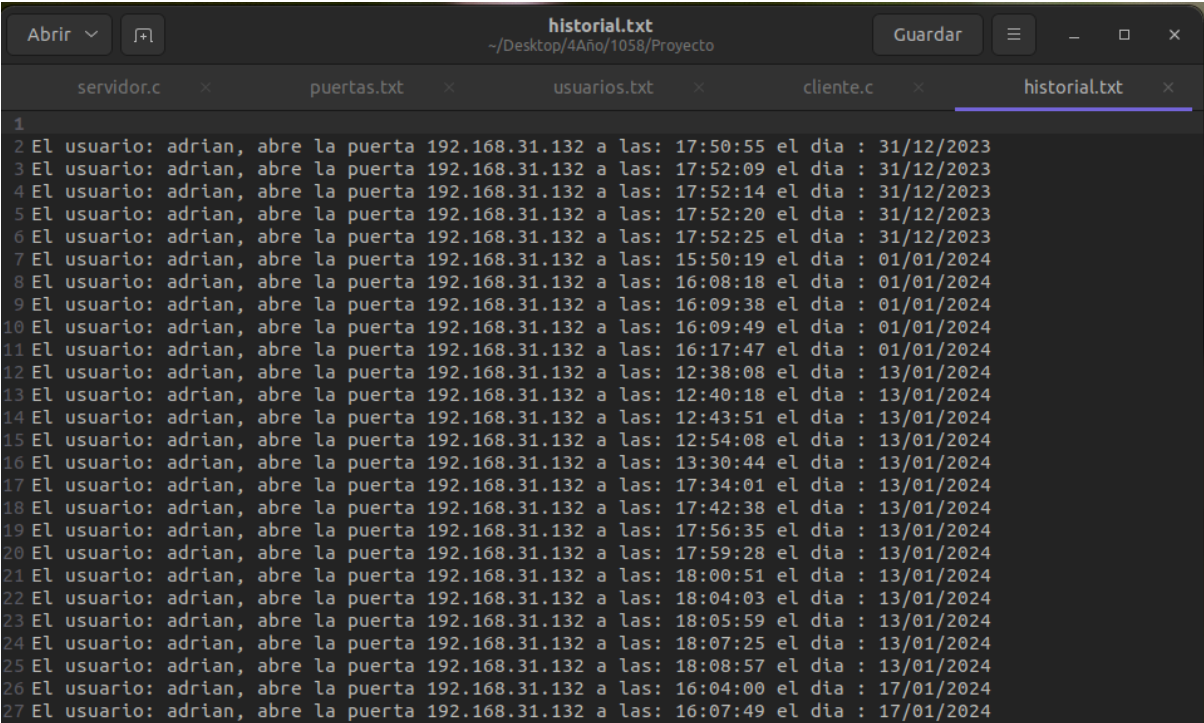


Soy Adrian Rubio Hernandez, esta es mi parte del proyecto que incluye la parte del servidor en linux con su respectivo código, la parte del ESP32 con su respectivo código y aparte he realizado un programa emulando la parte del cliente que se ejecuta desde el termina del ordenador con linux para poder comprobar el código del servidor y el código del ESP32. Faltará la parte de la aplicación móvil que la tiene que entregar mi compañero Sabino Corcella.

Primero explicaré los documentos que utiliza el servidor:

