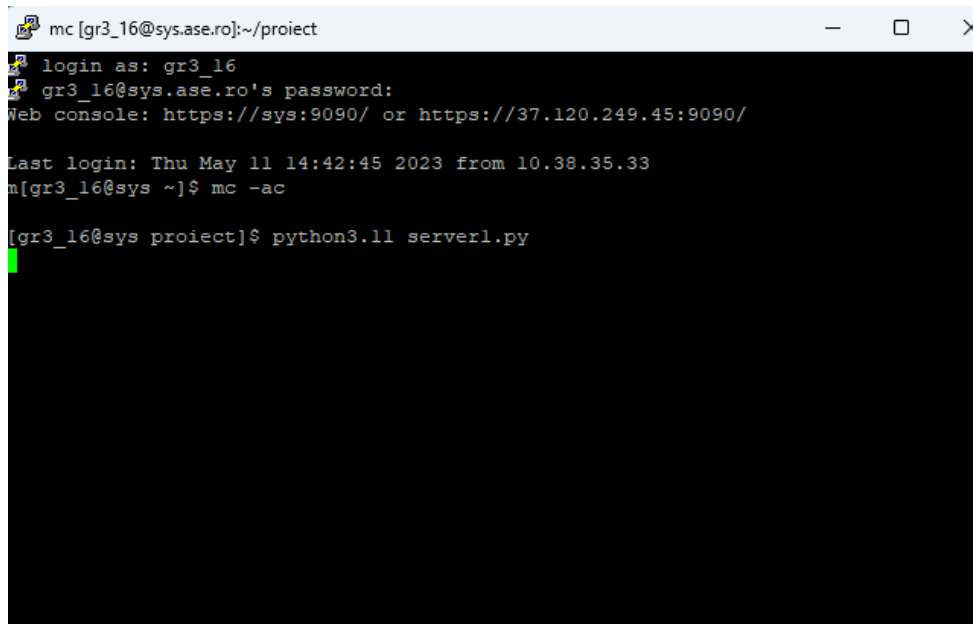
```
        print('Received message:', message)
```

```
        encrypted_message = encrypt(message)
```

```
        print('Encrypted message:', encrypted_message)
```

```
        conn.sendall(b'Mesajul a fost criptat cu succes\n')
```

2.2 Rularea programului server

A terminal window titled 'mc [gr3_16@sys.ase.ro]:~/proiect' with standard window controls. The terminal shows a login sequence for user 'gr3_16' on 'sys.ase.ro', including a password prompt and a web console URL. It displays the last login time and IP. The user runs 'mc -ac' and then 'python3.11 server1.py', with a green cursor visible on the second command line.

```
mc [gr3_16@sys.ase.ro]:~/proiect
login as: gr3_16
gr3_16@sys.ase.ro's password:
Web console: https://sys:9090/ or https://37.120.249.45:9090/

Last login: Thu May 11 14:42:45 2023 from 10.38.35.33
m[gr3_16@sys ~]$ mc -ac

[gr3_16@sys proiect]$ python3.11 server1.py
```

2.3 Programul Client

```
import socket

# setăm adresa IP și portul serverului

IP = '127.0.0.1'

PORT = 65432

# creăm socket-ul pentru a ne conecta la server

client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# ne conectăm la server

client_socket.connect((IP, PORT))

while True:

    # cerem utilizatorului să introducă mesajul
```

```
message = input('Introduceți mesajul pentru a fi criptat sau tastați "exit" pentru a  
închide conexiunea: ')
```

```
if message.lower() == 'exit':
```

```
    break
```

```
# trimitem mesajul către server
```

```
client_socket.sendall(message.encode())
```

```
# primim confirmarea că mesajul a fost criptat cu succes
```

```
response = client_socket.recv(1024).decode()
```

```
print(response)
```

```
# închidem conexiunea cu serverul
```

```
client_socket.close()
```

2.4 Rularea programului client

```
gr3_16@sys:~/proiect
login as: gr3_16
gr3_16@sys.ase.ro's password:
Web console: https://sys:9090/ or https://37.120.249.45:9090/

Last login: Thu May 11 14:49:58 2023 from 10.38.35.33
[gr3_16@sys ~]$ mc -ac

[gr3_16@sys proiect]$ python3.11 client1.py
Introduceți mesajul pentru a fi criptat sau tastați "exit" pentru a închide conexiunea: ana
Mesajul a fost criptat cu succes

Introduceți mesajul pentru a fi criptat sau tastați "exit" pentru a închide conexiunea: █
```

2.5 Rularea intregului program

```
gr3_16@sys:~/proiect
login as: gr3_16
gr3_16@sys.ase.ro's password:
Web console: https://sys:9090/ or https://37.120.249.45:9090/

Last login: Thu May 11 14:42:45 2023 from 10.38.35.33
[gr3_16@sys ~]$ mc -ac

[gr3_16@sys proiect]$ python3.11 server1.py
Connected by ("37.120.249.45", 38192)
Received message: ana
Encrypted message: dgd
Received message: A fost criptat cu succes
Encrypted message: 0 lrvv f4lndw fx wffhv
Received message: Se poate observa
Encrypted message: 7b xzdbn zvtboud
█

gr3_16@sys:~/proiect
login as: gr3_16
gr3_16@sys.ase.ro's password:
Web console: https://sys:9090/ or https://37.120.249.45:9090/

Last login: Thu May 11 14:49:58 2023 from 10.38.35.33
[gr3_16@sys ~]$ mc -ac


[gr3_16@sys proiect]$ python3.11 client1.py
Introduceți mesajul pentru a fi criptat sau tastați "exit" pentru a închide conexiunea: ana
Mesajul a fost criptat cu succes

Introduceți mesajul pentru a fi criptat sau tastați "exit" pentru a închide conexiunea: A fost criptat cu succes
Mesajul a fost criptat cu succes

Introduceți mesajul pentru a fi criptat sau tastați "exit" pentru a închide conexiunea: Se poate observa
Mesajul a fost criptat cu succes

Introduceți mesajul pentru a fi criptat sau tastați "exit" pentru a închide conexiunea: █
```

2.6 Portul cu identificarea protocolului

 gr3_16@sys:~/proiect

```
[gr3_16@sys proiect]$ netstat -a -n -p |grep 8016
(Not all processes could be identified, non-owned process info
will not be shown, you would have to be root to see it all.)
tcp        0      0 37.120.249.45:8016  0.0.0.0:*          LISTEN
2510288/python3.11
tcp        0      0 37.120.249.45:8016  37.120.249.45:38192 ESTABLISHED
2510288/python3.11
tcp        0      0 37.120.249.45:38192 37.120.249.45:8016 ESTABLISHED
2510451/python3.11
[gr3_16@sys proiect]$
```