



Comunicación teléfonos móviles a través de red
“Tor” y sincronización de archivos P2P con
“Syncthing”

Instalación, configuración & administración.

Manual usuario

versión 1.0 Beta

Agosto 2020.

OpenQbitP2P es una marca registrada de OpenQbit Inc, bajo licencia de uso libre y comercial. Términos y condiciones de uso en: www.OpenQbit.com

Contenido

1. Introducción.....	3
2. Instalación y configuración de red comunicaciones para teléfonos móviles.....	5
3. Configuración de almacenamiento “storage” dentro de Termux.....	8
4. Instalación de red “Tor” e instalación de “Syncthing”.....	9
5. Servidor SSH (Secure Shell).	10
6. Configuración de servidor SSH en teléfono móvil (smartphone).....	11
7. Configuración de red “Tor” con servicio de SSH (Secure Shell).	17
8. Configuración de sistema “Peer to Peer” con Syncthing de forma manual.....	20
9. Ambientes Blockly (App Inventor, AppyBuilder y Thunkable).	35
10. ¿Qué es un IdDevice o identificador de dispositivo en Syncthing?.....	36
11. Definición y uso de bloques en OpenQbitP2PwithSSH.	37
12. Extensión OpenQbitP2PwithSSH	38
13. Uso del REst API con API-Key en Syncthing.	52
14. Licenciamiento y uso de software.	55

1. Introducción.

Una red **peer-to-peer** o **P2P**, red de pares, red entre iguales o red entre pares (P2P, por sus siglas en inglés) es una red de ordenadores en la que todos o algunos aspectos funcionan sin clientes ni servidores fijos, sino una serie de nodos que se comportan como iguales entre sí. Es decir, actúan simultáneamente como clientes y servidores respecto a los demás nodos de la red.

Las redes P2P permiten el intercambio directo de información, en cualquier formato, entre los teléfonos móviles, tabletas, PCs y ordenadores interconectados.

Normalmente este tipo de redes se implementan como redes superpuestas construidas en la capa de aplicación de redes públicas como Internet.

El hecho de que sirvan para compartir e intercambiar información de forma directa entre dos o más usuarios ha propiciado que parte de los usuarios lo utilicen para intercambiar archivos cuyo contenido dan una mayor seguridad en información sensible o con derechos de autor.

Las redes peer-to-peer aprovechan, administran y optimizan el uso del ancho de banda de los demás usuarios de la red por medio de la conectividad entre los mismos, y obtienen así más rendimiento en las conexiones y transferencias que con algunos métodos centralizados convencionales, donde una cantidad relativamente pequeña de servidores provee el total del ancho de banda y recursos compartidos para un servicio o aplicación.

Dichas redes son útiles para diversos propósitos. A menudo se usan para compartir ficheros (archivos) de cualquier tipo (por ejemplo, audio, vídeo o software). Este tipo de red también suele usarse en telefonía VoIP para hacer más eficiente la transmisión de datos en tiempo real.

La eficacia de los nodos en el enlace y transmisión de datos puede variar según su configuración local (cortafuegos, NAT, ruteadores, etc.), velocidad de proceso, disponibilidad de ancho de banda de su conexión a la red y capacidad de almacenamiento en disco.

En nuestro caso estaremos usando las funcionalidades de dos desarrollos con madurez en su uso y características en comunicación, seguridad y diversidad para su uso.

La primer es la red “**Tor**”.- Tor (sigla de The Onion Router -en español- El Enrutador Cebolla) es un proyecto cuyo objetivo principal es el desarrollo de una red de comunicaciones distribuida de baja latencia y superpuesta sobre internet, en la que el encaminamiento de los mensajes intercambiados entre los usuarios no revela su identidad, es decir, su dirección IP (anonimato a nivel de red) y que, además, mantiene la integridad y el secreto de la información que viaja por ella.

Más referencia en: <https://www.torproject.org/>

Esta red “Tor” la usaremos para realizar cualquier tipo de comunicación entre teléfonos móviles ya sé que se encuentren en un ambiente Wifi o con un servicio de Internet móvil de cualquier compañía telefónica a nivel mundial. Más adelante la usaremos para crear un canal seguro a través de esta red con un servicio de SSH (Secure Shell) con esta combinación podremos enviar, distribuir, copiar, obtener cualquier tipo de datos.

Otra herramienta o servicio que estaremos para la sincronización y/o replicación de datos ser Syncthing.

Syncthing. - Es una aplicación de código abierto de sincronización de archivos punto a punto disponible para Windows, Mac, Linux, Android, Solaris, Darwin y BSD.

Puede sincronizar archivos entre dispositivos en una red local o entre dispositivos remotos a través de Internet. La seguridad de los datos está integrada en el diseño del software.

¿Cómo funciona?

El descubrimiento de dispositivos se logra a través de servidores de descubrimiento de acceso público alojados por los desarrolladores del proyecto, descubrimiento local (LAN) a través de mensajes de difusión, historial del dispositivo y direccionamiento / nombre de host estático. El proyecto también proporciona el programa Syncthing Discovery Server para alojar los propios servidores de descubrimiento, que se pueden usar junto o como reemplazo de los servidores públicos.

La red de servidores de retransmisión contribuidos por la comunidad permite que los dispositivos que están detrás de diferentes firewalls IPv4 NAT puedan comunicarse mediante la transmisión de datos cifrados a través de un tercero. La retransmisión realizada es de naturaleza similar al protocolo TURN, con el tráfico cifrado TLS de extremo a extremo entre dispositivos (por lo tanto, incluso el servidor de retransmisión no puede ver los datos, solo la secuencia cifrada). Los relés privados también se pueden configurar y configurar, con o sin relés públicos, si se desea. La sincronización cambiará automáticamente de retransmisión a conexiones directas de dispositivo a dispositivo si descubre que hay una conexión directa disponible.

La sincronización se puede usar sin conexión alguna con el proyecto o los servidores de la comunidad: las actualizaciones, los datos de uso opcionales, el descubrimiento y la retransmisión se pueden desactivar y / o configurar de forma independiente, por lo que la malla y su infraestructura se pueden ejecutar en un Sistema cerrado de privacidad o confidencialidad.

Más información puede consultar la referencia: <https://syncthing.net/>

Iniciemos...

2. Instalación y configuración de red comunicaciones para teléfonos móviles.

Necesitamos primero un ambiente Linux ya que todo sistema Android está basado en Linux por seguridad y flexibilidad en herramientas, usaremos la terminal “Termux” que contiene dicho ambiente en donde instalaremos la red de comunicaciones.

Termux es un emulador de Linux donde instalaremos los paquetes necesarios para crear nuestra red de comunicación entre nodos.

Una de las principales ventajas al usar Termux es que podrás instalar programas sin tener que “rootear” el móvil (Smartphone) esto asegura que por esta instalación no se pierde ningún tipo de garantía del fabricante.

Instalación de Termux.

Desde tu móvil ve a la aplicación icono de Google Play (play.google.com).



Busca por aplicación “Termux”, selecciónala e iniciar el proceso de instalación.



Inicio de la aplicación Termux.

Después de iniciar tendremos que ejecutar los siguientes dos comandos para realizar actualizaciones del emulador del sistema operativo Linux:

\$ apt update

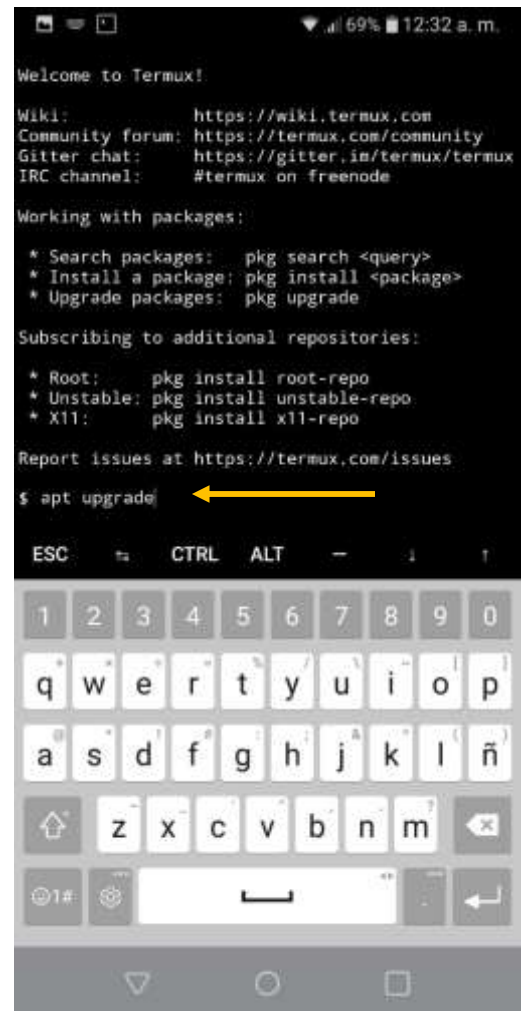
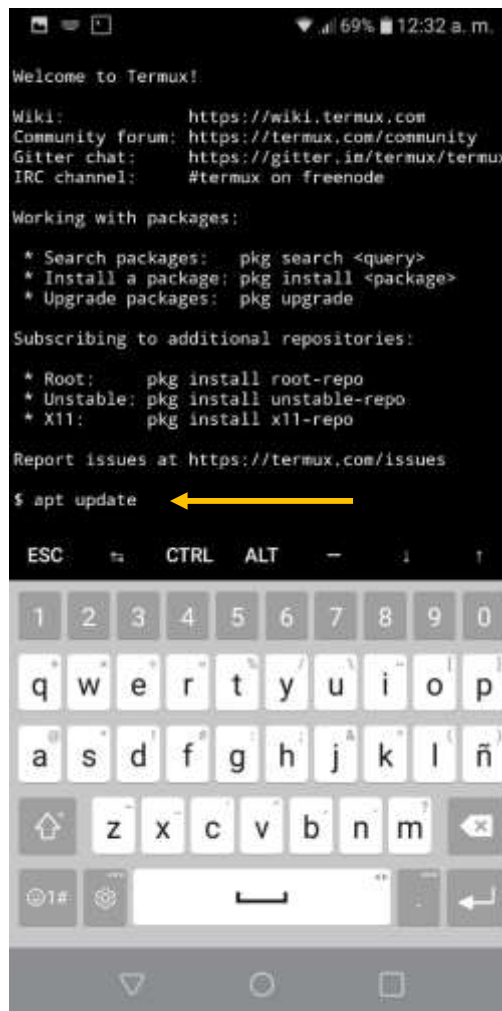
\$ apt upgrade

Confirmar todas las opciones Y(Yes)....

Inicio de Termux

\$ apt update

\$ apt upgrade



3. Configuración de almacenamiento “storage” dentro de Termux.

Después de haber realizado el update y upgrade del sistema Termux, empezaremos con la configuración de cómo poder ver el almacenamiento interno del teléfono en el sistema de Termux esto os ayudara a poder realizar intercambio de información entre Termux y nuestra información que tenemos en el teléfono.

Esto lo podemos realizar de una forma sencilla y rápida ejecutando el siguiente comando en una terminal de Termux.

\$ termux-setup-storage

Al ejecutar el anterior comando nos aparece una ventana pidiendo la confirmación de la creación de un **storage** virtual (directorio) en Termux damos click botón “PERMITIR” y creara el enlace al almacenamiento del teléfono. Verificamos dando el comando:

\$ ls



4. Instalación de red “Tor” e instalación de “Syncthing”.

\$ apt install tor

\$ apt install synching

Aceptar instalación tecleando Y mayúscula en ambos casos si lo pide...

\$ apt install tor



```
$ apt install tor
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libevent
The following NEW packages will be installed:
  libevent tor
0 upgraded, 2 newly installed, 0 to remove and 0
not upgraded.
Need to get 2319 kB of archives.
After this operation, 12.6 MB of additional disk
space will be used.
Do you want to continue? [Y/n] Y
```

\$ apt install synching



```
$ apt install synching
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  synching
0 upgraded, 1 newly installed, 0 to remove and 0
not upgraded.
Need to get 6407 kB of archives.
After this operation, 19.3 MB of additional disk
space will be used.
Get:1 https://dl.bintray.com/termux/termux-packa
ges-24 stable/main arm synching arm 1.5.0 [6407
kB]
27% [1 synching 2193 kB/6407 kB 34%]
```

5. Servidor SSH (Secure Shell).

```
$ apt install openssh
```

```
$ apt install sshpass
```

```
$ apt install openssh
```



```
$ apt install openssh
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  krb5 ldns libdb libedit termux-auth
The following NEW packages will be installed:
  krb5 ldns libdb libedit openssh termux-auth
0 upgraded, 6 newly installed, 0 to remove and 0
not upgraded.
Need to get 2255 kB of archives.
After this operation, 11.9 MB of additional disk
space will be used.
Do you want to continue? [Y/n] Y
Get:1 https://dl.bintray.com/termux/termux-packa
ges-24 stable/main arm libdb arm 18.1.32-4 [465
kB]
Get:2 https://dl.bintray.com/termux/termux-packa
ges-24 stable/main arm krb5 arm 1.18.1 [839 kB]
24% [2 krb5 131 kB/839 kB 16%]
```

```
$ apt install sshpass
```



```
$ apt install sshpass
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  sshpass
0 upgraded, 1 newly installed, 0 to remove and 0
not upgraded.
Need to get 7158 B of archives.
After this operation, 57.3 kB of additional disk
space will be used.
0% [Working]
```

Hemos terminado con la instalación de la red de comunicaciones, continuamos con la configuración de los paquetes: Tor y Syncthing.

6. Configuración de servidor SSH en teléfono móvil (smartphone).

Habilitaremos el servidor de SSH en el teléfono móvil para poder conectarnos desde nuestra PC al móvil y poder trabajar de una forma más rápida y cómoda, así mismo nos servirá para comprobar que el servicio del servidor SSH en el móvil funciona correctamente ya que este lo utilizaremos en la red de comunicación para usar el servicio P2P de la aplicación Syncthing.

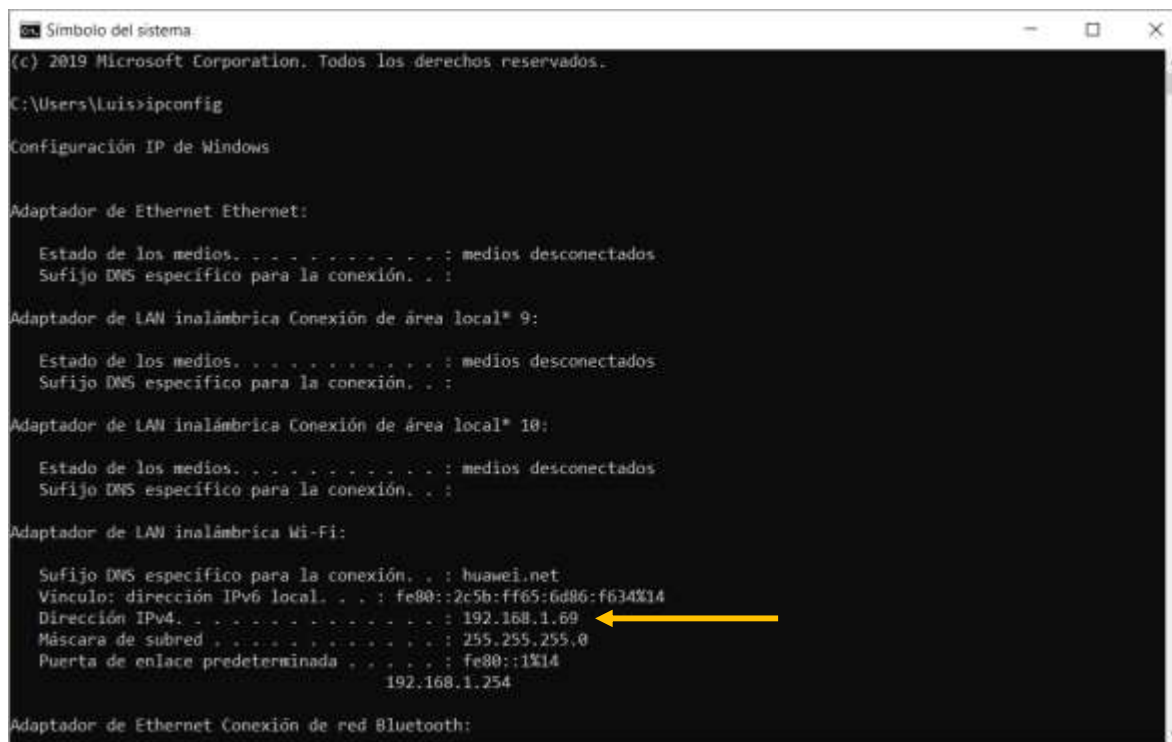
Lo primero que tenemos que hacer es conectar **a la misma red WiFi** el móvil y la PC para que se puedan ver. Las IPs o direcciones deben ser similares a 192.168.XXX.XXX los valores XXX son números variables que se asignan aleatoriamente en cada equipo.