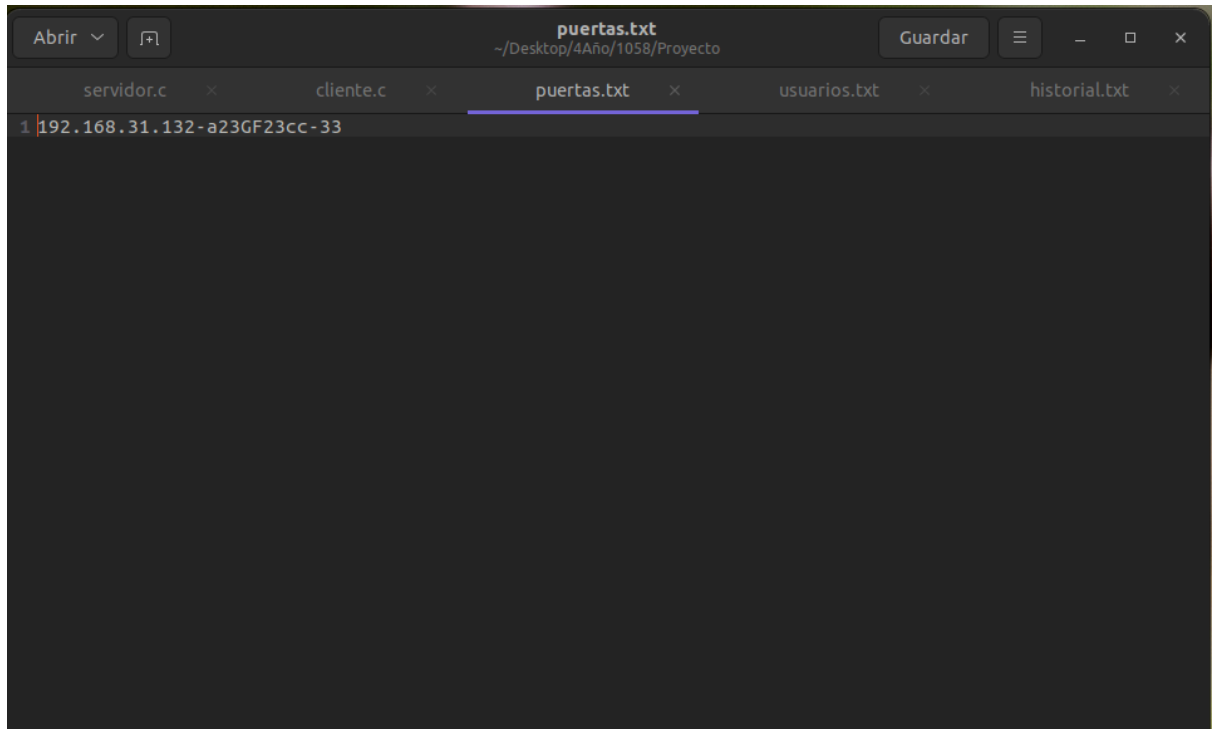
El servidor empezará comprobando si existen los correspondientes archivos txt:

- historial.txt: para guardar un historial de los usuario que abren la puerta y el contenido de este txt será en cada línea: "El usuario: x, abre la puerta y a las: w el dia: r" siendo x el nombre del usuario, y la ip de la puerta que abre, w la hora minutos y segundos , r el dia que se abre la puerta.



