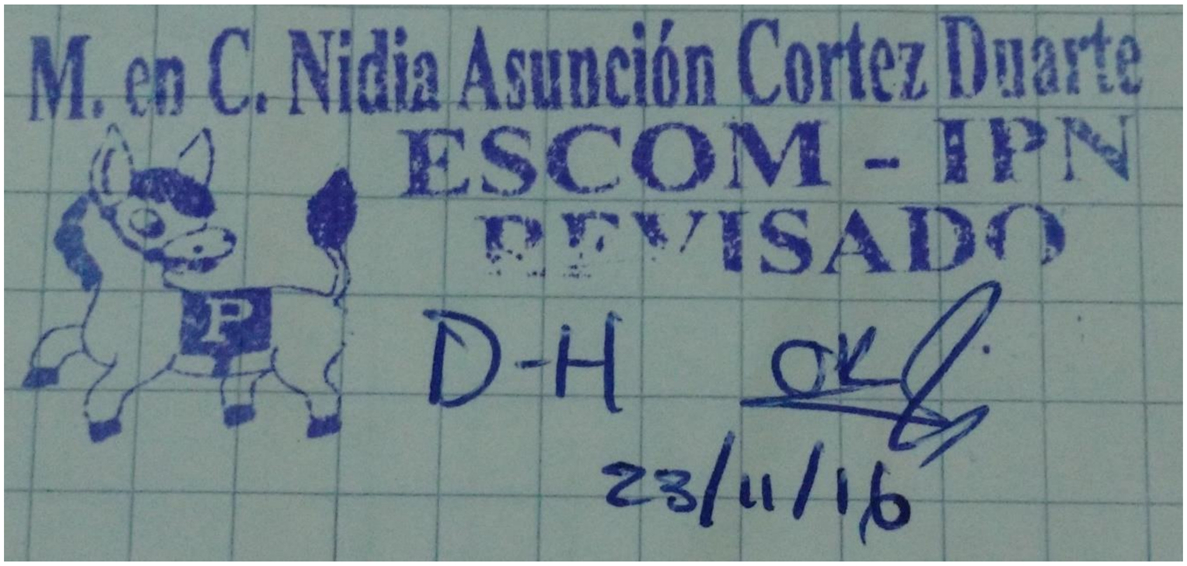
Instituto Politécnico Nacional
Escuela Superior de Cómputo
Cryptography

Practice 6: Diffie Hellman (Network)

By:
Moreno Zárate Víctor Gibrán
Leonardo Manzano Salazar

Professor:
M. en C. NIDIA ASUNCIÓN CORTEZ DUARTE
November 2016


M. en C. Nidia Asunción Cortez Duarte
ESCOM - IPN
REVISADO
D-H OK [illegible]
23/11/16

Contents

Problem

What is the behaviour of the Diffie Hellman?

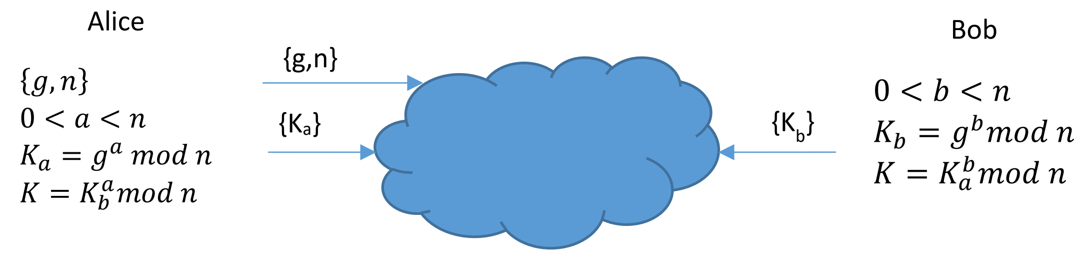
What are the main characteristics of the Diffie Hellman?

How does this method helps us to share a common key without send the real value?