Este ejemplo se probó en un móvil LG Q6 y una PC con Windows 10 Home.

Revisar la IP o dirección que tiene la PC conectada al WiFi deberemos abrir una terminal en Windows.

En el panel inferior donde está la lupa de buscar escribir cmd y presionar la tecla Enter. Se abrirá una terminal y en esta escribimos el comando:

```
C:\Users\nombre_usuario> ipconfig
```



```
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\Users\Luis>ipconfig

Configuración IP de Windows

Adaptador de Ethernet Ethernet:

    Estado de los medios. . . . . : medios desconectados
    Sufijo DNS específico para la conexión. . . :

Adaptador de LAN inalámbrica Conexión de área local* 9:

    Estado de los medios. . . . . : medios desconectados
    Sufijo DNS específico para la conexión. . . :

Adaptador de LAN inalámbrica Conexión de área local* 10:

    Estado de los medios. . . . . : medios desconectados
    Sufijo DNS específico para la conexión. . . :

Adaptador de LAN inalámbrica Wi-Fi:

    Sufijo DNS específico para la conexión. . . : huawei.net
    Vínculo: dirección IPv6 local. . . : fe80::2c5b:ff65:6d86:f634%14
    Dirección IPv4. . . . . : 192.168.1.69
    Máscara de subred. . . . . : 255.255.255.0
    Puerta de enlace predeterminada. . . : fe80::1%14
                                     192.168.1.254

Adaptador de Ethernet Conexión de red Bluetooth:
```

Nos mostrará la IP que tiene asignada la PC en este es la 192.168.1.69 sin embargo esta los más probable es que sea diferente en cada caso.

NOTA: debe tomarse la dirección donde dice “Dirección IPv4” no confundir con la Puerta de enlace.

Ahora en el caso del teléfono móvil en la terminal de Termux debemos teclear el siguiente comando para saber cómo se llama nuestro usuario que usaremos para conectarnos al servidor SSH que tiene nuestro teléfono, ejecutamos el siguiente comando:

\$ whoami

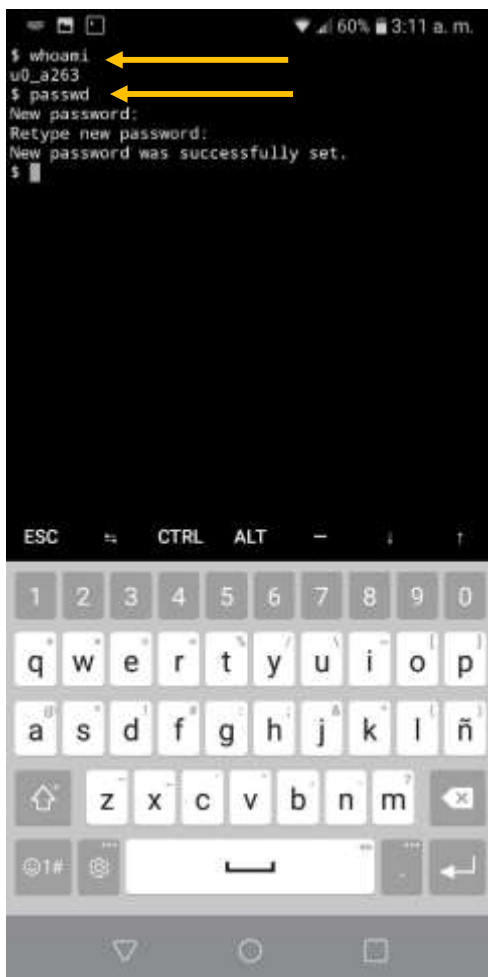
Posteriormente debemos darle un password a este usuario por lo que tenemos que ejecutar el siguiente comando:

\$ passwd

Nos pedirá que tecleemos un password y damos Enter, nuevamente nos pide el password confirmamos le damos el mismo y damos Enter, si ha sido exitosamente **“New password was successfully set”** en caso de marcar un error es posible que el password no se haya tecleado correctamente. Volver a realizar el procedimiento.

Y después para saber que IP tenemos en Termux tecleamos el siguiente comando, la IP esta después de la palabra **“inet”**:

\$ ifconfig -a



The screenshot shows a Termux terminal window with a black background and white text. The status bar at the top indicates 60% battery and 3:11 a.m. The terminal shows the following commands and output:

```
$ whoami
u0_a263
$ passwd
New password:
Retype new password:
New password was successfully set.
$
```

Two yellow arrows point to the 'whoami' and 'passwd' commands. Below the terminal is a virtual keyboard with a QWERTY layout and a numeric keypad.



The screenshot shows a Termux terminal window with a black background and white text. The status bar at the top indicates 61% battery and 2:57 a.m. The terminal shows the output of the 'ifconfig -a' command:

```
e 0
TX packets 0 bytes 0 (0.0 B)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>
> mtu 1500
    inet 192.168.1.68 netmask 255.255.255.0
    broadcast 192.168.1.255
    inet6 fe80::257:c1ff:fee6:3051 prefixlen 64 scopeid 0x20<link>
    unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 1000 (UNSPEC)
    RX packets 908745 bytes 947916536 (904.0 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 601034 bytes 93496881 (89.1 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

$
```

A yellow arrow points to the 'inet 192.168.1.68' line. Below the terminal is a virtual keyboard with a QWERTY layout and a numeric keypad.

Ahora es momento de iniciar el servicio del servidor de SSH en el teléfono para que pueda recibir sesiones desde la PC. Ejecutamos el siguiente comando en la terminal de Termux, este comando no arroja ningún resultado.

\$ sshd



Ahora tendremos que instalar un programa en la PC que se comunicara con el servidor SSH del teléfono desde la PC.

Tenemos que ir a la página <https://www.putty.org>

Seleccionar donde se encuentra el enlace “You can download PuTTY here”



Bitvise SSH Client

Bitvise SSH Client is an SSH and SFTP client for Windows. It is developed and supported prof supports all features supported by PuTTY, as well as the following:

- graphical SFTP file transfer;
- single-click Remote Desktop tunneling;
- auto-reconnecting capability;
- dynamic port forwarding through an integrated proxy;
- an FTP-to-SFTP protocol bridge.

Bitvise SSH Client is **free to use**. You can [download it here](#).

Escoger la versión de 32 bit, no importa si tu sistema es de 64 bits funcionara bien.

Download PuTTY: latest release

[Home](#) | [FAQ](#) | [Feedback](#) | [Licence](#) | [Updates](#) | [Mirrors](#)
Download: [Stable](#) | [Snapshot](#) | [Docs](#) | [Contact](#)

This page contains download links for the latest released version of PuTTY. Currently this is 0.73, released on 2019-09-29.

When new releases come out, this page will update to contain the latest, so this is a good page to bookmark or link to. Alternative

Release versions of PuTTY are versions we think are reasonably likely to work well. However, they are often not the most up-to-date. You can also download the [development snapshots](#), to see if the problem has already been fixed in those versions.

Package files

You probably want one of these. They include versions of all the PuTTY utilities.

(Not sure whether you want the 32-bit or the 64-bit version? Read the [FAQ entry](#).)

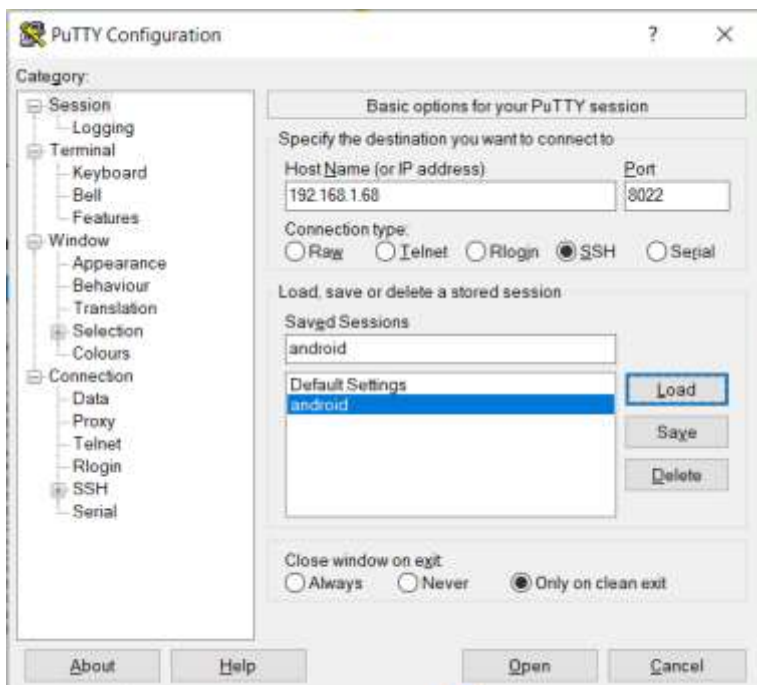
MSI ('Windows Installer')

32-bit:	putty-0.73-installer.msi	(or by FTP)	(signature)
64-bit:	putty-64bit-0.73-installer.msi	(or by FTP)	(signature)

Unix source archive

.tar.gz:	putty-0.73.tar.gz	(or by FTP)	(signature)
----------	-----------------------------------	-----------------------------	-----------------------------

Ya que se haya bajado en tu PC ejecútalo e instalado con las opciones por default. Después inicia la aplicación de PuTTY.



En esta sesión introduciremos los datos de nuestro servidor Openssh que instalamos en el teléfono móvil.

Introducir la IP del teléfono móvil.

HostName or IP address:

192.168.1.68 (ejemplo IP)

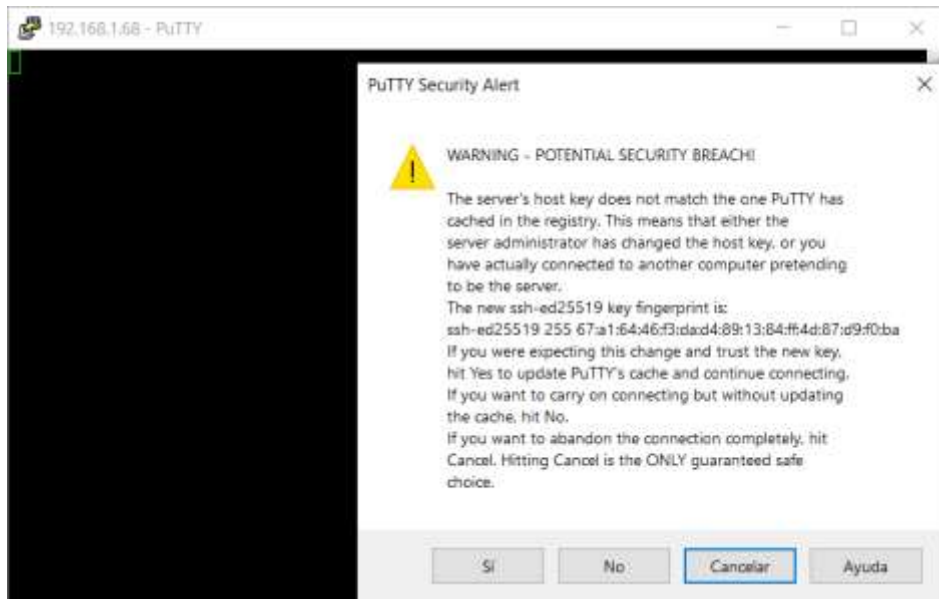
Port:

8022 (Puerto por default del servidor SSH del móvil).

Podemos dar un nombre de la sesión en "Saved Sessions" y damos click en el botón Save.

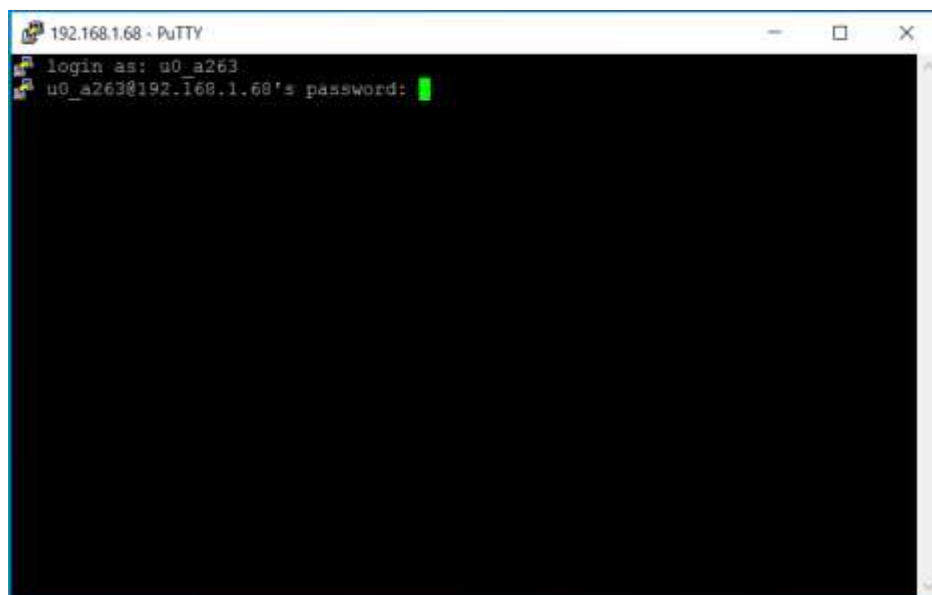
Posteriormente en la parte inferior presionamos para abrir una conexión al servidor dando el botón “Open”.

En la PC al conectarse por primera vez nos **pedirá por única vez** que confirmen la llave de cifrado de información le damos confirmación en el botón de “Si”.

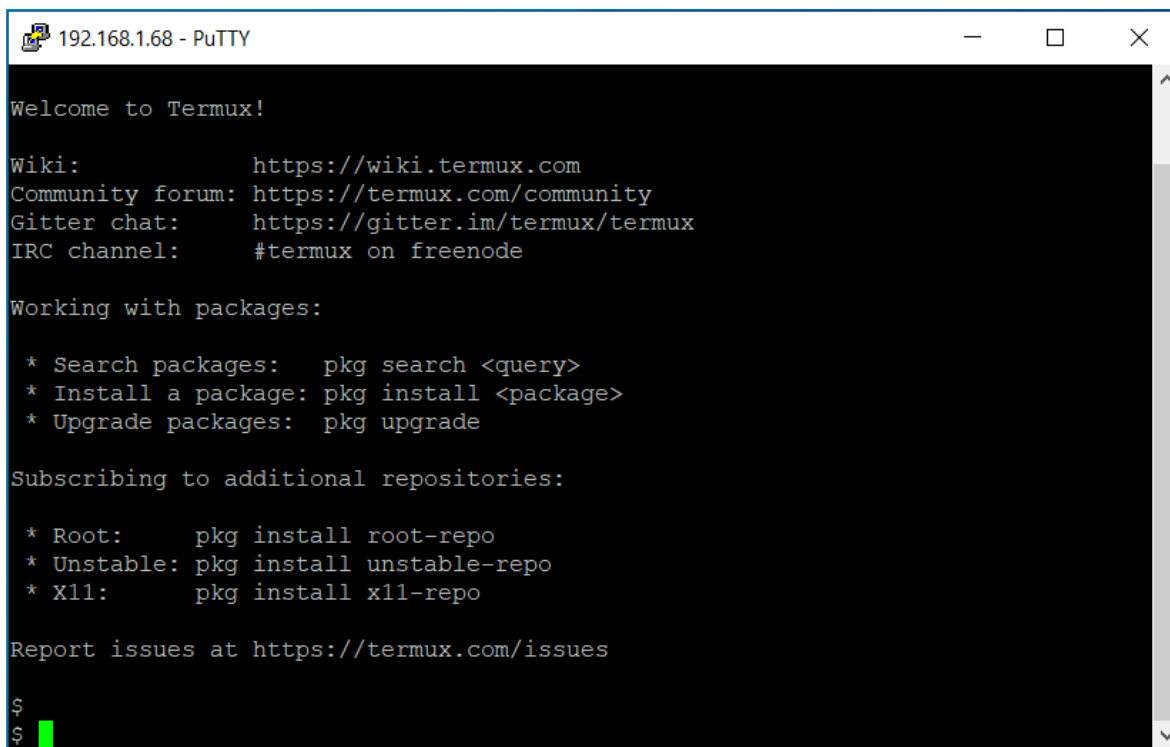


Posteriormente nos pedirá el usuario con el que nos vamos a conectar. Utilizaremos la información que sacamos con anterioridad (usuario y password).

