# Manual para el usuario de la aplicación

# “SecureChat”

SecureChat

En este manual de usuario para la aplicación SecureChat, se darán a conocer todos los pasos que se deben de llevar a cabo para la correcta instalación de la aplicación, y así ésta misma pueda ser utilizada por el usuario para poder establecer una comunicación escrita a través de un canal seguro de información entre dos usuarios.

# Acerca de "SecureChat"

## *¿Qué es la aplicación?*

SecureChat, es una aplicación para poder establecer una comunicación entre dos personas conectadas en dispositivos diferentes, ésta aplicación será capaz de generar llaves de acceso cada que se solicite inicio de sesión por parte de un usuario, de ésta manera, la aplicación procederá a utilizar esas llaves generadas mediante un algoritmo previamente establecido, y las llevará a un sistema de cifrado y descifrado de mensajes que se quieran enviar por cada uno de los usuarios que esté utilizando esta aplicación.

## *¿Para qué utilizarla?*

SecureChat se utiliza para poder comunicar de manera escrita a dos usuarios que tengan ésta aplicación. El distintivo de esta nueva aplicación, es que SecureChat representa "Seguridad, Confianza" lo que hoy en día muchos de los usuarios de estas aplicaciones están buscando.

## *¿Qué nos facilita en estos momentos?*

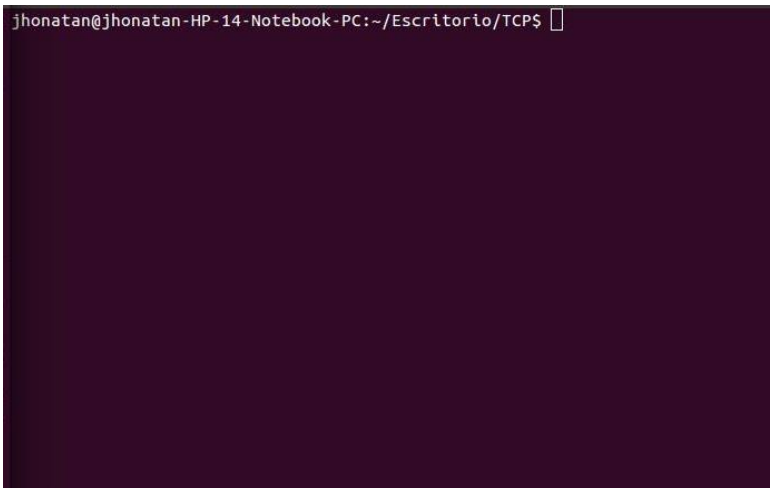
Nos facilita la comunicación escrita de manera instantánea.

# FUNCIONALIDAD

La aplicación SecureChat ofrece un chat entre dos personas que cuentan con dicha aplicación.

Para poder acceder a la aplicación se requiere una serie de pasos a seguir como se muestra a continuación:

**1.-** Abrimos la terminal de nuestro sistema operativo Ubuntu y accedemos a la carpeta donde se encuentran almacenados los archivos de la aplicación.




**2.-** Una vez que estamos dentro de esa carpeta, procedemos a compilar todos los archivos con extensión .c que se encuentran. Cabe señalar que es importante agregar la extensión –pthread a los archivos Cliente.c y server.c, ya que son los que nos permitirán establecer la conexión entre usuarios.

```
jhonatan@jhonatan-HP-14-Notebook-PC:~/Escritorio/TCP$ gcc -c cliente.c -pth
read
jhonatan@jhonatan-HP-14-Notebook-PC:~/Escritorio/TCP$ gcc -c DiffieC.c
jhonatan@jhonatan-HP-14-Notebook-PC:~/Escritorio/TCP$ gcc -c ElipticCurves.
c
jhonatan@jhonatan-HP-14-Notebook-PC:~/Escritorio/TCP$ gcc -c MatrixArray.c
jhonatan@jhonatan-HP-14-Notebook-PC:~/Escritorio/TCP$ gcc -c modularArit.c
modularArit.c: In function ‘saveArray’:
modularArit.c:89:8: warning: incompatible implicit declaration of built-in
function ‘strcat’ [enabled by default]
   file=strcat(file,".txt");
        ^
jhonatan@jhonatan-HP-14-Notebook-PC:~/Escritorio/TCP$ gcc -c server.c -pthr
ead
jhonatan@jhonatan-HP-14-Notebook-PC:~/Escritorio/TCP$ gcc -c socketHandle.c
jhonatan@jhonatan-HP-14-Notebook-PC:~/Escritorio/TCP$ gcc -c OFB.c
```