Diffie–Hellman key exchange (D–H) is a specific method of securely exchanging cryptographic keys over a public channel and was one of the first public-key protocols as originally conceptualized by Ralph Merkle and named after Whitfield Diffie and Martin Hellman. D–H is one of the earliest practical examples of public key exchange implemented within the field of cryptography. [1]

Traditionally, secure encrypted communication between two parties required that they first exchange keys by some secure physical channel, such as paper key lists transported by a trusted courier. The Diffie–Hellman key exchange method allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure channel. This key can then be used to encrypt subsequent communications using a symmetric key cipher. [1]

Diffie-Hellman Scheme



Hypothesis

A possible solution to the problem is to write a computer program in order to analyse the behaviour of the Diffie Hellman, this program will be able to encrypt and decrypt a text using the key generated with this technique. It also will work over a network.

If we program this software correctly, we will see how the character data is send and decrypted using the key previously computed using D-H. To encrypt/decrypt the messages, we will use the DES cipher block.

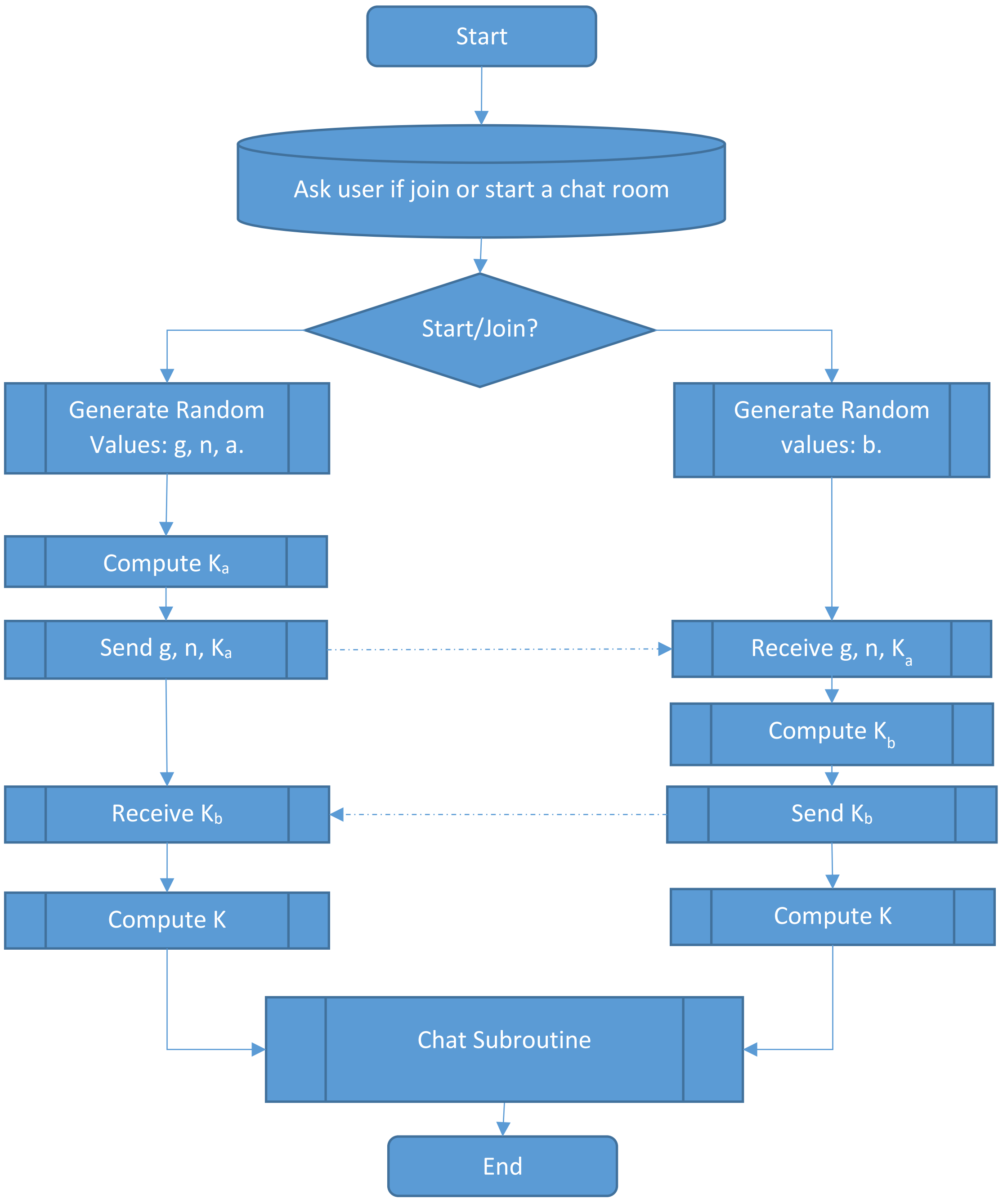
Software (libraries, package, tools)

