

SISTEMAS

Sockets seguros

2020

DISTRIBUIDOS

Murcia Gómez, Francisco Joaquín

48734281H

Universidad de Alicante

# índice

[índice 2](#_Toc57451669)

[Sockets seguros 3](#_Toc57451670)

[Mi Código 3](#_Toc57451671)

[Servidor 3](#_Toc57451672)

[Cliente 5](#_Toc57451673)

[Ejecución 7](#_Toc57451674)

[Capturas de terminal 7](#_Toc57451675)

[Capturas de paquetes 8](#_Toc57451676)

# Sockets seguros

Para la realización de esta práctica hemos segurizado los sockets de la practica 1 con la librería openSSL de C, SSl hace uso de certificados digitales para establecer comunicaciones seguras a través de Internet, dicha conexión es cifrada, en nuestro caso el servidor es el que otorga la clave de cifrado

SSL responde a dos principios:

* **El cifrado:** todos los datos que circulan entre cliente y servidor están cifrados por una clave
* **La autenticación:** se requiere de una autenticación (usuario y contraseña en nuestro caso) para poder empezar a intercambiar información con el servidor

El funcionamiento es muy simple:

1. El cliente solicita el establecimiento de conexión
2. El servidor envía el certificado y se establece una conexión segura
3. El servidor pregunta por una autenticación
4. El cliente se autentica
5. Comienza la comunicación segura

* El servidor envía un mensaje cifrado
* El cliente lo descifra y le contesta con un mensaje cifrado
* El servidor lo descifra y le contesta con un mensaje cifrado

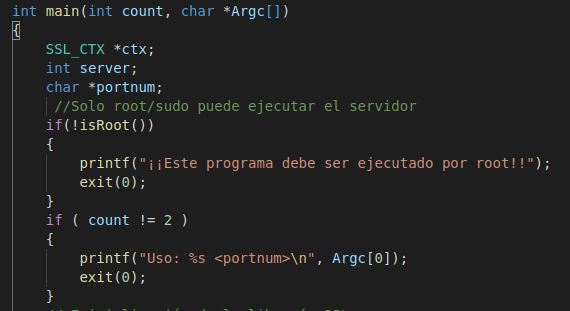
En nuestro caso usamos una clave de 2048 bits de openSSL que generamos con el siguiente comando(la clave es “1234”):

openssl req -x509 -days 365 -newkey rsa:2048 -keyout certServ.pem -out certServ.pem

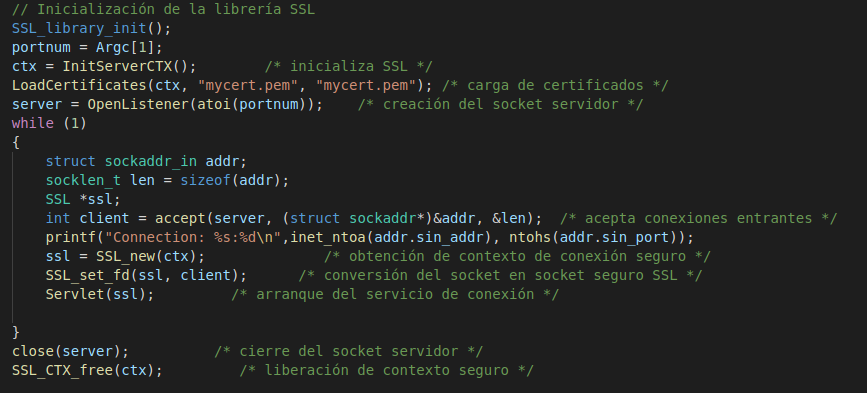
# Mi Código

## Servidor

En primer lugar, comprobamos que el programa no se ejecuta como root, ya que sería una falta de seguridad grave

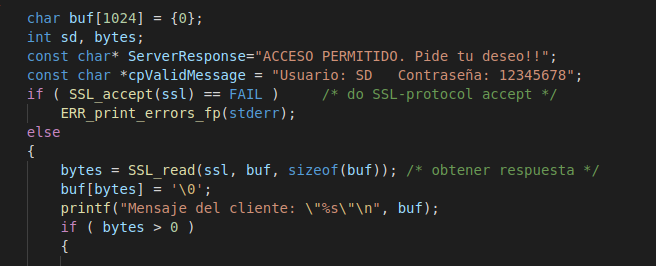


A continuación, iniciamos el protocolo SSL abrimos el socket y realizamos la conexión con el cliente