En el **Login as:** debemos introducir nuestro usuario y dar Enter, después nos pedirá el password le damos nuevamente el botón de Enter.



Si los datos fueron los correctos, estaremos en una sesión de SSH (Secure Shell) realizada desde la PC (Cliente) en el teléfono (Servidor SSH).



```
192.168.1.68 - PuTTY
Welcome to Termux!

Wiki:          https://wiki.termux.com
Community forum: https://termux.com/community
Gitter chat:   https://gitter.im/termux/termux
IRC channel:   #termux on freenode

Working with packages:

* Search packages:  pkg search <query>
* Install a package: pkg install <package>
* Upgrade packages: pkg upgrade

Subscribing to additional repositories:

* Root:    pkg install root-repo
* Unstable: pkg install unstable-repo
* X11:     pkg install x11-repo

Report issues at https://termux.com/issues

$
$
```

NOTA IMPORTANTE: Recordemos que la IP (dirección) de la PC y la IP (dirección) teléfono móvil conectados en la misma WiFi estarán cambiando probablemente cada vez que nos desconectemos y volvamos a conectar por lo que hay que volver a verificar que direcciones tienen cada dispositivo, esto nos asegura el éxito de la conexión entre dispositivos por medio del servidor de SSH del teléfono y PC (Cliente).

Hasta este momento hemos podido realizar la conexión únicamente en la misma red WiFi, sin embargo, si nos movemos con el teléfono fuera de la misma red donde está la PC no podremos tener conexión ya que son diferentes redes lo que involucra pasar por otros dispositivos de comunicación más complicados, esto lo resolveremos cuando configuremos la red “Tor”.

Recordemos que esta conexión la realizamos únicamente para verificar el servicio del servidor que instalamos en el teléfono y para tener un ambiente más cómodo de trabajo con una sesión remota de a PC al teléfono.

7. Configuración de red “Tor” con servicio de SSH (Secure Shell).

Con la sesión remota desde la PC empezaremos a configurar la red “Tor” con el servicio de SSH habilitado.

La importancia de tener la red “Tor” es darles la propiedad a los dispositivos de poder comunicarse en cualquier parte del mundo a través de internet sin estar en la misma red WiFi, no importa donde estemos podremos conectarnos y formar la red entre nodos.

La red “Tor” tiene mucha flexibilidad en su configuración ya que cuenta con varios parámetros en su archivo de configuración “torrc” este archivo se encuentra en la ruta o path (\$PREFIX/etc/tor/torrc) en nuestro caso con la sesión de Termux desde nuestra PC lo sabremos con escribir el siguiente comando:

```
$ echo $PREFIX
```

```
/data/data/com.termux/files/usr
```

Por lo anterior el archivo que necesitamos editar estará en la ruta o path:

PREFIX/etc/tor/torrc que es igual a **/data/data/com.termux/files/usr/etc/tor/torrc**

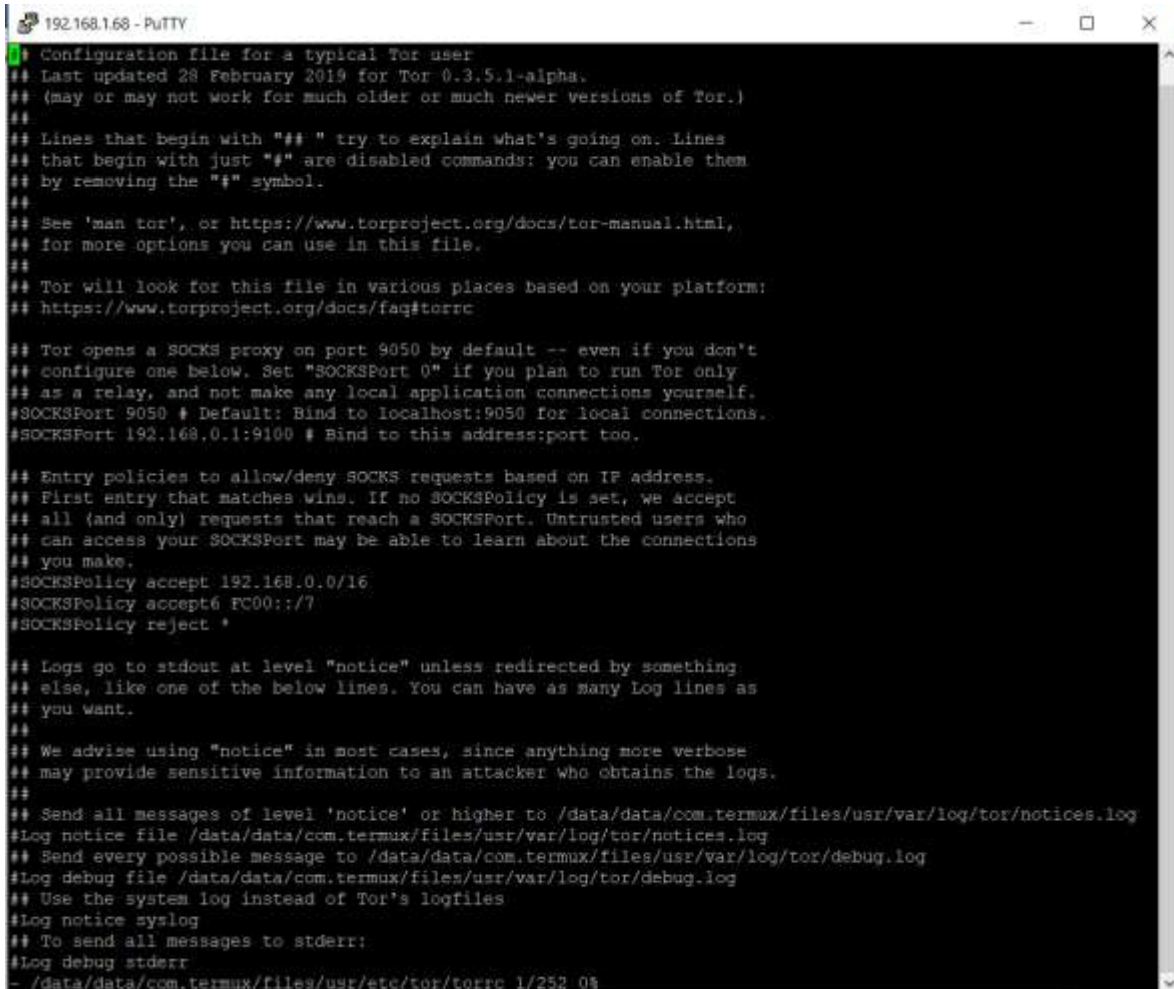
Procedemos a editar el archivo de configuración usando el editor en línea de comando “vi” ejecutando el siguiente comando:

```
$ vi /data/data/com.termux/files/usr/etc/tor/torrc
```

Nos editara dicho archivo, como ejemplo:



Archivo “torrc” editado:



```

Configuration file for a typical Tor user
## Last updated 28 February 2019 for Tor 0.3.5.1-alpha.
## (may or may not work for much older or much newer versions of Tor.)
##
## Lines that begin with "## " try to explain what's going on. Lines
## that begin with just "##" are disabled commands: you can enable them
## by removing the "##" symbol.
##
## See 'man tor', or https://www.torproject.org/docs/tor-manual.html,
## for more options you can use in this file.
##
## Tor will look for this file in various places based on your platform:
## https://www.torproject.org/docs/faq#torrc
##
## Tor opens a SOCKS proxy on port 9050 by default -- even if you don't
## configure one below. Set "SOCKSPort 0" if you plan to run Tor only
## as a relay, and not make any local application connections yourself.
SOCKSPort 9050 # Default: Bind to localhost:9050 for local connections.
SOCKSPort 192.168.0.1:9100 # Bind to this address:port too.
##
## Entry policies to allow/deny SOCKS requests based on IP address.
## First entry that matches wins. If no SOCKSPolicy is set, we accept
## all (and only) requests that reach a SOCKSPort. Untrusted users who
## can access your SOCKSPort may be able to learn about the connections
## you make.
SOCKSPolicy accept 192.168.0.0/16
SOCKSPolicy accept6 FC00::/7
SOCKSPolicy reject *
##
## Logs go to stdout at level "notice" unless redirected by something
## else, like one of the below lines. You can have as many Log lines as
## you want.
##
## We advise using "notice" in most cases, since anything more verbose
## may provide sensitive information to an attacker who obtains the logs.
##
## Send all messages of level 'notice' or higher to /data/data/com.termux/files/usr/var/log/tor/notices.log
Log notice file /data/data/com.termux/files/usr/var/log/tor/notices.log
## Send every possible message to /data/data/com.termux/files/usr/var/log/tor/debug.log
Log debug file /data/data/com.termux/files/usr/var/log/tor/debug.log
## Use the system log instead of Tor's logfiles
Log notice syslog
## To send all messages to stderr:
Log debug stderr
- /data/data/com.termux/files/usr/etc/tor/torrc 1/252 0%

```

En este archivo “torrc” tendremos que agregar o usamos las líneas que tiene el archivo realizando los cambios siguientes, tres líneas que son las siguientes:

Sintaxis: SOCKSPort <número de puerto de la aplicación>

Ejemplo: SOCKSPort 9050

La variable **SOCKSPort** nos indica que este socket de comunicación sobre el protocolo TCP-IP usará por default el dispositivo móvil (teléfono) y será abierto o en uso el puerto 9050.

Sintaxis: HiddenServiceDir <Directorio donde se guardará configuración de la aplicación>

Ejemplo: HiddenServiceDir /data/data/com.termux/files/

La variable **HiddenServiceDir** nos indica que será el directorio en donde se almacenará la configuración del servicio que será usado a través de la red Tor, en este directorio se encuentran archivo de configuración y seguridad del servicio y en este directorio se encuentra un archivo llamado **hostname**, en este se encuentra la dirección para el dispositivo móvil que servirá para ser localizado en internet o en una red Wifi.

Podemos ver la dirección asignada por la red Tor para el servicio específico creado en nuestro caso está creando un servicio SSH que estará implementado sobre la red Tor, el directorio que estamos nombrando como **hidden_ssh** contendrá el archivo **hostname**. Este directorio no es necesario crearlo ya que cuando iniciemos el servicio de la red Tor será creado automáticamente, esto pasará para cada servicio que demos de alta y tendremos que asigna un directorio diferente para cada servicio.

Para ver la dirección proporcionada por la red Tor podemos usar el comando “more”, antes de usar este comando, debemos terminar la configuración del archivo “torrc” y dar de alta el servicio de la red Tor para que sea creado el directorio **hidden_ssh** y los archivos dentro de este como es el caso del archivo **hostname**.

\$ more ~/.tor/hidden_ssh/hostname (ver que el directorio inicial tor es oculto y debe usarse un punto antes del nombre).

El anterior comando nos entregara como resultado una dirección con una cadena de caracteres alfanuméricos con una extensión **.onion** similar a:

uqwthf6ojdmoybyzvudoq3x2weq7k7rjitblhba3pjbawk2nvjmx3wer.onion

Por ultimo debemos agregar al archivo de configuración “torrc” la variable **HiddenServicePort** este parámetro indica a la red Tor cual puerto usara la aplicación que estamos dando de alta y que usaremos a través de la red Tor.

Sintaxis: **HiddenServicePort <Puerto de salida> <IP local o IP publica>: <Puerto de salida>**

O

HiddenServicePort <Puerto de salida>

Ejemplos:

HiddenServicePort 22 127.0.0.1:8022

O

HiddenServicePort 8022

Con lo anterior hemos terminado la configuración de la red Tor para servicios genéricos y servicio SSH (Secure Shell) este servicio se utilizará para la comunicación y envío de cualquier tipo de datos o archivos como pudiera ser la actualización de una base de datos como SQLite o Mysql o Progress u otra instalada en nuestro teléfono móvil.

Continuamos con la configuración de la red de comunicaciones P2P para sincronización y replica de datos con Syncthing.

8. Configuración de sistema “Peer to Peer” con Syncthing de forma manual.

Una arquitectura “Peer to Peer” o P2P es fundamental en un sistema de tecnológica de punto a punto ya que esta arquitectura da dos puntos medulares en los procesos de comunicación del sistema, uno es proporcionar la misma información actualizada (sincronizada) en cualquier momento en todos los nodos no importando si los nodos están en una red privada (Wifi) o una red pública (internet) o un híbrido de estas dos y un segundo punto de este tipo de arquitectura es que no dependen de un intermediario (servidor) para transferir, actualizar o consultar información entre nodos. Todo lo anterior se debe a que la comunicación se realiza de forma directa entre los nodos, en nuestro caso la red P2P realiza directamente entre los teléfonos que forman la red sin un intermediario.

Para esta tarea usaremos la herramienta opensource Syncthing que funciona basada en una arquitectura “Peer to Peer”.

La configuración de Syncthing para dispositivos (teléfonos móviles) la veremos de forma manual y forma automática. La forma manual se aplica en sincronización con máximo 5 nodos los nodos pueden ser teléfonos móviles, PCs, servidores o tabletas, en caso de tener un numero de gran volumen lo óptimo será la configuración automática, el proceso se enfoca en el registro de los nodos con los que deseamos realizar la sincronización de la información seleccionada.

Los requisitos para empezar son:

1. Tener iniciado el servicio de la red Tor (opcional).
2. (opcional – recomendable) Tener instalado una aplicación que pueda leer códigos QR, sugerimos la aplicación para Android de App inventor que es fácil y simple de instalar desde Google Play.
3. Tener iniciado el servicio de Synthing.



Mostramos la aplicación de App Inventor que usaremos para la lectura de códigos QR que nos servirá en el futuro para el registro de los nodos en Syncthing.

El siguiente ejemplo fue realizado entre dos dispositivos móviles (teléfonos) usando los siguientes modelos:

Dispositivo móvil #1: Modelo LG Q6

Dispositivo móvil #2: Modelo Sansung

Empezaremos con iniciar los servicios de la red Tor y los servicios de Syncthing usando los siguientes comandos, abrimos dos terminales dentro de Termux y ejecutamos por separado cada comando:

\$ tor

Después de haber tecleado el comando de inicio del programa de la red Tor, debemos revisar que se haya realizado la ejecución correctamente, esto lo hacemos revisando que fue realizado en su totalidad al 100% nos marcara como porcentaje.

Ejecución del programa Tor en una terminal de Termux, verificamos que este ejecutando al 100%.