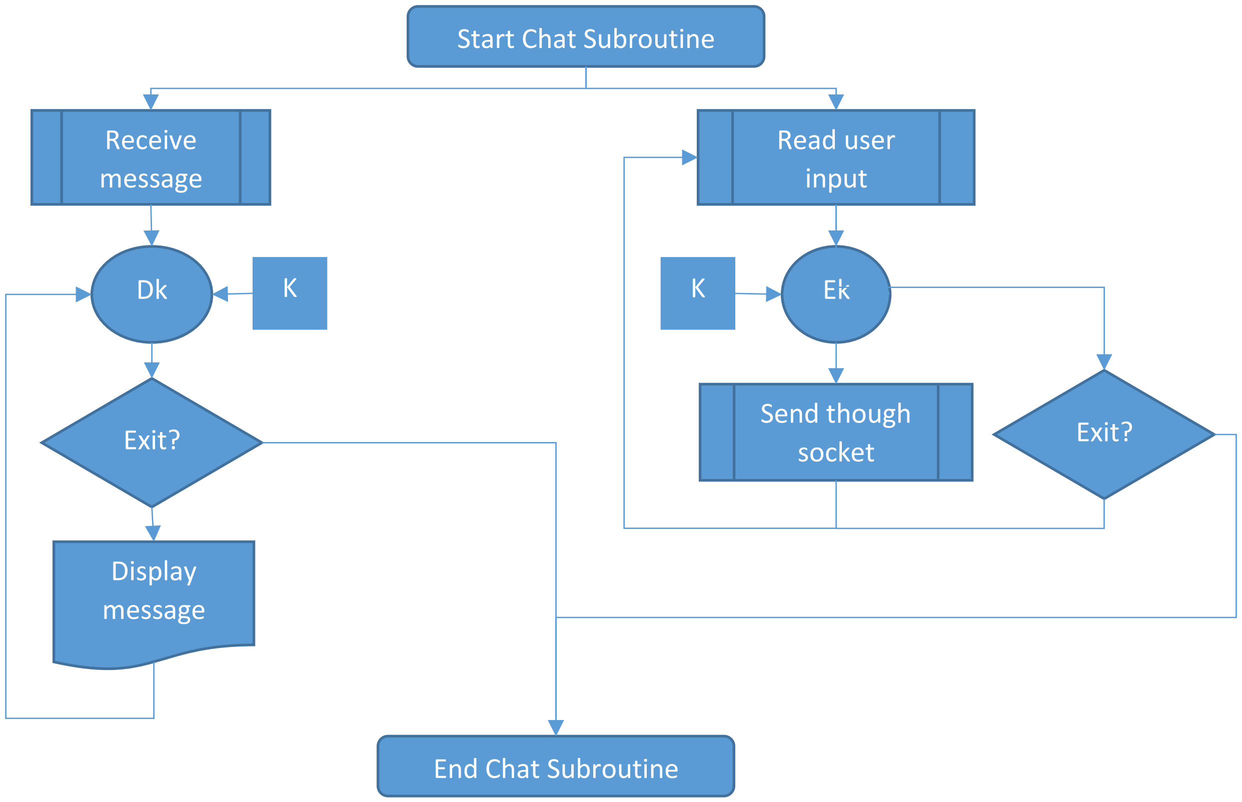
To do this practice, we used:

- Personal Computer
- Linux Operating System
- Text Editor

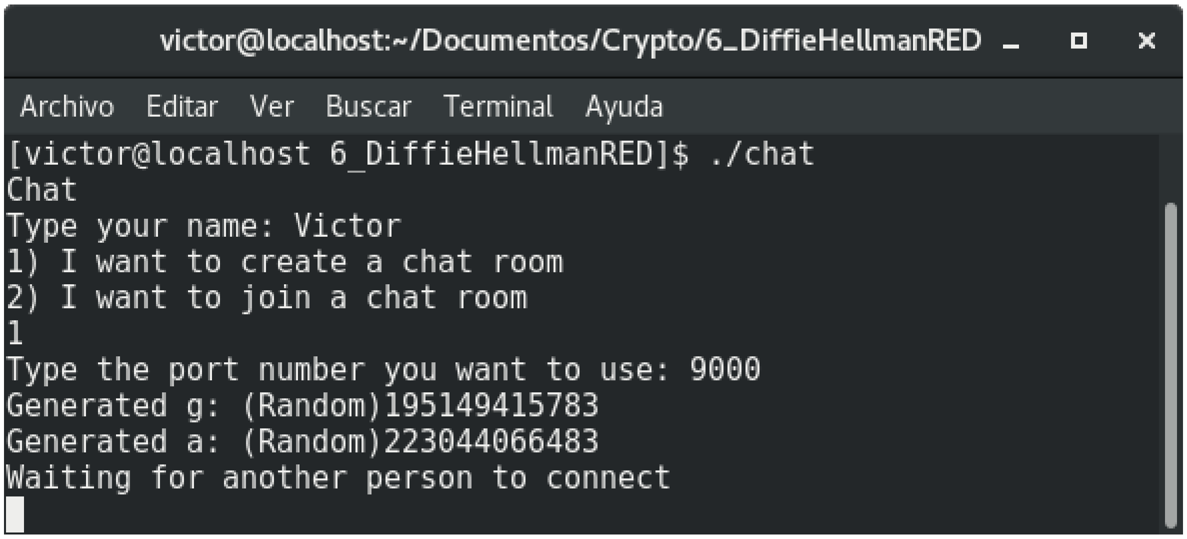
- GNU C++ Compiler
- Access to a computer network

Procedure



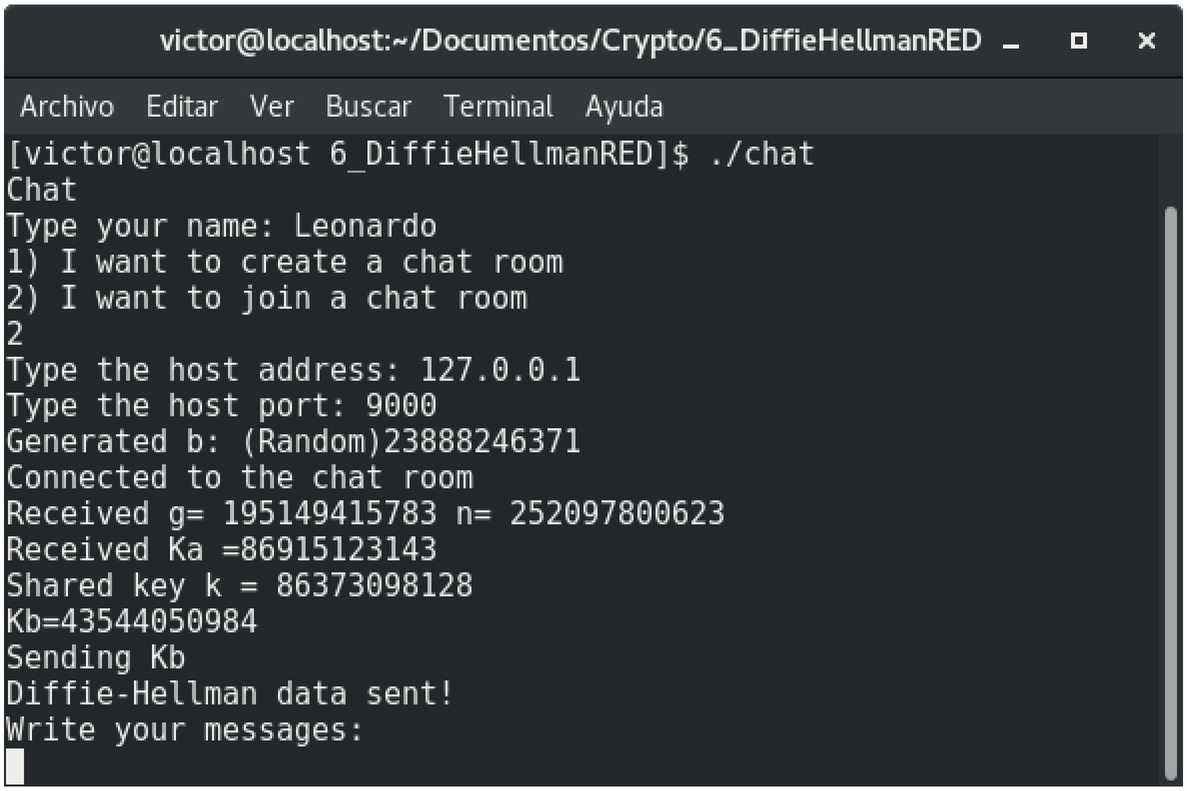


Results (Data)