El módulo void Servlet(SSL\* ssl){}, es donde administramos la conexión, en el enviamos y recibimos mensajes del cliente

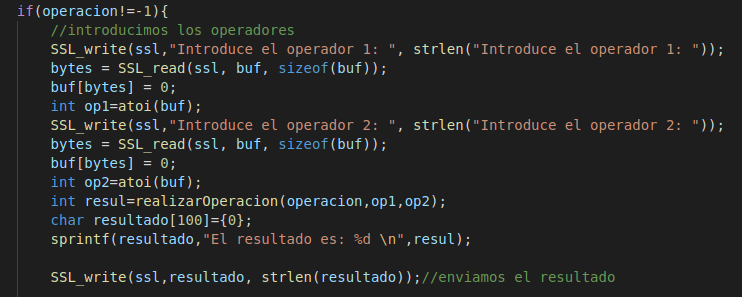
Comprobamos que la credencial del cliente sean correctas



En caso afirmativo, enviaremos el menú y validaremos la opción elegida por el cliente



Comprobamos que sea o multiplicar, sumar o elevar y solicitamos los operandos y mostramos el resultado

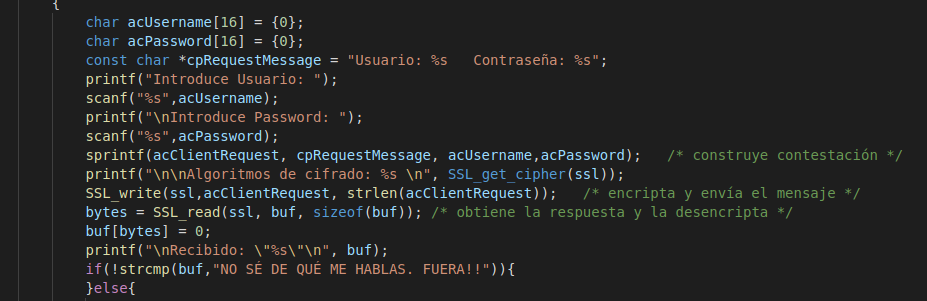


## Cliente

Cos conectamos al servidor con un método no seguro y solicitas una conexión segura



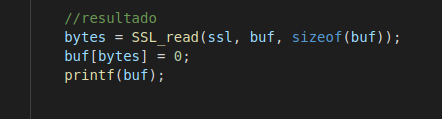
Nos autenticamos al servidor y ciframos los mensajes



Recibimos el menú y le enviamos al servidor el topo de operación validado y los operandos



Por último, recibimos el resultado



# Ejecución

## Capturas de terminal

Para coompilar y ejecutar el programa en el cliente:

* gcc -Wall -o cliente cliente.c -L/usr/lib -lssl -lcrypto
* ./cliente 127.0.0.1 8090

Para coompilar y ejecutar el programa del servidor:

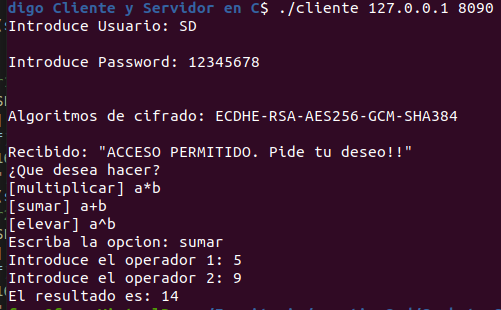
* gcc -Wall -o servidor servidor.c -L/usr/lib -lssl -lcrypto
* sudo ./servidor 8090

A continuación, mostrare el funcionamiento del Código

Ejecutamos el servidor con: ./servidor 8090

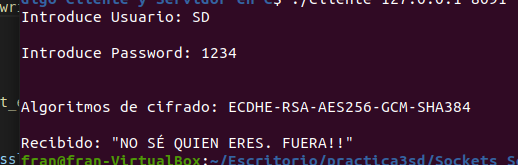
Ejecutamos el servidor con: ./ cliente 127.0.0.1 8090