\$ tor



```
$ tor
May 23 23:00:30.932 [notice] Tor 0.4.3.5 running
on Linux with Libevent 2.1.11-stable, OpenSSL 1
.1.1g, Zlib 1.2.11, Liblzma 5.2.5, and Libzstd N
/A.
May 23 23:00:30.934 [notice] Tor can't help you
if you use it wrong! Learn how to be safe at htt
ps://www.torproject.org/download/download#warnin
g
May 23 23:00:30.939 [notice] Read configuration
file "/data/data/com.termux/files/usr/etc/tor/to
rrc".
May 23 23:00:30.970 [notice] I think we have 8 C
PUS, but only 6 of them are available. Telling T
or to only use 6. You can override this with the
NumCPUs option
May 23 23:00:30.973 [notice] Opening Socks liste
ner on 127.0.0.1:9050
May 23 23:00:30.974 [notice] Opened Socks listen
er on 127.0.0.1:9050
May 23 23:00:30.000 [notice] Parsing GEOIP IPv4
file /data/data/com.termux/files/usr/share/tor/g
eoip.
May 23 23:00:32.000 [notice] Parsing GEOIP IPv6
file /data/data/com.termux/files/usr/share/tor/g
eoip6.
May 23 23:00:34.000 [notice] Bootstrapped 0% (st
arting): Starting
May 23 23:00:34.000 [notice] Starting with guard
context "default"
May 23 23:00:35.000 [notice] Bootstrapped 5% (co
nn): Connecting to a relay
May 23 23:00:36.000 [notice] Bootstrapped 10% (c
onn_done): Connected to a relay
May 23 23:00:36.000 [notice] Bootstrapped 14% (h
andshake): Handshaking with a relay
May 23 23:00:36.000 [notice] Bootstrapped 15% (h
andshake_done): Handshake with a relay done
May 23 23:00:36.000 [notice] Bootstrapped 20% (o
nehop_create): Establishing an encrypted directo
ry connection
May 23 23:00:36.000 [notice] Bootstrapped 25% (r
```



```
ps://www.torproject.org/download/download#warnin
g
May 24 01:33:32.982 [notice] Read configuration
file "/data/data/com.termux/files/usr/etc/tor/to
rrc".
May 24 01:33:33.007 [notice] I think we have 8 C
PUS, but only 6 of them are available. Telling T
or to only use 6. You can override this with the
NumCPUs option
May 24 01:33:33.010 [notice] Opening Socks liste
ner on 127.0.0.1:9050
May 24 01:33:33.010 [notice] Opened Socks listen
er on 127.0.0.1:9050
May 24 01:33:33.000 [notice] Parsing GEOIP IPv4
file /data/data/com.termux/files/usr/share/tor/g
eoip.
May 24 01:33:34.000 [notice] Parsing GEOIP IPv6
file /data/data/com.termux/files/usr/share/tor/g
eoip6.
May 24 01:33:35.000 [notice] Bootstrapped 0% (st
arting): Starting
May 24 01:33:37.000 [notice] Starting with guard
context "default"
May 24 01:33:38.000 [notice] Bootstrapped 5% (co
nn): Connecting to a relay
May 24 01:33:38.000 [notice] Bootstrapped 10% (c
onn_done): Connected to a relay
May 24 01:33:39.000 [notice] Bootstrapped 14% (h
andshake): Handshaking with a relay
May 24 01:33:39.000 [notice] Bootstrapped 15% (h
andshake_done): Handshake with a relay done
May 24 01:33:39.000 [notice] Bootstrapped 75% (e
nough_dirinfo): Loaded enough directory info to
build circuits
May 24 01:33:39.000 [notice] Bootstrapped 90% (a
p_handshake_done): Handshake finished with a rel
ay to build circuits
May 24 01:33:39.000 [notice] Bootstrapped 95% (c
ircuit_create): Establishing a Tor circuit
May 24 01:33:42.000 [notice] Bootstrapped 100% (
done): Done
```

Posteriormente ejecutamos el comando de syncthing:

\$ syncthing

Después de ejecutarse dicho comando nos abrirá una página de administración en el browser de nuestro teléfono en su caso si no se abre automáticamente, podemos nosotros ir a cualquier browser (navegador) que tengamos instalado donde normalmente navegamos en internet y podemos poner la siguiente:

<http://127.0.0.1:8384>

Nos abrirá la pantalla de administración de la herramienta que nos ayudará a sincronizar nuestra información entre todos los nodos (teléfonos) del sistema.



Ya que tenemos abierta la pantalla de administración de “syncthing” procedemos a dar de alta el nodo o nodos que deseamos sincronizar la información. En este punto es donde ocuparemos el programa que lee códigos QR.

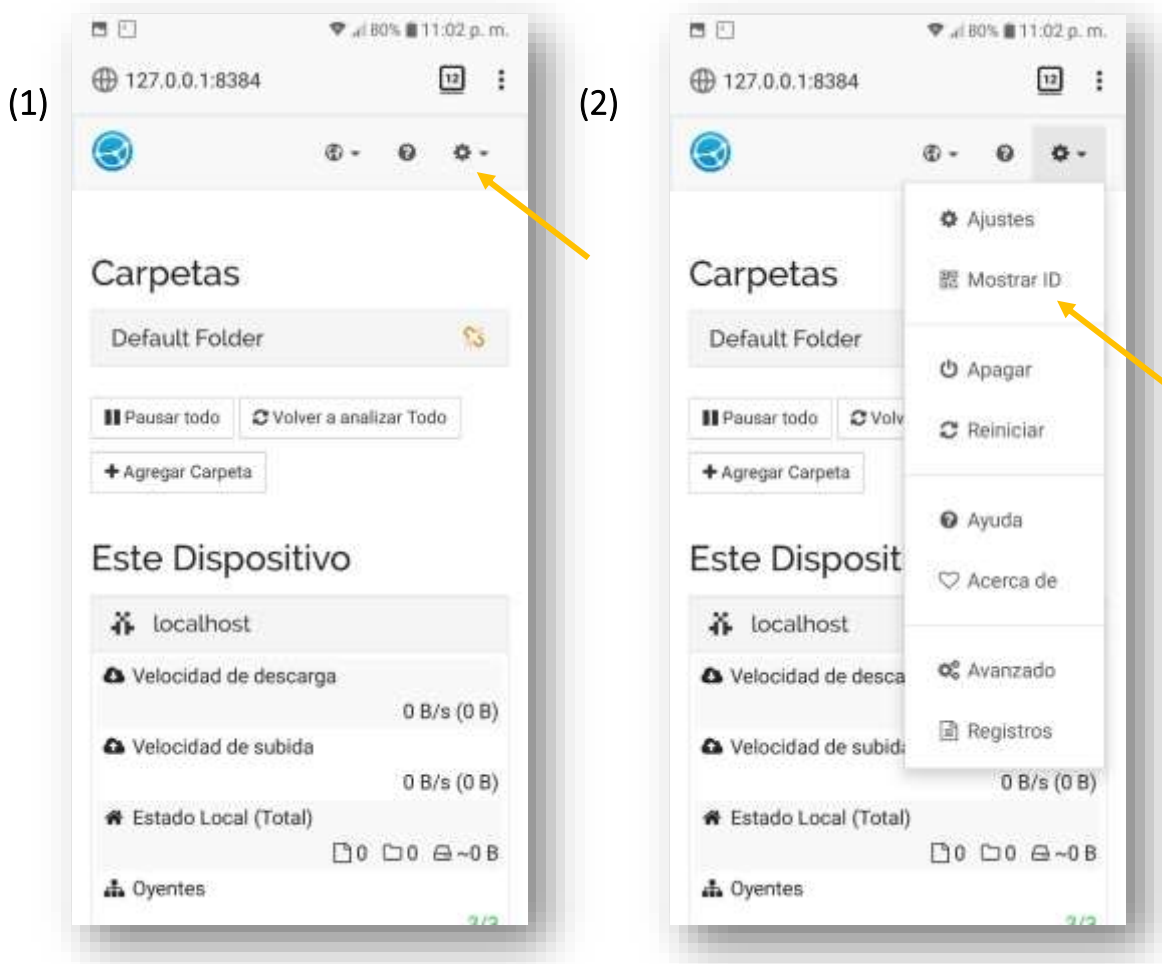
El programa de syncthing cuando se inicia por primera vez se crea un identificador único del teléfono que está compuesto por un grupo de ocho conjuntos de caracteres alfanuméricos en mayúsculas, este identificador (ID) es el que registraremos en el nodo o nodos que deseamos sincronizar la información.

En nuestro caso el ID del teléfono LG Q6 se tendrá que registrar en el teléfono Sansumg y el ID del teléfono Sansumg se tendrá que registrar en el LG Q6. Deben estar en ambos teléfonos para que pueda funcionar correctamente.

Realizaremos los pasos del registro del teléfono móvil Sansumg en el teléfono LG Q6.

Primero (1) en la parte superior de la pantalla de administración (navegador de internet) del teléfono Sansumg con synchthing daremos click en la pestaña de menú lado superior derecho.

Segundo (2) paso nos aparecerá un menú, en este damos click en “Mostrar ID” del Sansumg.

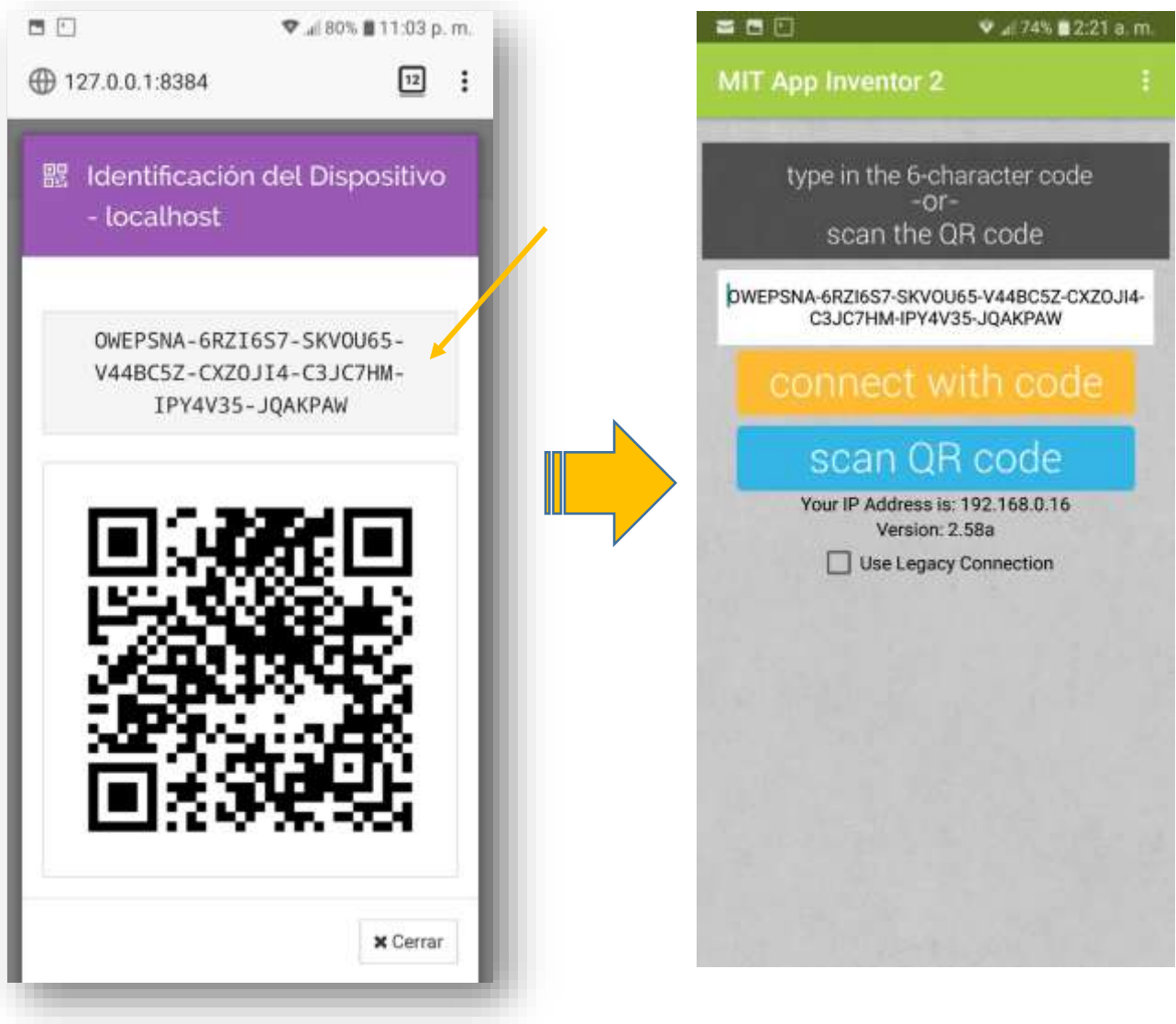


Al dar click en “Mostrar ID” nos aparecerá la siguiente pantalla que es un código QR del teléfono Sansumg en nuestro caso el ID que identifica al teléfono es:

OWEPSNA-6RZI6S7-SKVOU65-V44BC5Z-CXZOJI4-C3JC7HM-IPY4V35-JQAKPAW

Después en el teléfono LG Q6, el programa que nos ayudara a capturar este código QR del Sansumg es el que sugerimos de la aplicación App Inventor (opcional), aunque en caso de no tenerlo también simplemente podemos introducirlo de forma manual cuando iniciemos el registro en el LG Q6, sin embargo, para evitar algún error sugerimos usarlo.

Usando programa de App inventor instalado en el móvil LG Q6 para capturar código QR del teléfono remoto Sansumg que deseamos sincronizar.



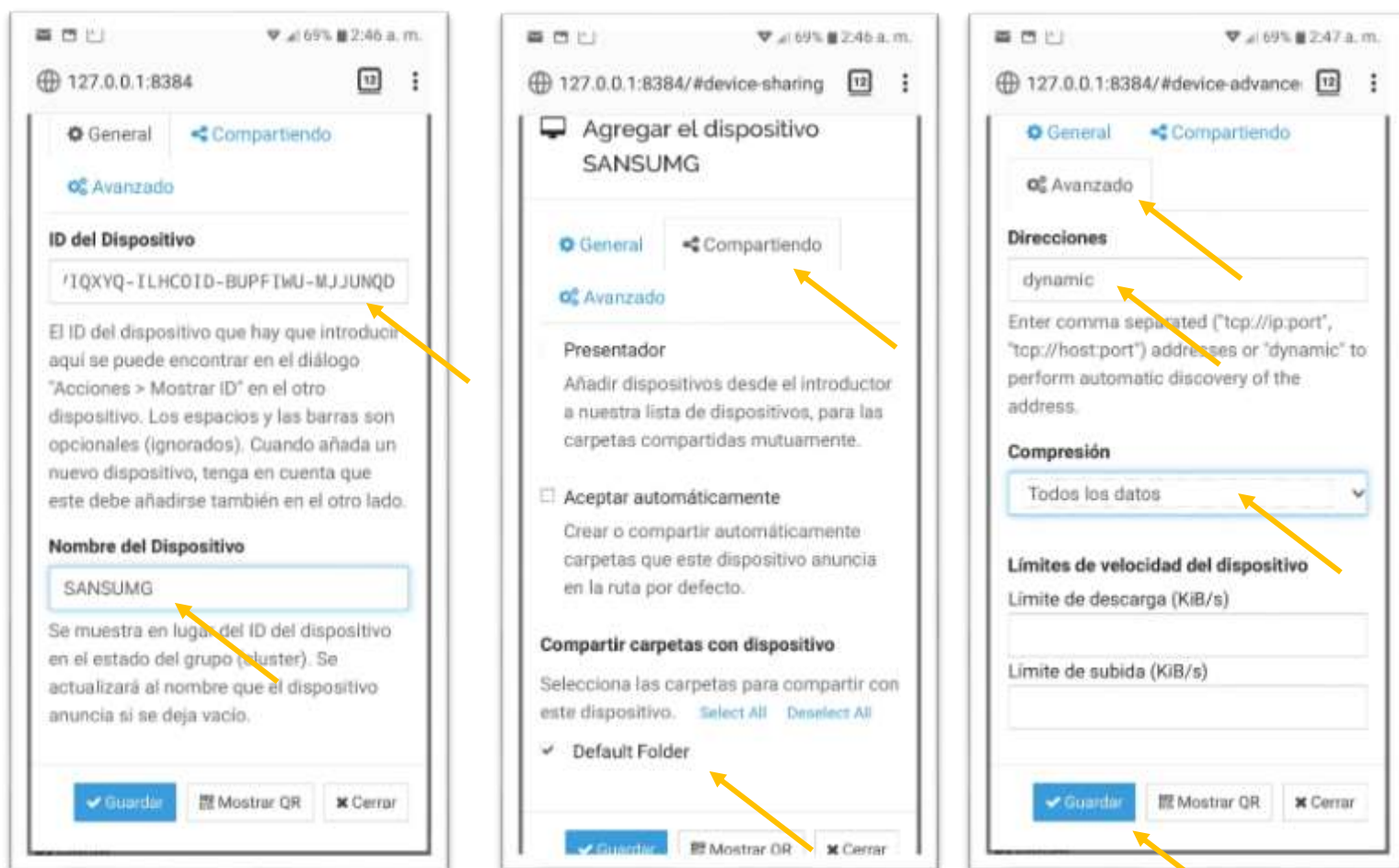
Posteriormente copiamos al portapapeles el código QR en el programa App inventor dando click en el código capturado, escogemos “SELEC TODO” → “COPIAR”.

Con esta información procedemos a registrar el Sansumg en el LG Q6. En la pantalla de administración nos desplazamos a la parte inferior nos situamos en donde dice “Otros dispositivos” y damos click en el botón de “Añadir un dispositivo”, introducimos el ID del Sansumg y le damos nombre de registro.

Posteriormente en la pestaña superior de “Compartir” damos click y en esta marcamos la opción inferior en folder “Default” con esto estaremos compartiendo un directorio que se creó por Default llamado Sync en nuestro dispositivo.