**3.-** Luego de compilar todos esos archivos, se generarán nuevos archivos en la carpeta con extensión .o, lo que haremos será correr esos archivos .o, estos archivos nos ayudaran a crear un cliente y un servidor, para poder comenzar la comunicación. Es importante que estos archivos se corran con la extensión ***cl*** para establecer el cliente, y ***ser*** para tener un servidor como se muestra a continuación:

```
jhonatan@jhonatan-HP-14-Notebook-PC:~/Escritorio/TCP$ gcc -o cl socketHandl
e.o DiffieC.o ElipticCurves.o MatrixArray.o modularArit.o OFB.o operaciones
GF.o saes.o cliente.c -pthread -lm
jhonatan@jhonatan-HP-14-Notebook-PC:~/Escritorio/TCP$ gcc -o ser socketHand
le.o DiffieC.o ElipticCurves.o MatrixArray.o modularArit.o OFB.o operacione
sGF.o saes.o server.c -pthread -lm
jhonatan@jhonatan-HP-14-Notebook-PC:~/Escritorio/TCP$
```

**4.-** Una vez que ya hicimos los pasos anteriores, procedemos a abrir una nueva terminal. En ella lo que haremos será montar el servidor escribiendo en la terminal ***./ser*** para poder comenzar la conexión entre el cliente

```
jhonatan@jhonatan-HP-14-Notebook-PC:~/Escritorio/TCP$ ./ser
Socket created...
Binding done...
G(77,14,)*93: (42,29)
Waiting for a connection...
```

**5.-** Regresamos a la terminal donde habíamos compilado los archivos .c de la carpeta, y lo que haremos será montar el cliente escribiendo en la terminal ***.cl 127.0.0.1*** como se muestra a continuación:

```
jhonatan@jhonatan-HP-14-Notebook-PC:~/Escritorio/TCP$ gcc -o cl socketHandl
e.o DiffieC.o ElipticCurves.o MatrixArray.o modularArit.o OFB.o operaciones
GF.o saes.o cliente.c -pthread -lm
jhonatan@jhonatan-HP-14-Notebook-PC:~/Escritorio/TCP$ gcc -o ser socketHand
le.o DiffieC.o ElipticCurves.o MatrixArray.o modularArit.o OFB.o operacione
sGF.o saes.o server.c -pthread -lm
jhonatan@jhonatan-HP-14-Notebook-PC:~/Escritorio/TCP$ ./cl 127.0.0.1
Socket created...
Connected to the server...
Punto en el infinito
A(42,29,)*(b=39): Cs(65,70)
```

Si después de montar el cliente aparece en nuestra terminal ***socket created... conected to the server...*** quiere decir ya tenemos establecida una conexión entre nuestras dos terminales y podremos comenzar con la comunicación entre ellos. Lo que aparece en la imagen como ***A(42,29)*** es un punto bajo la curva elíptica donde está trabajando el cifrado, ***(b=39)*** es la llave pública del servidor y ***Cs(65,70)*** es la clave de inicio de sesión que comparten tanto el cliente y el servidor, ésta clave es necesaria para compartir mensajes.

Como se muestra a continuación, ya se puede empezar a compartir mensajes entre el cliente y el servidor una vez que se tiene la conexión entre ellos.