1. Introducimos el usuario “SD”
2. Introducimos la contraseña “12345678”
3. Escribimos el tipo de operación a realizar “sumar”
4. Escribimos el operando 1 “5”
5. Escribimos el operando 1 “9”
6. Recibimos el resultado “14”



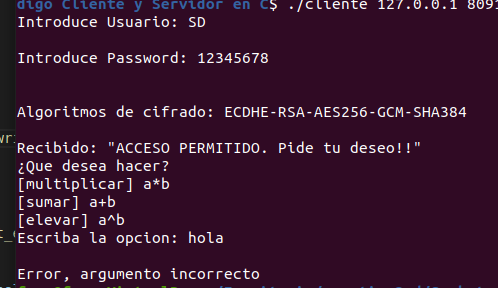
Si nos autenticamos más:

1. Introducimos el usuario “SD”
2. Introducimos la contraseña “1234”
3. El servidor nos echa



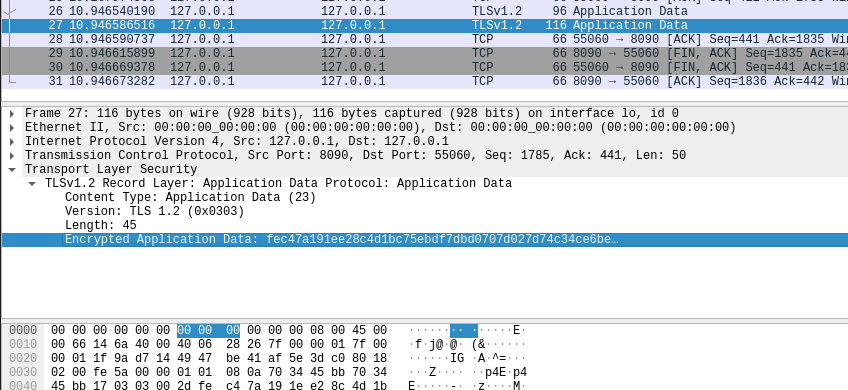
Si seleccionamos mal la operación:

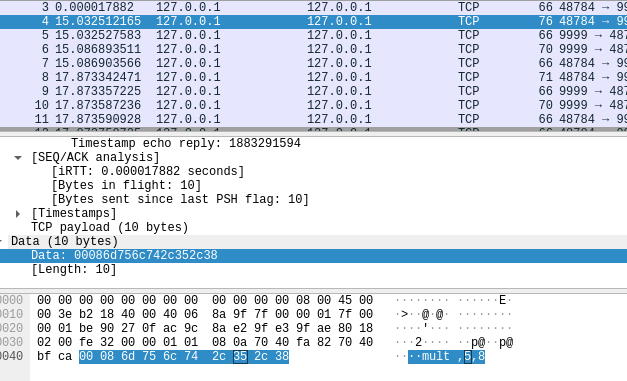
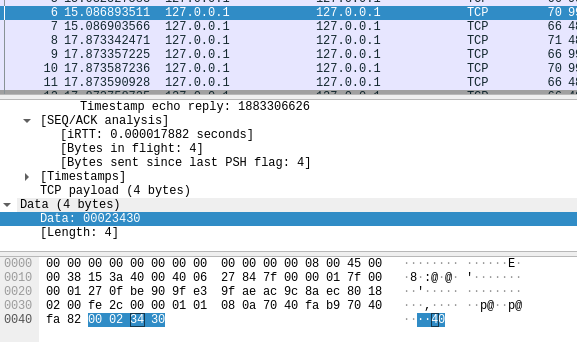
1. Introducimos el usuario “SD”
2. Introducimos la contraseña “12345678”
3. Escribimos el tipo de operación a realizar “hola”
4. Se detecta el error y se aborta



## Capturas de paquetes

Si por ejemplo capturamos los paquetes con el programa de capturas de paquetes wireshark si inspecionamos los paquetes observamos que la informacion esta encriptada



Sin embargo, si capturamos los paquetes de la práctica anterior podremos ver los datos como los operadores (imagen de la izquierda) o el resultado (imagen de la derecha)

Como podemos deducir esto es una ganancia en seguridad importante ya que con un simple programa como wireshark pueden robar la información fácilmente