Después en la parte superior damos click en la pestaña “Avanzado” y seleccionamos la opción de compresión de datos en “Todos los datos”.

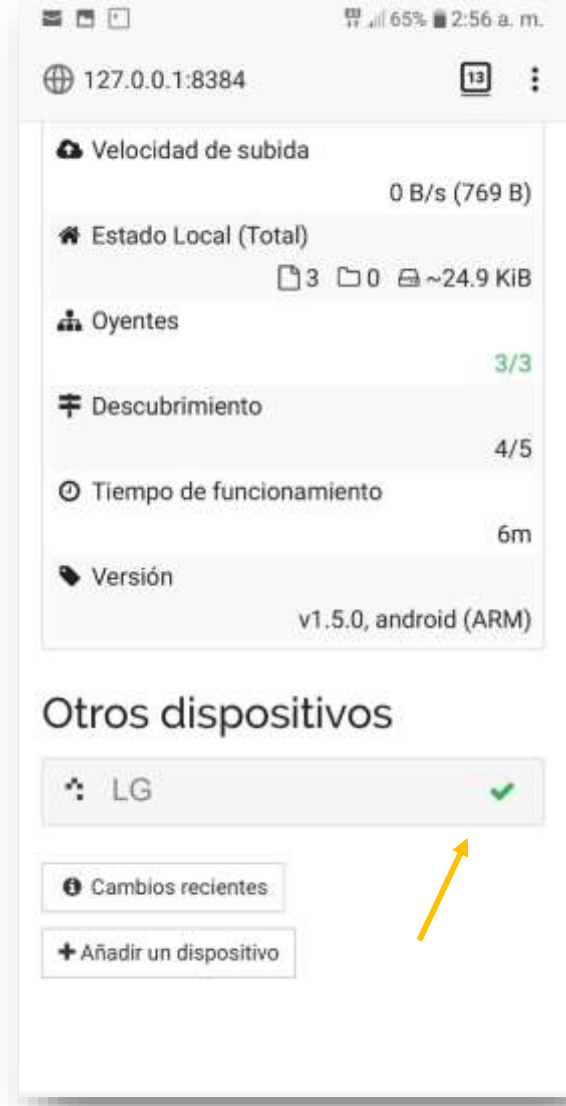
Por ultimo damos click en el botón inferior de guardar, Terminamos el registro en un nodo, este mismo proceso debe hacerse en el teléfono o teléfonos remotos que deseamos sincronizar.



Al guardar e registro en ambos teléfonos esperaremos un periodo máximo de 30 segundos en lo que se encuentran los dispositivos y aparecerá como buena la conexión entre dispositivos dando una confirmación en color verde.

Dispositivo #1 LG Q6

Dispositivo #2 Sansumg



Toda la información que se encuentre en el directorio **Sync** situado en la ruta `/data/data/com.termux/file/home/Sync` se sincronizará, cualquier cambio será copiado y sincronizado.

Instalación de herramienta de administración de syncthing para nodos. Procedemos a la instalación de **SyncthingManager**.

Primero instalamos lo que necesitara para que funcione correctamente SyncthingManager.

```
$ apt install Python
```

```
$ pip3 install --upgrade pip
```

```
$ pip install syncthing
```

\$ pip3 install syncthingmanager

La herramienta SyncthingManager nos ayudara para la sincronización “peer to Peer” de una forma automática y no manual para nuevos y existentes nodos.

```
$ apt install python
Reading package lists... Done
Building dependency tree
Reading state information... Done
python is already the newest version (3.8.3).
0 upgraded, 0 newly installed, 0 to remove and 0
not upgraded.
$
```

```
$ pip3 install --upgrade pip
Requirement already up-to-date: pip in /data/data/com.termux/files/usr/lib/python3.8/site-packages (20.1.1)
$
```

```
$ pip3 install syncthingmanager
Requirement already satisfied: syncthingmanager in /data/data/com.termux/files/usr/lib/python3.8/site-packages (0.1.0)
Requirement already satisfied: syncthing in /data/data/com.termux/files/usr/lib/python3.8/site-packages (from syncthingmanager) (2.3.1)
Requirement already satisfied: python-dateutil==2.6.1 in /data/data/com.termux/files/usr/lib/python3.8/site-packages (from syncthing->syncthingmanager) (2.6.1)
Requirement already satisfied: requests==2.18.4 in /data/data/com.termux/files/usr/lib/python3.8/site-packages (from syncthing->syncthingmanager) (2.18.4)
Requirement already satisfied: six>=1.5 in /data/data/com.termux/files/usr/lib/python3.8/site-packages (from python-dateutil==2.6.1->syncthing->syncthingmanager) (1.15.0)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in /data/data/com.termux/files/usr/lib/python3.8/site-packages (from requests==2.18.4->syncthing->syncthingmanager) (3.0.4)
Requirement already satisfied: idna<2.7,>=2.5 in /data/data/com.termux/files/usr/lib/python3.8/site-packages (from requests==2.18.4->syncthing->syncthingmanager) (2.6)
Requirement already satisfied: certifi>=2017.4.17 in /data/data/com.termux/files/usr/lib/python3.8/site-packages (from requests==2.18.4->syncthing->syncthingmanager) (2020.4.5.1)
Requirement already satisfied: urllib3<1.23,>=1.21.1 in /data/data/com.termux/files/usr/lib/python3.8/site-packages (from requests==2.18.4->syncthing->syncthingmanager) (1.22)
$
```


NOTA IMPORTANTE: En el proceso de instalación de syncthingmanager versión 0.1.0, en la versión con Python 3.8.X al ser instalada en la terminal de Termux puede dar errores de instalación esto se debe a que la versión de Python 3.8.X contiene algunos procesos nuevos para instalar syncthingmanager y tiene errores con la versión de Python syncthing 2.4.2, si se diera el siguiente problema de instalación:

```

$ pip3 install syncthingmanager
Collecting syncthingmanager
  Downloading syncthingmanager-0.1.0.tar.gz (22
kB)
Collecting syncthing
  Downloading syncthing-2.4.2.tar.gz (12 kB)
Installing build dependencies ... done
Getting requirements to build wheel ... done
Preparing wheel metadata ... error
ERROR: Command errored out with exit status
1:
  command: /data/data/com.termux/files/usr/bi
n/python3 /data/data/com.termux/files/usr/lib/py
thon3.8/site-packages/pip/_vendor/pep517/_in_pro
cess.py prepare_metadata_for_build_wheel /data/d
ata/com.termux/files/usr/tmp/tmp0r84lvxf
  cwd: /data/data/com.termux/files/usr/tmp
p/pip-install-6py75806/syncthing
Complete output (16 lines):
Traceback (most recent call last):
  File "/data/data/com.termux/files/usr/lib/
python3.8/site-packages/pip/_vendor/pep517/_in_p
rocess.py", line 280, in <module>

```

```

  File "/data/data/com.termux/files/usr/tmp/p
ip-build-env-qluzxkrl/overlay/lib/python3.8/sit
e-packages/poetry/masonry/builders/builder.py",
line 65, in __init__
    self.module = Module(
  File "/data/data/com.termux/files/usr/tmp/p
ip-build-env-qluzxkrl/overlay/lib/python3.8/sit
e-packages/poetry/masonry/utis/module.py", line
58, in __init__
    raise ModuleOrPackageNotFound(
poetry.masonry.utis.module.ModuleOrPackageN
otFound: No file/folder found for package python
-syncthing
ERROR: Command errored out with exit status 1: /
data/data/com.termux/files/usr/bin/python3 /data
/data/com.termux/files/usr/lib/python3.8/site-pa
ckages/pip/_vendor/pep517/_in_process.py prepare
_metadata_for_build_wheel /data/data/com.termux/
files/usr/tmp/tmp0r84lvxf Check the logs for ful
l command output.

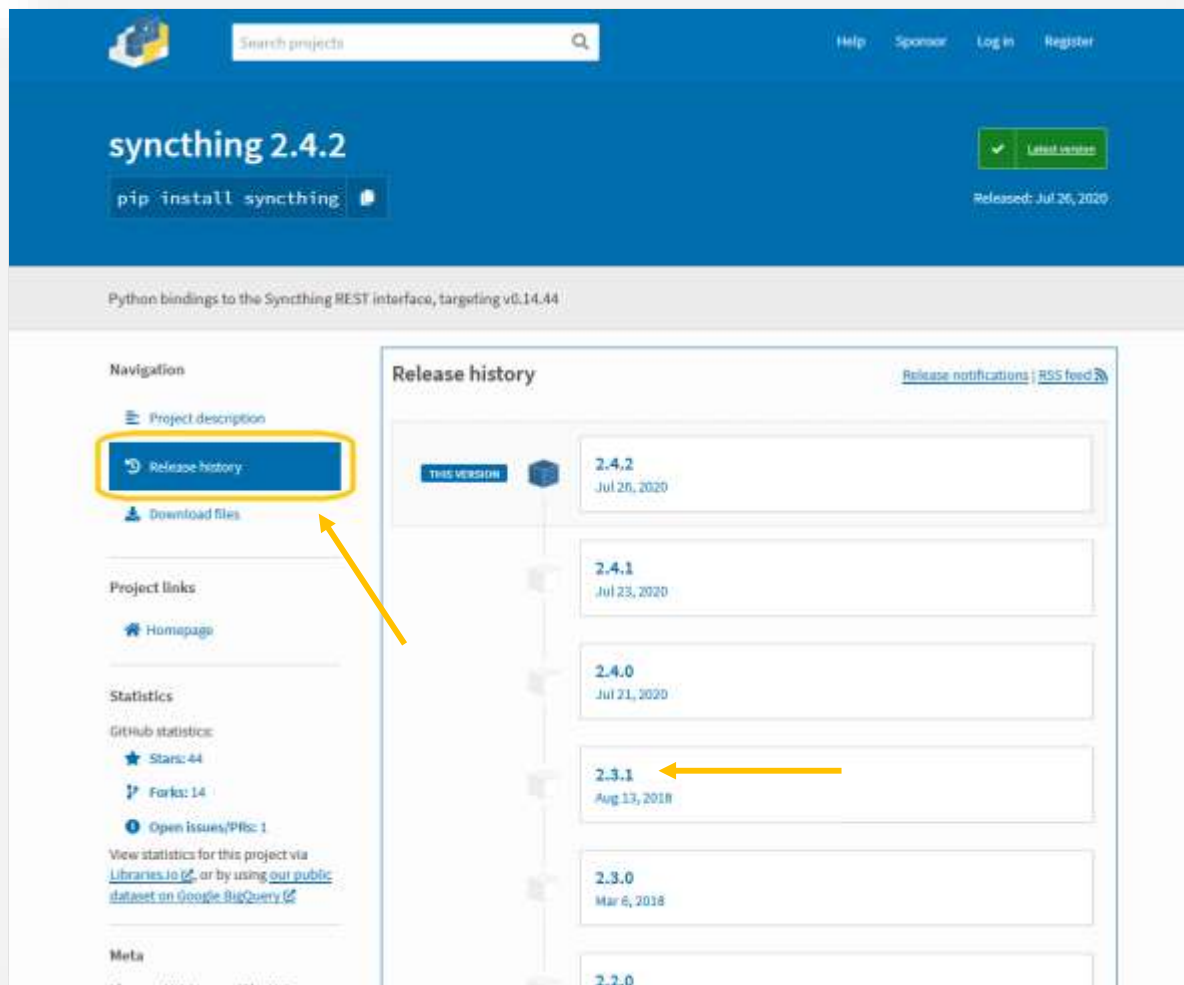
```

Para solucionar este problema tenemos la siguiente opción.

1.- Hacer un downgrade de instalación o tomar una versión anterior a syncthing para Python y posteriormente realizar la instalación de syncmanager esto sería revisar las versiones disponibles en Python para syncthing, ir al sitio:

<https://pypi.org/project/syncthing/>

Ahí podremos revisar todas las versiones anteriores en los “Release history” en este caso nosotros probamos syncthing versión 2.3.1 y se puede instalar sin complicaciones.



Para instalar una versión anterior en Python solo ejecutamos el siguiente comando usando la siguiente sintaxis \$ pip install <paquete == versión>:

```
$ pip install syncthing==2.3.1
```

Posteriormente realizar la instalación de syncthingmanager de forma normal usando la versión 3 de pip con:

```
$ pip3 install syncthingmanager
```

Después de la instalación debemos ejecutar una configuración previa para las variables de ambiente de syncthingmanager ejecutando el siguiente comando en la terminal de Termux:

```
$ stman configure
```

```

$ pip install syncthing==2.3.1
Collecting syncthing==2.3.1
  Downloading syncthing-2.3.1.tar.gz (11 kB)
Collecting python-dateutil==2.6.1
  Downloading python_dateutil-2.6.1-py2.py3-none-any.whl (194 kB)
10 kB 4.
20 kB 2.
30 kB 68.
40 kB 85.
51 kB 88.
61 kB 93.
71 kB 95.
81 kB 84.
92 kB 90.
102 kB 9.
112 kB 9.
122 kB 9.
133 kB 9.
143 kB 9.
153 kB 9.
163 kB 9.
174 kB 9.
184 kB 9.
194 kB 9.
64 kB/s
Collecting requests==2.18.4
  Downloading requests-2.18.4-py2.py3-none-any.whl (88 kB)
10 kB 3.
20 kB 3.
30 kB 3.
40 kB 3.
51 kB 3.
61 kB 4.
71 kB 4.
81 kB 4.
88 kB 98.
4 kB/s
Collecting six>=1.5
  Using cached six-1.15.0-py2.py3-none-any.whl (10 kB)
ESC  =  CTRL  ALT  -  _  |  +

```

```

y.whl (156 kB)
Collecting urllib3<1.23,>=1.21.1
  Downloading urllib3-1.22-py2.py3-none-any.whl (132 kB)
10 kB 6.
20 kB 3.
30 kB 3.
40 kB 3.
51 kB 3.
61 kB 4.
71 kB 4.
81 kB 4.
92 kB 4.
102 kB 4.
112 kB 4.
122 kB 4.
132 kB 4.
.4 MB/s
Collecting chardet<3.1.0,>=3.0.2
  Using cached chardet-3.0.4-py2.py3-none-any.whl (133 kB)
Collecting idna<2.7,>=2.5
  Downloading idna-2.6-py2.py3-none-any.whl (56 kB)
10 kB 5.
20 kB 6.
30 kB 7.
40 kB 7.
51 kB 8.
56 kB 73.
4 kB/s
Using legacy 'setup.py install' for syncthing, since package 'wheel' is not installed.
Installing collected packages: six, python-dateutil, certifi, urllib3, chardet, idna, requests, syncthing
  Running setup.py install for syncthing ...done
Successfully installed certifi-2020.6.20 chardet-3.0.4 idna-2.6 python-dateutil-2.6.1 requests-2.18.4 six-1.15.0 syncthing-2.3.1 urllib3-1.22
$
ESC  =  CTRL  ALT  -  _  |  +

```

```

$ pip3 install syncthingmanager
Collecting syncthingmanager
  Using cached syncthingmanager-0.1.0.tar.gz (22 kB)
Requirement already satisfied: syncthing in /data/data/com.termux/files/usr/lib/python3.8/site-packages (from syncthingmanager) (2.3.1)
Requirement already satisfied: python-dateutil==2.6.1 in /data/data/com.termux/files/usr/lib/python3.8/site-packages (from syncthing->syncthingmanager) (2.6.1)
Requirement already satisfied: requests==2.18.4 in /data/data/com.termux/files/usr/lib/python3.8/site-packages (from syncthing->syncthingmanager) (2.18.4)
Requirement already satisfied: six>=1.5 in /data/data/com.termux/files/usr/lib/python3.8/site-packages (from python-dateutil==2.6.1->syncthing->syncthingmanager) (1.15.0)
Requirement already satisfied: certifi>=2017.4.17 in /data/data/com.termux/files/usr/lib/python3.8/site-packages (from requests==2.18.4->syncthing->syncthingmanager) (2020.6.20)
Requirement already satisfied: urllib3<1.23,>=1.21.1 in /data/data/com.termux/files/usr/lib/python3.8/site-packages (from requests==2.18.4->syncthing->syncthingmanager) (1.22)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in /data/data/com.termux/files/usr/lib/python3.8/site-packages (from requests==2.18.4->syncthing->syncthingmanager) (3.0.4)
Requirement already satisfied: idna<2.7,>=2.5 in /data/data/com.termux/files/usr/lib/python3.8/site-packages (from requests==2.18.4->syncthing->syncthingmanager) (2.6)
Using legacy 'setup.py install' for syncthingmanager, since package 'wheel' is not installed.
Installing collected packages: syncthingmanager
  Running setup.py install for syncthingmanager ...done
Successfully installed syncthingmanager-0.1.0
$
ESC  =  CTRL  ALT  -  _  |  +