victor@localhost:~/Documentos/Crypto/6_DiffieHellmanRED
Archivo Editar Ver Buscar Terminal Ayuda

```
[victor@localhost 6_DiffieHellmanRED]$ ./chat
Chat
Type your name: Victor
1) I want to create a chat room
2) I want to join a chat room
1
Type the port number you want to use: 9000
Generated g: (Random)195149415783
Generated a: (Random)223044066483
Waiting for another person to connect
```

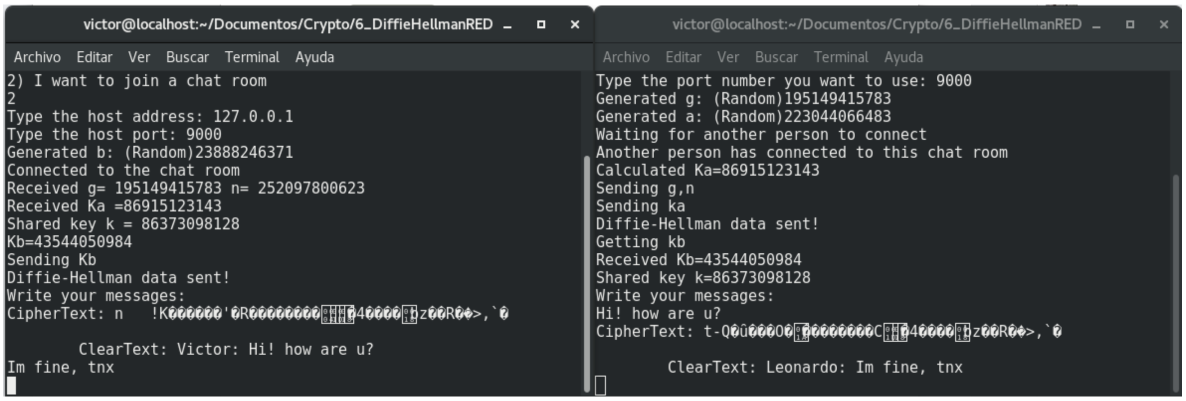
Figure 1: Starting a Chat Room



victor@localhost:~/Documentos/Crypto/6_DiffieHellmanRED
Archivo Editar Ver Buscar Terminal Ayuda

```
[victor@localhost 6_DiffieHellmanRED]$ ./chat
Chat
Type your name: Leonardo
1) I want to create a chat room
2) I want to join a chat room
2
Type the host address: 127.0.0.1
Type the host port: 9000
Generated b: (Random)23888246371
Connected to the chat room
Received g= 195149415783 n= 252097800623
Received Ka =86915123143
Shared key k = 86373098128
Kb=43544050984
Sending Kb
Diffie-Hellman data sent!
Write your messages:
```

Figure 2: Joining a Chat Room



victor@localhost:~/Documentos/Crypto/6_DiffieHellmanRED
Archivo Editar Ver Buscar Terminal Ayuda
victor@localhost:~/Documentos/Crypto/6_DiffieHellmanRED
Archivo Editar Ver Buscar Terminal Ayuda

```
2) I want to join a chat room
2
Type the host address: 127.0.0.1
Type the host port: 9000
Generated b: (Random)23888246371
Connected to the chat room
Received g= 195149415783 n= 252097800623
Received Ka =86915123143
Shared key k = 86373098128
Kb=43544050984
Sending Kb
Diffie-Hellman data sent!
Write your messages:
CipherText: n   !K������'�R���������������4�����z��R��>,`�

       ClearText: Victor: Hi! how are u?
Im fine, tnx
```

```
Type the port number you want to use: 9000
Generated g: (Random)195149415783
Generated a: (Random)223044066483
Waiting for another person to connect
Another person has connected to this chat room
Calculated Ka=86915123143
Sending g,n
Sending ka
Diffie-Hellman data sent!
Getting kb
Received Kb=43544050984
Shared key k=86373098128
Write your messages:
Hi! how are u?
CipherText: t-Q�û���O���������C���4�����z��R��>,`�

       ClearText: Leonardo: Im fine, tnx
```

Figure 3: Sending/Receiving encrypted data and displaying it into the terminal.