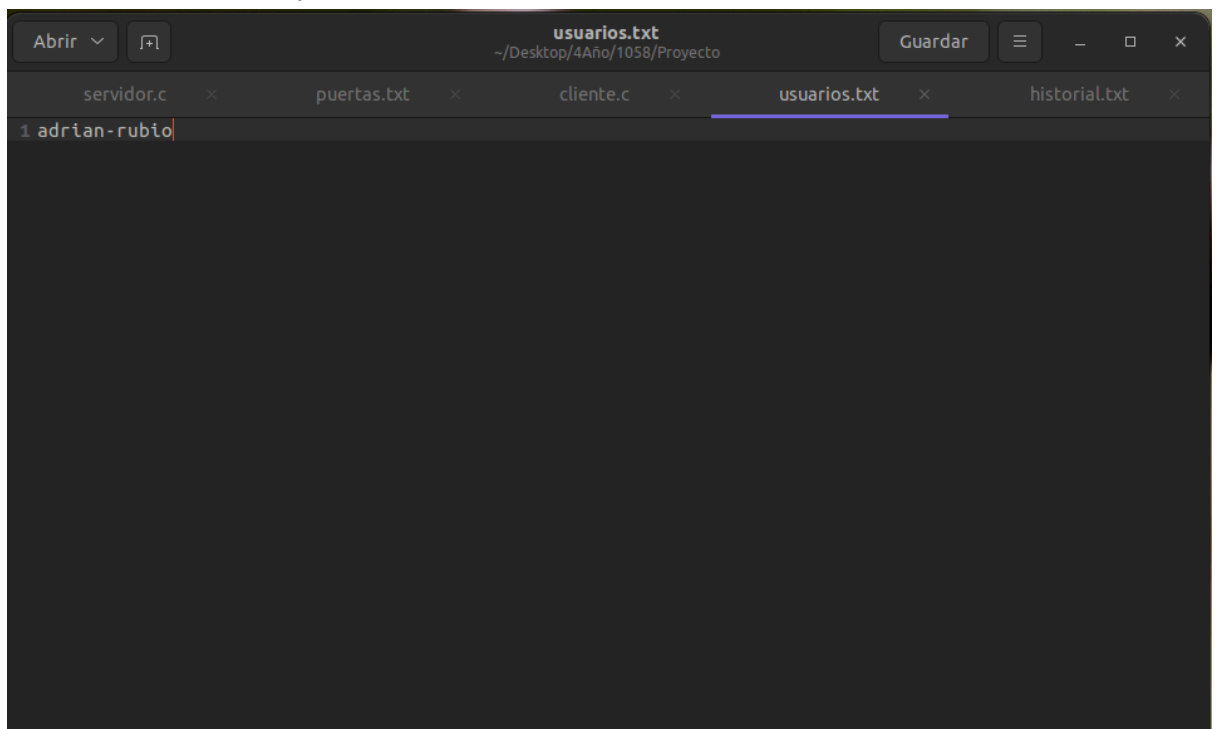
```
1
2 El usuario: adrian, abre la puerta 192.168.31.132 a las: 17:50:55 el dia : 31/12/2023
3 El usuario: adrian, abre la puerta 192.168.31.132 a las: 17:52:09 el dia : 31/12/2023
4 El usuario: adrian, abre la puerta 192.168.31.132 a las: 17:52:14 el dia : 31/12/2023
5 El usuario: adrian, abre la puerta 192.168.31.132 a las: 17:52:20 el dia : 31/12/2023
6 El usuario: adrian, abre la puerta 192.168.31.132 a las: 17:52:25 el dia : 31/12/2023
7 El usuario: adrian, abre la puerta 192.168.31.132 a las: 15:50:19 el dia : 01/01/2024
8 El usuario: adrian, abre la puerta 192.168.31.132 a las: 16:08:18 el dia : 01/01/2024
9 El usuario: adrian, abre la puerta 192.168.31.132 a las: 16:09:38 el dia : 01/01/2024
10 El usuario: adrian, abre la puerta 192.168.31.132 a las: 16:09:49 el dia : 01/01/2024
11 El usuario: adrian, abre la puerta 192.168.31.132 a las: 16:17:47 el dia : 01/01/2024
12 El usuario: adrian, abre la puerta 192.168.31.132 a las: 12:38:08 el dia : 13/01/2024
13 El usuario: adrian, abre la puerta 192.168.31.132 a las: 12:40:18 el dia : 13/01/2024
14 El usuario: adrian, abre la puerta 192.168.31.132 a las: 12:43:51 el dia : 13/01/2024
15 El usuario: adrian, abre la puerta 192.168.31.132 a las: 12:54:08 el dia : 13/01/2024
16 El usuario: adrian, abre la puerta 192.168.31.132 a las: 13:30:44 el dia : 13/01/2024
17 El usuario: adrian, abre la puerta 192.168.31.132 a las: 17:34:01 el dia : 13/01/2024
18 El usuario: adrian, abre la puerta 192.168.31.132 a las: 17:42:38 el dia : 13/01/2024
19 El usuario: adrian, abre la puerta 192.168.31.132 a las: 17:56:35 el dia : 13/01/2024
20 El usuario: adrian, abre la puerta 192.168.31.132 a las: 17:59:28 el dia : 13/01/2024
21 El usuario: adrian, abre la puerta 192.168.31.132 a las: 18:00:51 el dia : 13/01/2024
22 El usuario: adrian, abre la puerta 192.168.31.132 a las: 18:04:03 el dia : 13/01/2024
23 El usuario: adrian, abre la puerta 192.168.31.132 a las: 18:05:59 el dia : 13/01/2024
24 El usuario: adrian, abre la puerta 192.168.31.132 a las: 18:07:25 el dia : 13/01/2024
25 El usuario: adrian, abre la puerta 192.168.31.132 a las: 18:08:57 el dia : 13/01/2024
26 El usuario: adrian, abre la puerta 192.168.31.132 a las: 16:04:00 el dia : 17/01/2024
27 El usuario: adrian, abre la puerta 192.168.31.132 a las: 16:07:49 el dia : 17/01/2024
```

- **puertas.txt:** contendrá en cada línea “x-y-w” siendo x la ip de la puerta que puede abrir el servidor, w la contraseña que tiene que enviar el servidor a la puerta que es el ESP32 para que esta se abra y w es el número con el que se encripta esta contraseña con el metodo cesar que la esp32 también tendrá este número y descifra la contraseña.