```

Para comprobar si la instalación esta correcta podemos usar el comando:

`$ stman -h`

Este nos dara el siguiente resultado:

```

$ stman -h
usage: stman [-h] [--config CONFIG]
            [--device NAME]
            action ...

positional arguments:
  action
    configure           configure stman. If the configuration file
                        specified in -c (or the default) does not
                        exist, it will be created. To edit an
                        existing configuration, specify all options
                        again.
  device
  folder               work with devices
                        work with folders

optional arguments:
  -h, --help            show this help message and exit
  --config CONFIG, -c CONFIG
                        stman configuration file
  --device NAME, -d NAME
                        the configured API to use

```


Para usar syncthingmanager podremos apoyarnos usando la extensión (OpenQbitP2PwithSSH) con esta extensión podremos ejecutar todos los comandos en línea de “stman” del syncthingmanager como requisito debemos tener ejecutando syncthing en la terminal de Termux para poder ejecutar los comandos en línea ejemplos enseguida y/o usar la extensión (OpenQbitP2PwithSSH) que contiene todos los comandos de syncthingmanager:

Para ejecutar syncthing en la terminal de termux solamente tenemos que usar el siguiente comando:

\$ syncthing

Ejemplos de comandos en línea para administra syncthing a través de syncthingmanager desde una terminal de Termux.

```
# Autoconfiguration
$ stman configure

# List configured devices
$ stman device list
syncthingmanager-test      This Device
    ID:      LYAB7ZG-XDVMAVM-OUZ7EAB-5N3UVWY-DXTFRJ4-U2MTHGQ-7TIBRJE-PC56BQ6

another-device             Connected
    At:      # Address removed
    Folders: dotest
    ID:      H2AJWNR-5VYNWKM-PS2L2EE-QJYBG2U-3IFN5XM-EKSI IKF-NVLAG2E-KIQE4AE

# List configured folders
$ stman folder list
Default Folder
    Shared With:
    Folder ID:  default
    Folder Path: /home/syncthing/Sync/

do-test
    Shared With: another-device
    Folder ID:  dotest
    Folder Path: /home/syncthing/stman-test/

# Adding a device
$ stman device add MFZWI3D-BONSGYC-YLTMRWG-C43ENR5-QXGZDMM-FZWI3DP-BONSGYY-LTMRWAD -n yet-another-device -i
```

\$ stman device list

syncthingmanager-test This Device
ID: LYAB7ZG-XDVMAVM-OUZ7EAB-5N3UVWY-DXTFRJ4-U2MTHGQ-7TIBRJE-PC56BQ6

another-device Connected
At: #Address removed
Folders: dotest
ID: H2AJWNR-5VYNWKM-PS2L2EE-QJYBG2U-3IFN5XM-EKSIKIF-NVLAG2E-KIQE4AE

yet-another-device Not Connected
Folders:
ID: MFZWI3D-BONSGYC-YLTMRWG-C43ENR5-QXGZDMM-FZWI3DP-BONSGYY-LTMRWAD

Share a folder with a device

\$ stman folder share dotest yet-another-device

\$ stman folder list

Default Folder
Shared With:
Folder ID: default
Folder Path: /home/syncthing/Sync/

do-test
Shared With: another-device, yet-another-device
Folder ID: dotest
Folder Path: /home/syncthing/stman-test/

Configure and view advanced options

\$ stman folder versioning dotest simple --versions 15

\$ stman folder edit dotest -r 70

\$ stman folder info dotest

do-test
Shared With: another-device, yet-another-device
Folder ID: dotest
Folder Path: /home/syncthing/stman-test/
Rescan Interval: 70
File Pull Order: alphabetic
Versioning: simple
Keep Versions: 15

Más información de uso syncthingmanager en:

<https://github.com/classicsc/syncthingmanager>

9. Ambientes Blockly (App Inventor, AppyBuilder y Thunkable).

App Inventor es un entorno de desarrollo de software creado por Google Labs para la elaboración de aplicaciones destinadas al sistema operativo Android. El usuario puede, de forma visual y a partir de un conjunto de herramientas básicas, ir enlazando una serie de bloques para crear la aplicación. El sistema es gratuito y se puede descargar fácilmente de la web. Las aplicaciones creadas con App Inventor son muy fáciles de crear debido a que no se necesita conocimiento de ningún tipo de lenguaje de programación.

Todos los ambientes actuales que usan la tecnología de Blockly como son AppyBuilder y Thunkable entre otros tienen su versión gratuita, su forma de uso puede ser a través de internet en sus diversos sitios o también puede ser instalado en casa.

Los bloques que forman la arquitectura “Peer to Peer” han sido probados en App inventor y AppyBuilder sin embargo por su optimización de código deberán funcionar en las otras plataformas.

Versiones de por internet:

App Inventor.

<https://appinventor.mit.edu/>

AppyBuilder.

<http://appybuilder.com/>

Thunkable.

<https://thunkable.com/>

Versión para ser instalada en tu equipo (PC):

<https://sites.google.com/site/aprendeappinventor/instala-app-inventor>

Ambiente para desarrolladores de bloques Blockly.

<https://editor.appybuilder.com/login.php>

10. ¿Qué es un IdDevice o identificador de dispositivo en Syncthing?

Comprender los identificadores de dispositivos

Cada dispositivo se identifica mediante una ID de dispositivo. La identificación del dispositivo se utiliza para la resolución de direcciones, autenticación y autorización. El término "ID de dispositivo" podría haber sido intercambiable "ID de clave" ya que la ID de dispositivo es una propiedad directa de la clave pública en uso.

Llaves

Para comprender las ID de los dispositivos, debemos observar los mecanismos subyacentes. En el primer inicio, Syncthing creará un par de claves público / privado.

Actualmente se trata de una clave ECDSA de 384 bits (RSA de 3072 bits anterior a v0.12.5, que es lo que se usa como ejemplo en este artículo). Las claves se guardan en forma de clave privada (key.pem) y un certificado autofirmado (cert.pem). La parte de firma automática no agrega ninguna seguridad o funcionalidad en lo que respecta a Syncthing, pero permite el uso de las claves en un intercambio TLS estándar.

ID de dispositivo

Para formar una ID de dispositivo, se calcula el hash SHA-256 de los datos del certificado en forma DER. Esto significa que el hash cubre toda la información del Certificado: sección anterior.

El hash resulta en un hash de 256 bits que codificamos usando base32. Base32 codifica cinco bits por carácter, por lo que necesitamos $256/5 = 51.2$ caracteres para codificar la ID del dispositivo. Esto se convierte en 52 caracteres en la práctica, pero 52 caracteres de base32 se descodificarían a 260 bits, lo que no es un número entero de bytes. La codificación base32 agrega relleno a 280 bits (el siguiente múltiplo de 5 y 8 bits) para que la ID resultante se vea más o menos así:

```
MFZWI3DBONSGYYLTMRWGC43ENRQXGZDMMFZWI3DBONSGYYLTMRWA ====
```

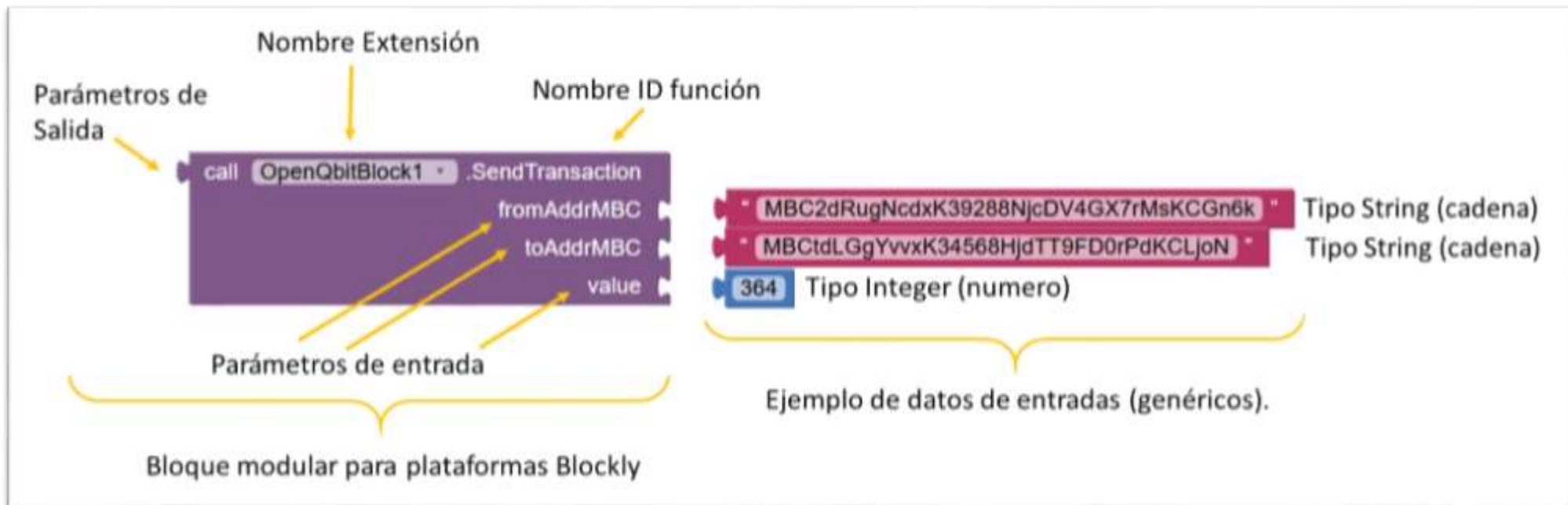
El relleno (====) se elimina, la ID del dispositivo se divide en cuatro grupos y se agregan dígitos de verificación para cada grupo. Para fines de presentación, la ID del dispositivo se agrupa con guiones, lo que da como resultado el valor final ejemplo:

MFZWI3D-BONSGYC-YLTMRWG-C43ENR5-QXGZDMM-FZWI3DP-BONSGYY-LTMRWAD

11. Definición y uso de bloques en OpenQbitP2PwithSSH.

Empezaremos con explicar la distribución de los datos que tendrá todos los bloques, su sintaxis de uso y configuración.

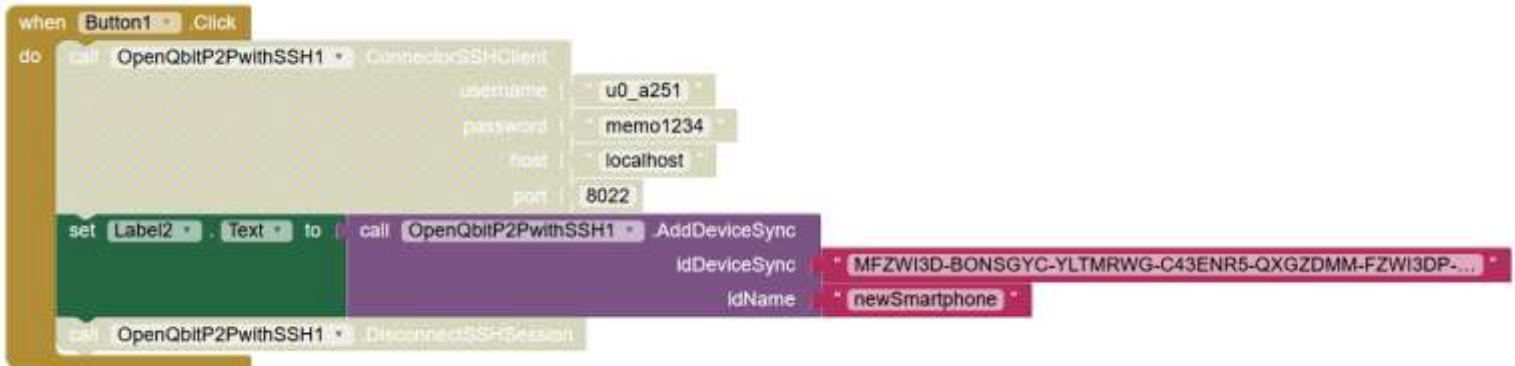
En el siguiente ejemplo genérico podemos ver un bloque modular y sus parámetros de entrada y salida, así como los tipos de datos de entrada, estos datos pueden ser de tipo String (cadena de caracteres) o Integer (numero entero o decimal). Mostramos como es su uso y configurarlo para su funcionamiento adecuado.



Cada bloque modular tendrá su descripción y se nombrará en caso de tener alguna(s) dependencia(s) obligatoria(s) u opcional(es) de otros bloques usados como parámetros de entrada y se anunciará el proceso de integración.

12. Extensión OpenQbitP2PwithSSH.

Bloque para agregar un nuevo dispositivo (**AddDeviceSync**)



Dependencia obligatoria: Bloque (**ConnectorSSHClient**), Bloque (**DisconnectSSHSession**).

Parámetros de entrada: **idDeviceSyn** <string>, **idName** <string>

idDevice. - es un arreglo de caracteres en mayúsculas dividido en 8 conjuntos, cada conjunto está integrado de 7 elementos. La cadena debe medir un total de 56 caracteres.

Ejemplo:

MFZWI3D-BONSGYC-YLTMRWG-C43ENR5-QXGZDMM-FZWI3DP-BONSGYY-LTMRWAD

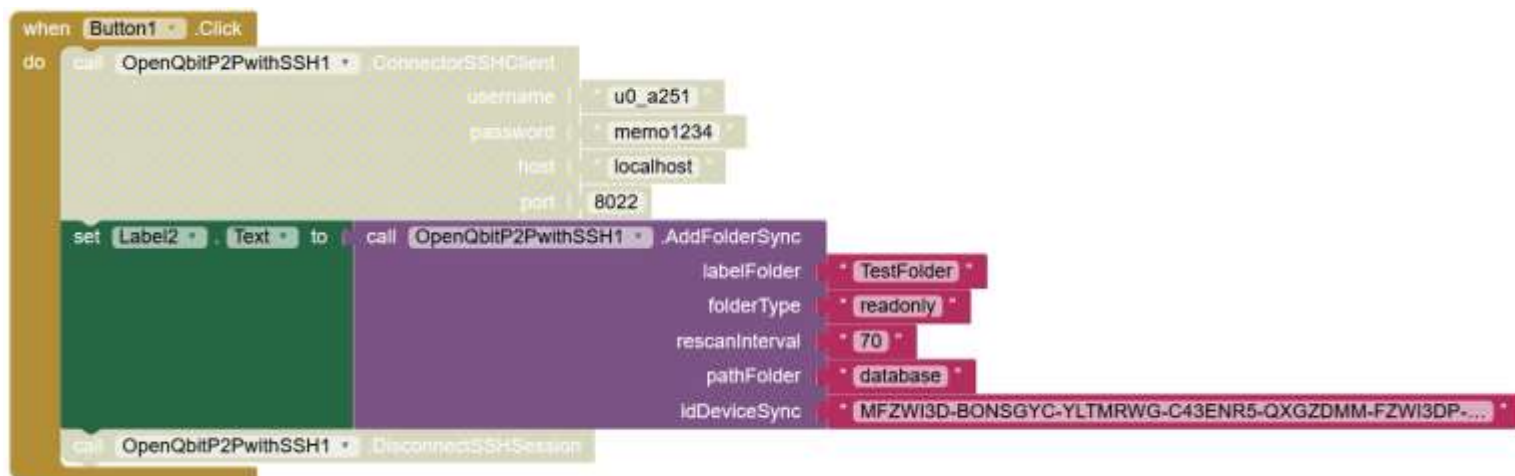
idName. - es un identificador o etiqueta que el usuario escoge para identificar al nuevo dispositivo.

Parámetros de salida: Entrega un mensaje. **"Device already configured: idDevice"**.

Usar enseguida el bloque (**GetDeviceList**) para comprobar que se agregó el Id dispositivo.

Descripción: Bloque de comunicación para agregar un nuevo dispositivo para conectar a syncthing ejecutándose en la terminal Termux, vía protocolo de comunicación SSH (Secure Shell).

Bloque para agregar un nuevo folder o directorio para sincronizar (**AddFolderSync**)



Dependencia obligatoria: Bloque (**ConnectorSSHClient**), Bloque (**DisconnectSSHSession**).