Conclusions

In this practice, we accepted the hypothesis because we could send a common key without sending the real value though an insecure channel, using the Diffie-Hellman scheme.

We could send and receive data encrypted with the DES block cipher, and using the previously common key obtained with the D-H, we could transform it into plaintext. This is useful because we made a basic lightweight "secure" chat using D-H and DES.

Reference

[1] “Diffie–Hellman key exchange", *Wikipedia*, 2016. [Online]. Available: https://en.wikipedia.org/wiki/Diffie%E2%80%93Hellman_key_exchange . [Accessed: 27- Nov-2016].

Code

```
#include "./librerias/SocketFlujo.hpp"
#include <iostream>
#include <random>

#include <unistd.h>
#include <sys/wait.h>
#include <openssl/des.h>
#define tam_buffer 255
using namespace std;


unsigned long long int mulmod(unsigned long long int a,unsigned long long int b,unsigned long long int c){
    unsigned long long int x = 0,y=a%c;
    while(b > 0){
        if(b%2 == 1){
            x = (x+y)%c;
        }
        y = (y*2)%c;
        b /= 2;
    }
    return x%c;
}

struct messageDH{
    unsigned long long int g;
    unsigned long long int n;
};

struct keyDH{
    unsigned long long int kn;
};

struct common_msg{
    char data[tam_buffer];
    };

int main(){
    SocketFlujo *socket;
    string port;
    string option;
    string name;
    //Diffie-Hellman Variables
    unsigned long long int g;
    unsigned long long int n;
    unsigned long long int a;
    unsigned long long int b;
    unsigned long long int k;
    //Random
    std::random_device rd;
```

```cpp
    std::mt19937 rng(rd());    // random-number engine used (Mersenne-Twister in this case)
    std::uniform_int_distribution<unsigned long long int> uni(1,252097800623-1);


    cout<<"Chat"<<endl;
    cout<<"Type your name: ";
    getline(cin, name);
    name+=": ";

    cout<<"1) I want to create a chat room"<<endl;
    cout<<"2) I want to join a chat room"<<endl;
    getline(cin, option);

    if(option=="1"){
        SocketFlujo s_in=SocketFlujo();
        cout<<"Type the port number you want to use: ";
        getline(cin, port);
        s_in.enlazar(stoi(port));
        //Diffie-Hellman Code
        string number_input;
        g=uni(rng);
        cout<<"Generated g: (Random)"<<g<<endl;
        n=uni(rng);
        a=uni(rng);
        cout<<"Generated a: (Random)"<<a<<endl;
        //Waiting for another client
        cout<<"Waiting for another person to connect"<<endl;
        s_in.escucha();
        socket=new SocketFlujo(s_in.aceptar());
        cout<<"Another person has connected to this chat room"<<endl;
    }else if(option=="2"){
        socket= new SocketFlujo();
        string ip;
        cout<<"Type the host address: ";
        getline(cin, ip);
        cout<<"Type the host port: ";
        getline(cin, port);
        b=uni(rng);
        cout<<"Generated b: (Random)"<<b<<endl;
        socket->conectar(ip,stoi(port));
        cout<<"Connected to the chat room"<<endl;
    }else{
        cout<<"Invalid option"<<endl;
        return 0;
    }

    //Code before fork
    if(option=="1"){//We send g,n,ka
            unsigned long long int k_a;
            k_a=mulmod(g,a,n);
                cout<<"Calculated Ka="<<k_a<<endl;
                //Sending g,a
                cout<<"Sending g,n"<<endl;
                messageDH hello_msg;
                hello_msg.g=g;
                hello_msg.n=n;
                Paquete gn_paq=Paquete((char *)&hello_msg,sizeof hello_msg);
                socket->envia(gn_paq);
                //Sending ka
                cout<<"Sending ka"<<endl;
```

```
                        keyDH key_msg;
                        key_msg.kn=k_a;
                        Paquete k_paq=Paquete((char *)&key_msg,sizeof key_msg);
                        socket->envia(k_paq);
                        cout<<"Diffie-Hellman data sent!"<<endl;
                        ////Getting Kb
                        cout<<"Getting kb"<<endl;
                        Paquete k_paq2=Paquete(sizeof key_msg);
                        socket->recibe(k_paq2);
                        memcpy(&key_msg,k_paq2.obtieneDatos(),sizeof key_msg);
                        cout<<"Received Kb="<<key_msg.kn<<endl;
                        k=mulmod(key_msg.kn,a,n);
                        cout<<"Shared key k="<<k<<endl;

                }else if(option=="2"){//We only send kb once its computed
                        //Getting g,n
                        messageDH hello_msg;