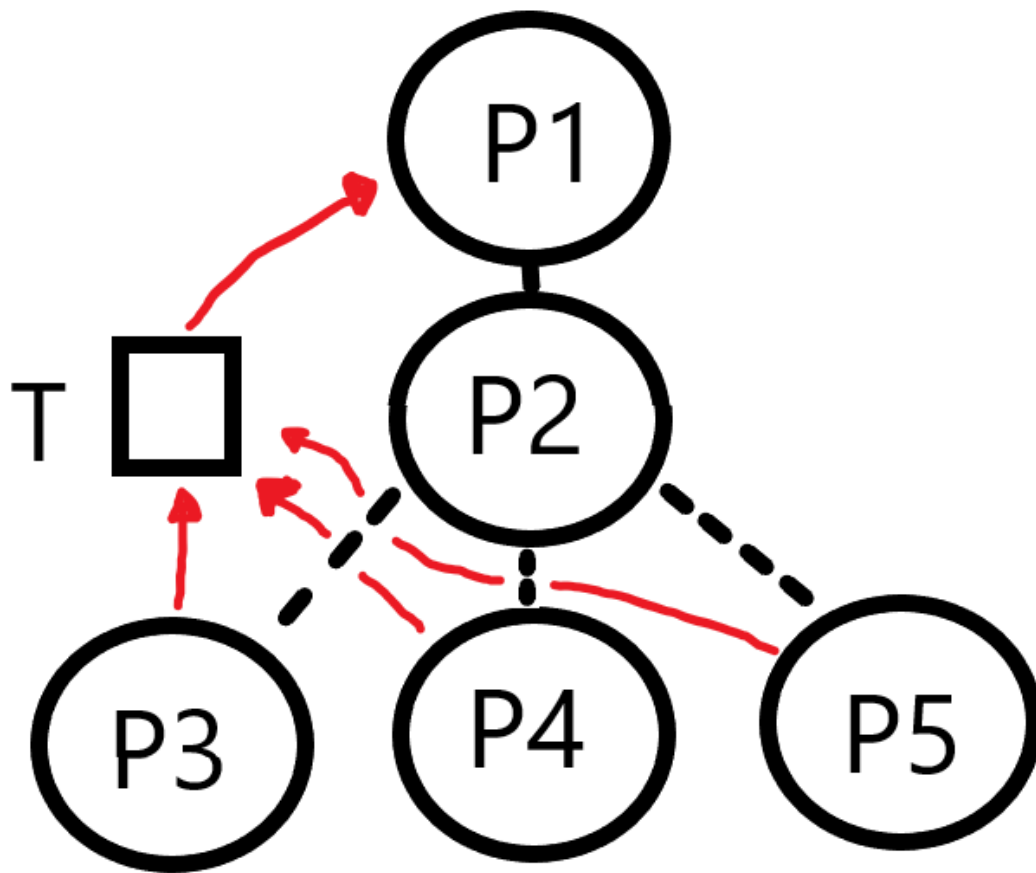
A screenshot of a text editor window titled "puertas.txt" with the path "~/Desktop/4Año/1058/Proyecto". The editor has tabs for "servidor.c", "cliente.c", "puertas.txt", "usuarios.txt", and "historial.txt". The "puertas.txt" tab is active and shows a single line of text: "1 192.168.31.132-a23GF23cc-33".

- **usuarios.txt:** cada línea contendrá un usuario y su contraseña “x,y” siendo “x” el nombre del usuario e “y” la contraseña asociada a este usuario.



A screenshot of a text editor window titled "usuarios.txt" with the path "~/Desktop/4Año/1058/Proyecto". The editor has tabs for "servidor.c", "puertas.txt", "cliente.c", "usuarios.txt", and "historial.txt". The "usuarios.txt" tab is active and shows a single line of text: "1 adrian-rubio".