Parámetros de entrada: **labelFolder** <string>, **folderType** <string – opciones: readonly or readwrite>, **rescanInterval** <string>, **pathFolder** <string>, **idDevice**<string>

labelFolder. - Etiqueta que identifica al directorio que será compartido entre los dispositivos.

folderType. - Este será los permisos que tendrá el directorio si solo lectura “readonly” o lectura y escritura “readwrite”.

rescanInterval. - Es el intervalo de tiempo en segundo para que revise las deltas de cambios en el origen y realice la sincronización al directorio local.

pathFolder. - Es la ruta “path” del directorio o folder que será compartido.

idDevice. - Es un arreglo de caracteres en mayúsculas dividido en 8 conjuntos, cada conjunto está integrado de 5 elementos. La cadena debe medir un total de 40 caracteres.

Ejemplo:

MFZWI3D-BONSGYC-YLTMRWG-C43ENR5-QXGZDMM-FZWI3DP-BONSGYY-LTMRWAD


idName. - Es un identificador o etiqueta que el usuario escoge para identificar al nuevo dispositivo.

Parámetros de salida: Entrega un mensaje. **"The folder ID idDevice is already in use."** En caso de ser éxito la ejecución del comando no entrega resultado.

Usar enseguida el bloque (**InfoFolderSync**) para comprobar que se configuro el directorio.

Descripción: Bloque de comunicación para agregar una nueva configuración del directorio para conectar a syncthing ejecutándose en la terminal Termux, vía protocolo de comunicación SSH (Secure Shell).

Bloque para obtener el API-KEY del syncthing local que se esté ejecutando (**ApyKeySyncthing**)

 call **OpenQbitP2PwithSSH1** ▾ **.ApyKeySyncthing**

Dependencia obligatoria: Bloque (**ConnectorSSHClient**), Bloque (**DisconnectSSHSession**).

Parámetros de entrada: **Ninguno – N/A**

Parámetros de salida: No entrega el contenido del archivo de configuración en donde se encuentra el API-KEY para ser utilizado con el Rest API de Syncthing.

Ejemplo:



```
[DEFAULT]
name = localhost

[localhost]
apikey = MafkDvpagX5J6oMzxm9HwDSXJPSQKPFS
hostname = localhost
port = 8384

[remote-device]
apikey = h9mifaKwDq3SSPPmgUuDjsrivFg3dtkK
hostname = some-host
port = 9001
```

Descripción: Bloque de comunicación obtener el contenido del archivo de configuración de syncthing que contiene el API-KEY.

Bloque para Ejecutar comandos en terminal Termux Linux (**CommandLineExecuteSSH**).

 call **OpenQbitP2PwithSSH1** ▾ **.CommandLineExecuteSSH**
command  **"md5sum ./test.txt"**

Dependencia obligatoria: Bloque (**ConnectorSSHClient**), Bloque (**DisconnectSSHSession**).

Parámetros de entrada: **command** <string>

Parámetros de salida: Dato variable, depende del comando ejecutado o programa.

Descripción: Bloque de ejecución de comandos en la terminal Termux pre-requisito hacer una conexión con el bloque (**ConnectorSSHClient**) puede ejecutar todo tipo de comandos en línea y/o obtener datos específicos de ejecución de scripts o programas que tengan un CLI (Command-Line Interface) en línea.

Bloque Conector SSH Cliente (ConnectorSSHClient)

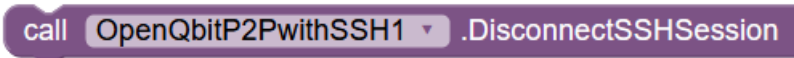


Parámetros de entrada: **username** <string>, **password** <string>, **host** <string>, **port**<integer>

Parámetros de salida: No aplica – N/A.

Descripción: Bloque de comunicación para conectar al servidor SSH de la terminal Termux, vía protocolo de comunicación SSH (Secure Shell).

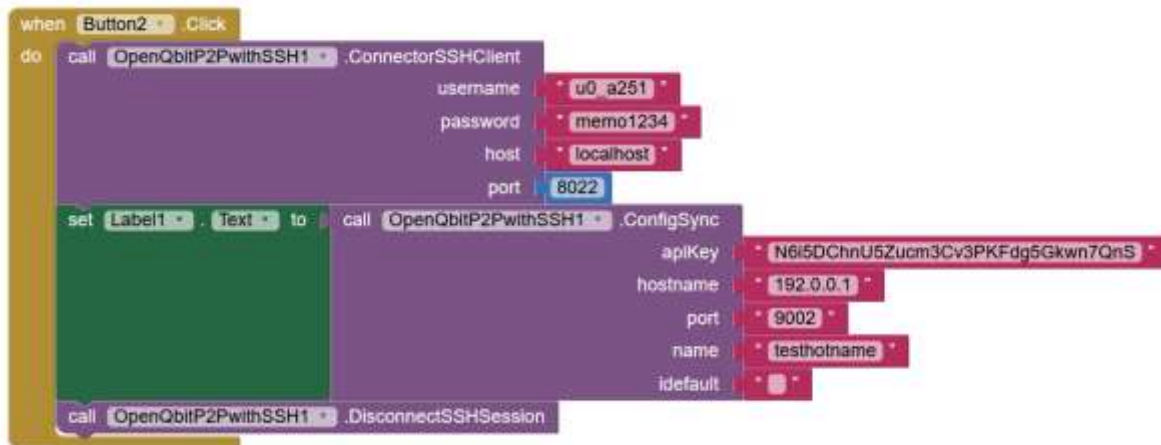
Bloque para cerrar la sesión SSH (DisconnectSSHSession).



Parámetros de entrada y salida: No aplica (ninguno).

Descripción: Bloque para cerrar sesión de SSH.

Bloque para modificar el archivo de configuración (ConfigSync)



Dependencia obligatoria: Usar antes con Bloque (ConnectorSSHClient), Bloque (DisconnectSSHSession).

Parámetros de entrada: **a piKey** <string>, **hostname**<string>, **port**<string>, **name**<string>, **pdefault**<string>.

apiKey. - Es el API-KEY del syncthing local o remoto que se desee cambiar parámetros del archivo de configuración de syncthingmanager.

hostname. - Es el nombre del dispositivo al cual se hará referencia los cambios, para cambiar el local archivo del dispositivo el parámetro a usar es "localhost".

port. - Es el puerto en el cual se conectarán las peticiones al syncthing.

name. - Es el identificador o etiqueta dentro del archivo de configuración de syncthingmanager.

iddefault.- Es el parámetro para definir el dispositivo local como default si la opción se deja vacía no será el default, en caso de querer que sea esta configuración la default se tiene que usar el parámetro - - **default** , el parámetro inicia con dos signos de menos "-" y después la palabra default todo debe ir junto sin espacios.

Parámetros de salida: Escribe los parámetros en el archivo de configuración de syncthingmanager que está en la ruta: `~/config/syncthingmanager/syncthingmanager.conf`

Descripción: Bloque de ejecución parámetros para modificar el archivo de configuración de syncthingmanager.

Bloque para eliminar un dispositivo que y está registrado en el syncthing local (**DeleteDeviceSync**)

```
call OpenQbitP2PwithSSH1 .DeleteDeviceSync
idDeviceSync " MFZWI3D-BONSGYC-YLTMRWG-C43ENR5-QXGZDMM-FZWI3DP-..."
```

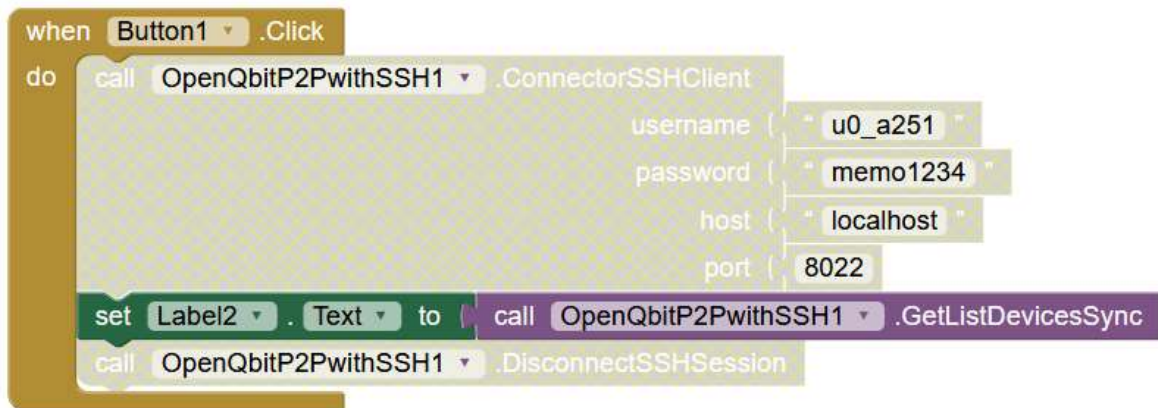
Dependencia obligatoria: Usar antes con Bloque (**ConnectorSSHClient**), Bloque (**DisconnectSSHSession**).

Parámetros de entrada: No aplica – N/A

Parámetros de salida: Entrega el mensaje "Successfully delete id = **idDevideSync**" después de ejecutar este bloque se puede ejecutar el bloque (**GetDeviceList**) para verificar que se haya eliminado el idDevice deseado.

Descripción: Bloque para borrar un dispositivo del syncthing local.

Bloque para listar todos los idDevice en el synching local (**GetListDeviceSync**)



Dependencia obligatoria: Usar antes con Bloque (**ConnectorSSHClient**), Bloque (**DisconnectSSHSession**).

Parámetros de entrada: No aplica (ninguno).

Parámetros de salida: Entrega la lista de IdDevice o identificadores de los dispositivos registrados en el synching local.

Ejemplo:

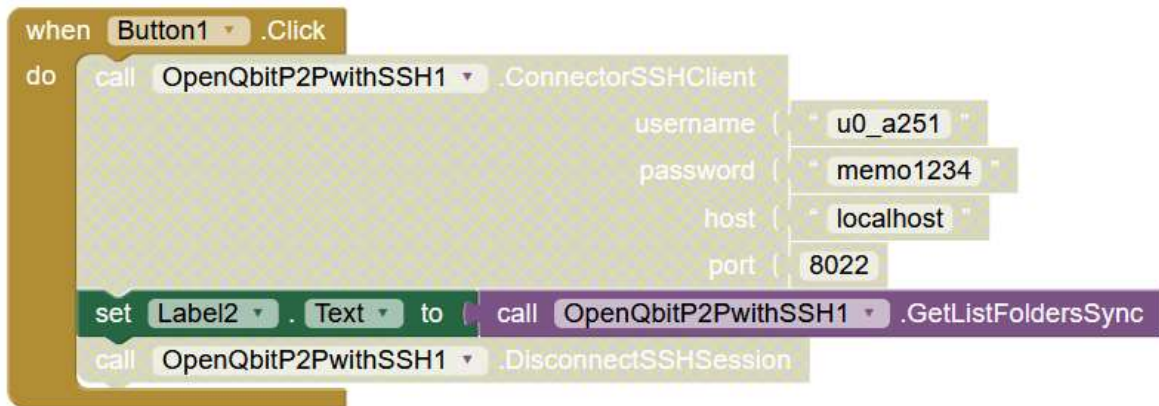
```
syncthingmanager-test      This Device
  ID:      LYAB7ZG-XDVMAVM-OUZ7EAB-5N3UVWY-DXTFRJ4-U2MTHGQ-7TIBRJE-PC56BQ6
```

```
another-device      Connected
  At:      #Address removed
  Folders:  dotest
  ID:      H2AJWNR-5VYNWKM-PS2L2EE-QJYBG2U-3IFN5XM-EKSIIKF-NVLAG2E-KIQE4AE
```

```
yet-another-device      Not Connected
  Folders:
  ID:      MFZWI3D-BONSGYC-YLTMRWG-C43ENR5-QXGZDMM-FZWI3DP-BONSGYY-LTMRWAD
```

Descripción: Bloque para generar la lista de ID.

Bloque para generar lista de los directorios compartidos (**GetListFoldersSync**)



Dependencia obligatoria: Usar antes con Bloque (**ConnectorSSHClient**), Bloque (**DisconnectSSHSession**).

Parámetros de entrada: No aplica (ninguno).

Parámetros de salida: Entrega la lista de folders o directorios compartidos.

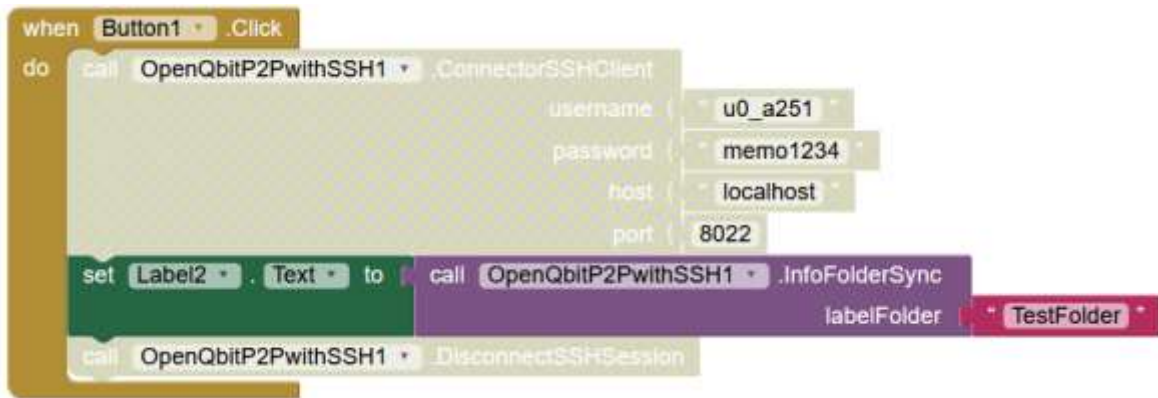
Ejemplo:

```
Default Folder
  Shared With:
  Folder ID: default
  Folder Path: /home/syncthing/Sync/
```

```
do-test
  Shared With: another-device
  Folder ID: dotest
  Folder Path: /home/syncthing/stman-test/
```

Descripción: Bloque para generar la lista de folders o directorios compartidos locales y remotos.

Bloque para obtener la configuración de un folder o directorio (**InfoFolderSync**)



Dependencia obligatoria: Usar antes con Bloque (**ConnectorSSHClient**), Bloque (**DisconnectSSHSession**).

Parámetros de entrada: No aplica (ninguno).

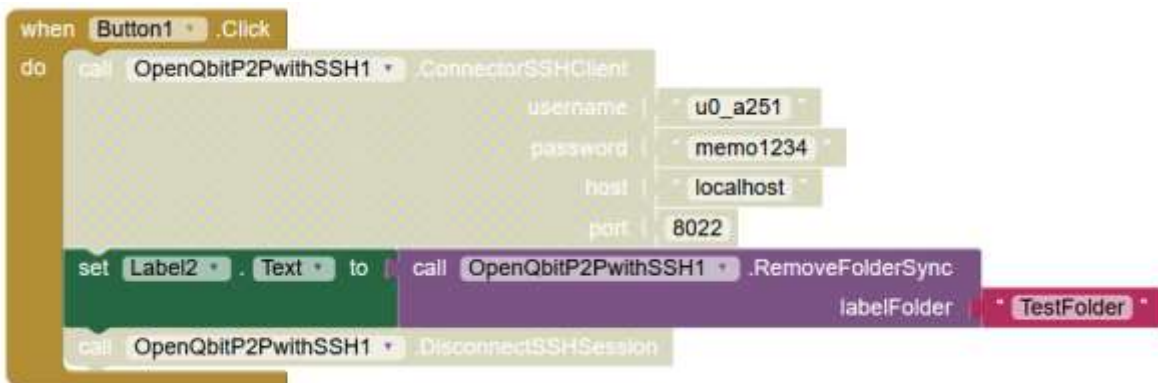
Parámetros de salida: Entrega la configuración del folder o directorio compartidos.

Ejemplo:

```
TestFolder
Shared With:  another-device, yet-another-device
Folder ID:   dotest
Folder Path:  /home/syncthing/stman-test/
Rescan Interval:  70
File Pull Order: alphabetic
Versioning:      simple
Keep Versions:   15
```

Descripción: Bloque para ver la configuración parámetros que tiene un folder o directorio seleccionado ya sea local y remoto.

Bloque para borrar un folder o directorio compartido (**RemoveFoldersSync**)



Dependencia obligatoria: Usar antes con Bloque (**ConnectorSSHClient**), Bloque (**DisconnectSSHSession**).