                        Paquete gn_paq=Paquete(sizeof hello_msg);
                        socket->recibe(gn_paq);
                        memcpy(&hello_msg,gn_paq.obtieneDatos(),sizeof hello_msg);
                        g=hello_msg.g;
                        n=hello_msg.n;
                        cout<<"Received g= "<<g<<" n= "<<n<<endl;
                        //Getting Ka
                        keyDH key_msg;
                        Paquete k_paq=Paquete(sizeof key_msg);
                        socket->recibe(k_paq);
                        //Computing k
                        memcpy(&key_msg,k_paq.obtieneDatos(),sizeof key_msg);
                        cout<<"Received Ka ="<<key_msg.kn<<endl;
                        k=mulmod(key_msg.kn,b,n);
                        cout<<"Shared key k = "<<k<<endl;
                        //Computing kb
                        unsigned long long int k_b;
                        k_b=mulmod(g,b,n);
                        cout<<"Kb="<<k_b<<endl;
                        key_msg.kn=k_b;
                        //sending kb
                        cout<<"Sending Kb"<<endl;
                        Paquete k_paq2=Paquete((char *)&key_msg,sizeof key_msg);
                        socket->envia(k_paq2);
                        cout<<"Diffie-Hellman data sent!"<<endl;
                }

                //Chat code
                //DES variables
                DES_key_schedule schedule;
                DES_cblock ivec={0xE1, 0xE2, 0xE3, 0xD4, 0xD5, 0xC6, 0xC7, 0xA8};
                DES_cblock key;
                const int numbits=8;
                //Key manipulation
                memcpy(&key,&k,sizeof key);
                DES_set_key(&key,&schedule);


                pid_t pid;
                pid=fork();
                if(pid>0){//Main process, sender

                    //Chat Code
                    cout<<"Write your messages: "<<endl;
                    Paquete *out;
                    do{
                        string input="";
                        getline(cin, input);
                        input=name+input;
                        //Encrypt message
                        unsigned char input_cipher[tam_buffer];
                        char cleartxt[tam_buffer];
                        strcpy(cleartxt,input.c_str());
                        DES_ofb_encrypt((unsigned char*)cleartxt,input_cipher,numbits,tam_buffer,&schedule,(C_Block *)ivec);
                        //Send message
                        common_msg msg;
                        memcpy(msg.data,input_cipher,tam_buffer);
                        out= new Paquete((char*)&msg,sizeof msg);
                        socket->envia(*out);
                        delete out;

                        //Exit chat
                        if(input.find("--exit")!=string::npos){
                            delete socket;
                            kill(pid, SIGKILL);
                            break;
                        }

                    }while(true);

                }else if(pid==0){//Child process, receiver
                    do{
                        string message="";
                        Paquete *in= new Paquete(sizeof (common_msg));
                        socket->recibe(*in);
                        common_msg msg;
                        memcpy(&msg,in->obtieneDatos(),sizeof msg);
                        message=string(msg.data);
                        //Decrypt message
                        unsigned char output[tam_buffer];
                        DES_ofb_encrypt((unsigned char*)msg.data, output,numbits,tam_buffer,&schedule,(C_Block *)ivec);
                        string text_clr((char *)output);

                        //Exit chat
                        if(text_clr.find("--exit")!=string::npos){
                            cout<<"The other person exited the room"<<endl;
                            delete in;
                            delete socket;
                            kill(getppid(), SIGKILL);
                            break;
                        }

                        //Show messages in console
                        cout<<"CipherText: "<<message<<endl<<endl;
                        cout<<"\tClearText: "<<text_clr<<endl;

                        delete in;
                    }while(true);

                }else{//Error
                    cerr << "Error in fork: "<<strerror(errno) << endl;
                }

                wait(0);

                return 0;

            }
```