Explicación del servidor a nivel de procesos:



El P1 proceso padre creara el servidor de escucha tcp del servidor y creará el hijo P2, una vez creado este hijo el proceso padre se quedara leyendo de la tubería para poder realizar la escritura en el archivo historial.txt.

El P2 este proceso se encargará de el socket tcp el cual escuchara y cuando un cliente abra la aplicación por defecto se conectará al servidor entonces este proceso detectará al cliente y creará un nuevo hijo haciendo que este nuevo proceso se encargue del cliente, una vez creado el nuevo proceso que puede ser P3,P4,P5 el proceso P2 volverá a ponerse a escuchar del socket tcp por sí un nuevo cliente se conecta.

El P3,P4,P5.... estos procesos se encargaran de la comunicación del cliente realizando que una vez que el cliente ponga los datos de usuario y la ip de la puerta que quiere abrir comprobará si el usuario y contraseña son correctos y después si la puerta con la IP existe, si hay algún fallo este proceso devolverá al cliente mediante el socket tcp un mensaje con el fallo si es de usuario y contraseña o de ip de la puerta. Una vez que ambas cosas son correctas enviará un mensaje al cliente y la aplicación podrá abrir la puerta o cerrar la sesión. Si el usuario clicka abrir la puerta el servidor buscará la contraseña de la puerta que tiene que enviar y tambien el numero para el cifrado César para cifrar la contraseña y enviarla al ESP32 mediante un socket UDP que también creará este proceso y si la esp32 devuelve un mensaje poniendo "okko" sera que todo es correcto y se abre la puerta lo que hara que estos procesos envíen la informacion oportuna como el usuario y la puerta para que el proceso P1 escriba en el txt historial lo correspondiente información para tener el historial.

Explicacion del código del servidor y lo que realiza:

El código incluye 7 funciones que explicare ahora:

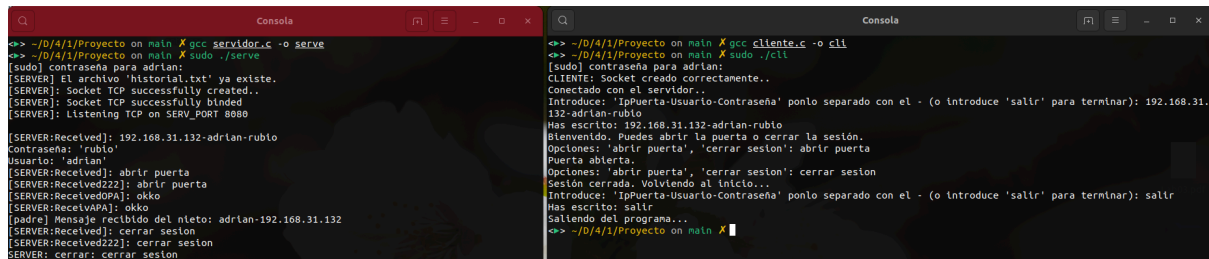
- `int crearsockTCP (int port, struct sockaddr_in servaddr){....}`
Esta función crea y devuelve un socket TCP el cual se usara para crear el socket del servidor, el servidor estará escuchando este socket y cuando algún cliente abra la app se creará un proceso hijo.
- `int crearSocketUDP(const char *serverIP, int port, struct sockaddr_in servidorAddr) {....}`
Esta función crea y devuelve un socket UDP el cual se usará para crear el socket del ESP 32, el servidor enviara la contraseña cifrada y escuchará la respuesta del ESP32.
- `int autenticarUsuario(const char *usuario, const char *contrasena) {.....}`
Esta función se encarga de comprobar si el nombre de usuario y contraseña que envia el usuario al servidor se encuentra en el fichero usuarios.txt.
- `int autenticarIP(const char *ip) {.....}`
Esta función se encarga de comprobar si la ip de la "puerta" esp32 que envia el usuario al servidor se encuentra en el fichero puertas.txt.
- `char* encrypt_cesar(const char *input, int key) {.....}`
Se encarga de cifrar la contraseña que se enevia al ESP32
- `char* decrypt_cesar(const char *input, int key) {.....}`
Se encarga de descifrar un char cifrado con cesar
- `char* contrasenaIP(const char *ip) {.....{`
Se encarga de encontrar la contraseña de la puerta ip que pongas y cifrar la contraseña con el numero cesar que tenga y devuelve la contraseña cifrada para enviarla

Con todo esto y Explicación del servidor a nivel de procesos se entiende como funciona el programa servidor.c

Explicare cliente.c para como compruebo que funcione el servidor y el ESP32

Una vez ejecutado se conectara al servidor con la ip que esté puesta en el programa y mostrara como introducir los datos que el cliente envía desde la aplicación el primer mensaje es 'IpPuerta-Usuario-Contraseña' si es correcto desde la app se podra clicar en los botones de cerrar sesión o abrir puerta, como se muestra primero al abrir puerta le envias al servidor ese comando y se comunicara con el esp32 y una vez abierta el cliente recibe la respuesta del servidor y como se ve se puede volver a abrir la puerta o cerrar la sesion, despues pongo cerrar sesión y el servidor recibe que se ha cerrado la sesion y el cliente ya no podra abrir la puerta y tendrá que volver a loguearse, por ultimo pongo salir pero es como si cerrases la aplicación sin mas.

Foto con el procedimiento SERVIDOR - CLIENTE:



```
<> ~/D/4/1/Proyecto on main X gcc servidor.c -o serve
<> ~/D/4/1/Proyecto on main X sudo ./serve
[sudo] contraseña para adrian:
[SERVER] El archivo 'historial.txt' ya existe.
[SERVER]: Socket TCP successfully created..
[SERVER]: Socket TCP successfully binded
[SERVER]: Listening TCP on SERV_PORT 8080

[SERVER:Received]: 192.168.31.132-adrian-rubio
Contraseña: 'rubio'
Usuario: 'adrian'
[SERVER:Received]: abrir puerta
[SERVER:Received22]: abrir puerta
[SERVER:ReceivedOPA]: okko
[SERVER:ReceivedAPA]: okko
[padre] Mensaje recibido del nroto: adrian-192.168.31.132
[SERVER:Received]: cerrar sesion
[SERVER:Received22]: cerrar sesion
SERVER: cerrar: cerrar sesion

<> ~/D/4/1/Proyecto on main X gcc cliente.c -o cli
<> ~/D/4/1/Proyecto on main X sudo ./cli
[sudo] contraseña para adrian:
CLIENTE: Socket creado correctamente..
Conectado con el servidor..
Introduce: 'IpPuerta-Usuario-Contraseña' ponlo separado con el - (o introduce 'salir' para terminar): 192.168.31.132-adrian-rubio
Has escrito: 192.168.31.132-adrian-rubio
Bienvenido. Puedes abrir la puerta o cerrar la sesión.
Opciones: 'abrir puerta', 'cerrar sesion': abrir puerta
Puerta abierta.
Opciones: 'abrir puerta', 'cerrar sesion': cerrar sesion
Sesión cerrada. Volviendo al inicio..
Introduce: 'IpPuerta-Usuario-Contraseña' ponlo separado con el - (o introduce 'salir' para terminar): salir
Has escrito: salir
Saliedo del programa...
<> ~/D/4/1/Proyecto on main X
```

Foto con el procedimiento ESP32 :

Esta foto corresponde a un cliente abriendo la puerta dos veces:



```
Output Serial Monitor x
Message (Enter to send message to 'ESP32 Dev Module' on 'COM4')

WiFi connected.
IP address:
192.168.31.132
Servidor UDP iniciado en el puerto 9009
IP del cliente: 192.168.31.185, Puerto del cliente: 53396
Contraseña recibida: h56NM56jj
Contras descifrada: a23GF23cc
Contraseña descifrada: a23GF23cc
La contraseña es validaIP del cliente: 192.168.31.185, Puerto del cliente: 42070
Contraseña recibida: h56NM56jj
Contras descifrada: a23GF23cc
Contraseña descifrada: a23GF23cc
La contraseña es valida
```

El código del ESP32:

El código primero se conecta a una red ya puesta en el mismo codigo, después de esto crea un puerto de escucha UDP y cuando desde el servidor le envía información lo gestiona descifrando la contraseña, abriendo la puerta y por último enviando el mensaje de respuesta al servidor.