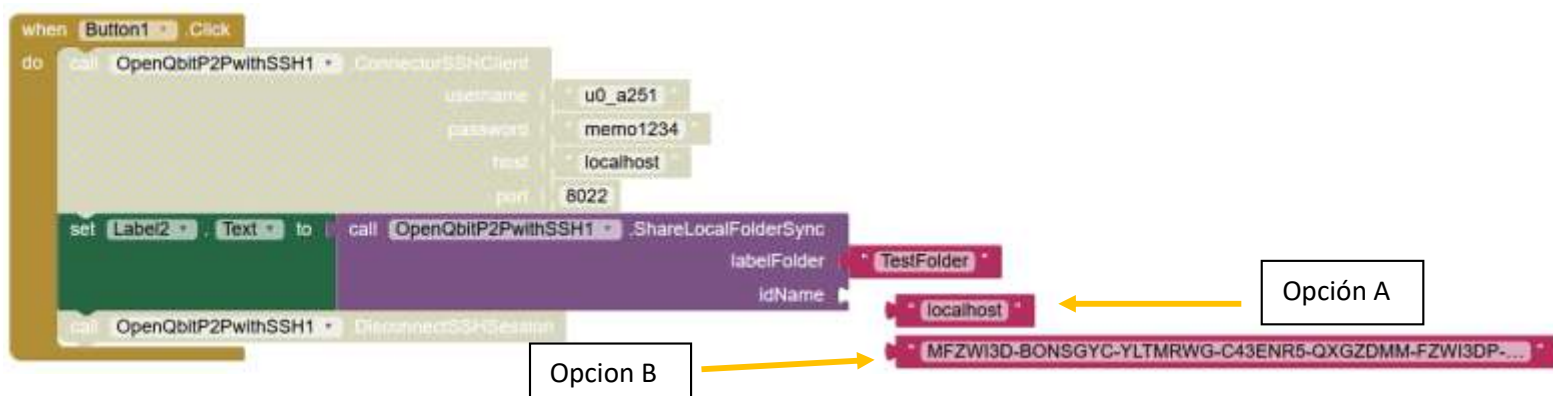
Parámetros de entrada: No aplica (ninguno).

Parámetros de salida: No aplica – N/A.

Para revisar si se ha realizado correctamente el borrado en la lista de directorios compartidos usar el bloque (**GetListFoldersSync**).

Descripción: Remueve o borra un folder o directorio que esta en la lista para compartir en syncthing.

Bloque para compartir folder o directorio (**ShareLocalFolderSync**)



Dependencia obligatoria: Usar antes con Bloque (**ConnectorSSHClient**), Bloque (**DisconnectSSHSession**), Bloque (**AddFoldersSync**)

Parámetros de entrada: **labelFolder** <string>, **idName**<string>.

labelFolder. - Etiqueta que identifica al directorio que dio cuando se registró el folder o directorio con el bloque (**AddFoldersSync**) esta etiqueta de puede ver ejecutando el bloque (**GetListFoldersSync**).

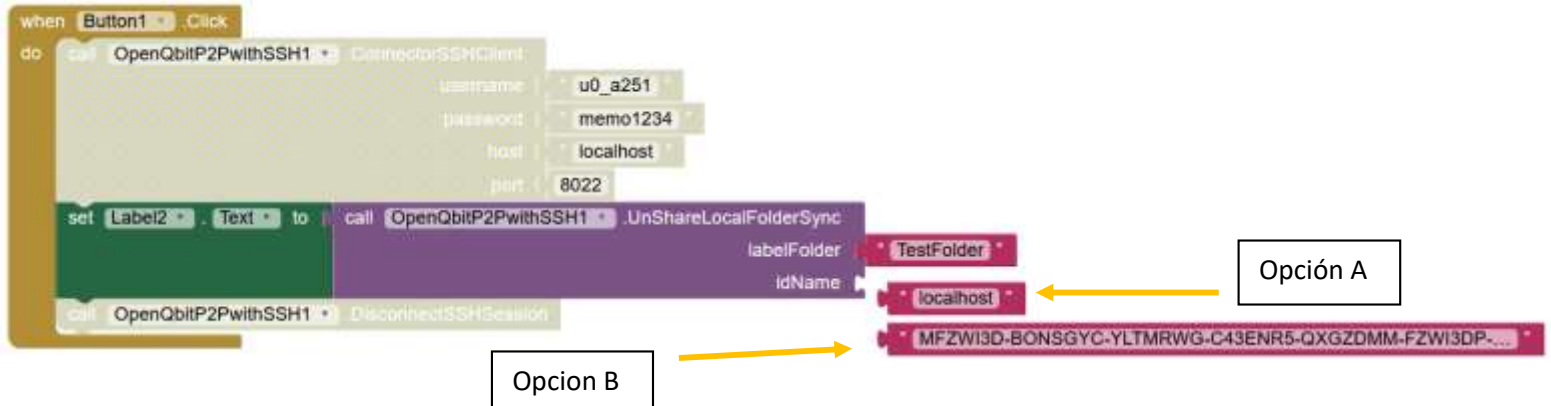
Ejemplo de l información del bloque (**GetListFoldersSync**).

```
Default Folder
  Shared With:
  Folder ID:  default
  Folder Path:  /home/syncthing/Sync/
```

```
TestFolder
  Shared With:  another-device
  Folder ID:  dotest
  Folder Path:  /home/syncthing/stman-test/
```

idName. – Hay dos opciones para usar la opción A es utilizar para el dispositivo local “localhost” o como opción B también se puede introducir el Identificador del dispositivo **idDevice** para dispositivo local o remoto.

Bloque para quitar el servicio de compartir folder o directorio (**UnShareLocalFoldersSync**)



Dependencia obligatoria: Usar antes con Bloque (**ConnectorSSHClient**), Bloque (**DisconnectSSHSession**), bloque (**ShareLocalFoldersSync**)

Parámetros de entrada: **labelFolder** <string>, **idName**<string>.

labelFolder. - Etiqueta que identifica al directorio que dio cuando se registró el folder o directorio con el bloque (**AddFoldersSync**) esta etiqueta de puede verse ejecutando el bloque (**GetListFoldersSync**).

Ejemplo de l información del bloque (**GetListFoldersSync**).

```
Default Folder
  Shared With:
  Folder ID: default
  Folder Path: /home/syncthing/Sync/
```

```
TestFolder
  Shared With: another-device
  Folder ID: dotest
  Folder Path: /home/syncthing/stman-test/
```

idName. – Hay dos opciones para usar la opción A es utilizar para el dispositivo local “localhost” o como opción B también se puede introducir el Identificador del dispositivo **idDevice** para dispositivo local o remoto.

Bloque para configurar la propiedad de versión de folder o directorio (**VersioningFoldersSyn**)



Dependencia obligatoria: Usar antes con Bloque (**ConnectorSSHClient**), Bloque (**DisconnectSSHSession**).

Parámetros de entrada: **labelFolder** <string>, **vType**<string>, **numVersion**<string>

labelFolder. - - Etiqueta que identifica al directorio que dio cuando se registro el folder o directorio con el bloque (**AddFoldersSync**) esta etiqueta de puede verse ejecutando el bloque (**GetListFoldersSync**).

vType.- Se manejan tres tipos (Simple, Staggered y External) para ver detalles hacer referencia al sitio: <https://docs.syncthing.net/users/versioning.html?highlight=version>

numVersion.- Es el número de versión especificada al folder o directorio.

Parámetros de salida: No aplica – N/A.

Para revisar si se ha realizado correctamente los parámetros de versión usar el bloque (**InfoFoldersSync**).

Ejemplo:

```
TestFolder
Shared With:  another-device, yet-another-device
Folder ID:   dotest
Folder Path:  /home/syncthing/stman-test/
Rescan Interval:  70
File Pull Order:  alphabetic
Versioning:      simple
Keep Versions:    15
```

Descripción: Coloca los parámetros para el control de versión en directorios.

Como usar los bloques para usar la red “Tor”.

Antes debemos instalar el comando para usarlo en línea de la red “Tor” con la ejecución del siguiente comando en la Terminal de termux o en su caso en el Linux de la PC o servidor instalado.

Para Terminal termux:

\$ pkg install torsocks

Para Linux:

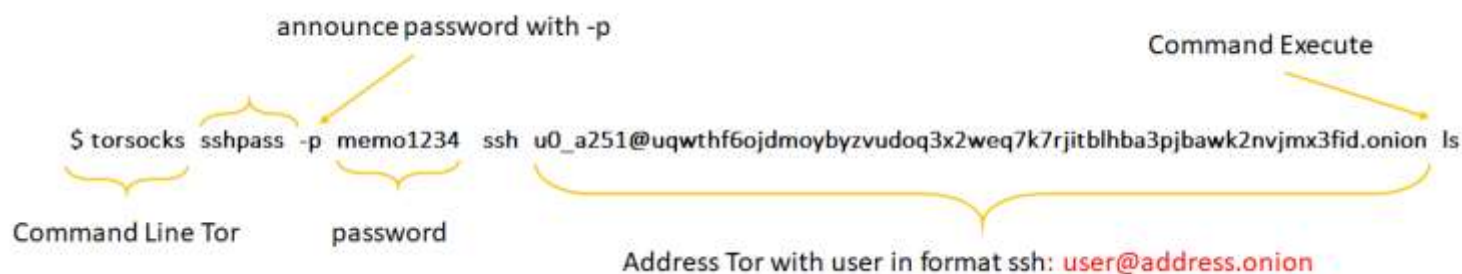
\$ apt install torsocks

Para usar el comando en línea **torsocks** necesitamos tener la dirección Tor del dispositivo remoto. Tenemos dos formas de saber esta, una es en el dispositivo remoto damos el siguiente comando en la Terminal de termux o Linux:

\$ more ~/.tor/hidden_ssh/hostname (ver que el directorio inicial tor es oculto y debe usarse un punto “.” antes del nombre directorio tor). El servicio de Tor ya debe estar ejecutándose y configurado como se vio anteriormente.

La segunda opción es usar el bloque (**GetAddressTor**) en el dispositivo que deseamos tener la dirección Tor, antes de usar este bloque se considera que ya se configuro e inicio el servicio de Tor para SSH como se vio anteriormente en la sección de configuración servicio Tor para SSH.

Empezamos a usar la red “Tor”, el primer paso que hay que considerar es que estaremos usando la red “Tor” a través de un túnel de SSH (Secure Shell) lo que debemos primero ejecutar en línea en una Terminal de Shell ya sea en el Teléfono móvil (Termux), PCs, Tableta o servidor por primera y única vez el siguiente comando desde el dispositivo origen al dispositivo (remoto) destino de la siguiente forma:



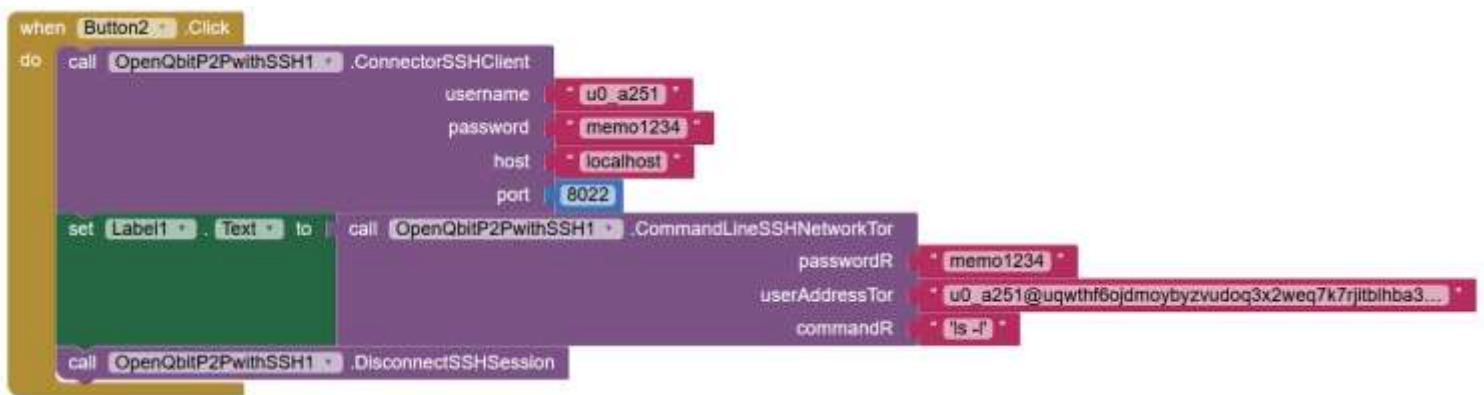
El anterior comando nos creara los certificados de autorizacion del dispositivo destino al remoto, despues de ejecutarlo nos dara el siguiente mensaje que debemos validar y confirmar para crear las llaves del certificado SSH.

Ejemplo:

```
The authenticity of host
'u0_a251@uqwthf6ojdmoybyzvudoq3x2weq7k7rjitblhba3pjbaw2nvjmx3fid.onion
(127.42.42.0)' can't be established. ECDSA key fingerprint is
SHA256:f22LX7WJfLG0iKxP+0+cA/15Q1GsJLFA30ZyMyGLMl4. Are you sure you want
to continue connecting (yes/no)? yes
```

Ahora ya podemos utilizar los bloques de la red "Tor".

Bloque para ejecutar comando SSH a través de red "Tor" (CommandSSHNetworkTor)



Dependencia obligatoria: Usar antes con Bloque (ConnectorSSHClient), Bloque (DisconnectSSHSession).

Parámetros de entrada: **passwordR** <string>, **userAddressTor**<string>, **commandR**<string>

passwordR. - - Es el password del usuario con el que nos conectaremos en el dispositivo remoto (destino).

userAddressTor. - Es la dirección del usuario en formato SSH, un requisito es saber primero cual es la dirección Tor del dispositivo remoto, eso lo podremos obtener de dos formas:

Ejemplo:

u0_a251@uqwthf6ojdmoybyzvudoq3x2weq7k7rjitblhba3pjbaw2nvjmx3fid.onion

Address Tor with user in format ssh: **user@address.onion**

commandR. - Comando que se ejecutara en el dispositivo remoto (destino).

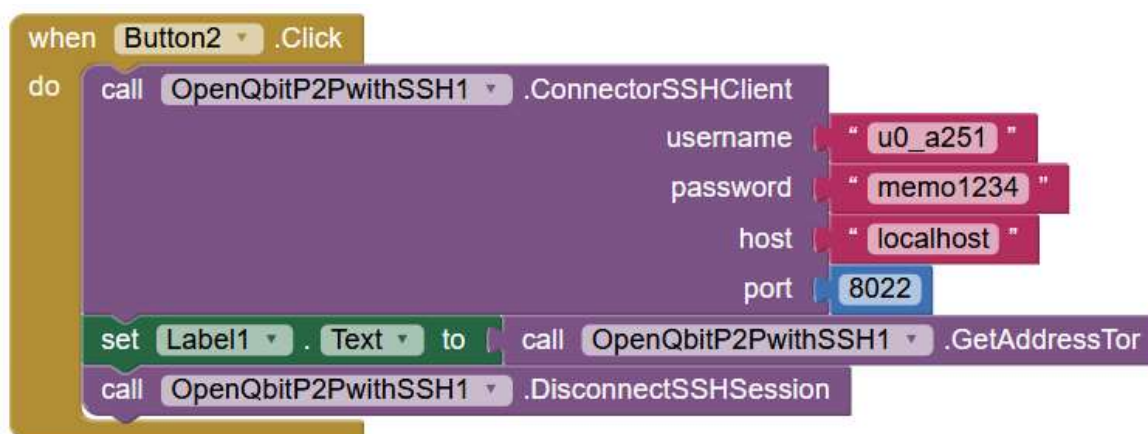
NOTA IMPORTANTE: si el comando tiene varios parámetros se debe introducir con la siguiente sintaxis entre comillas sencillas:

Ejemplo:

'cat HellofromMexico.txt'

Descripción: Bloque para ejecutar cualquier comando en línea a través de SSH mediante comunicación de la red "Tor".

Bloque para obtener la dirección de la red "Tor" local (**GetAddressTor**)



Dependencia obligatoria: Usar antes con Bloque (**ConnectorSSHClient**), Bloque (**DisconnectSSHSession**).

Parámetros de entrada: No aplica – N/A

Parámetros de salida: Nos dara la dirección de la red "Tor" con extension .onion

Ejemplo:

7yfbhadahgvvz25fxysojxd6tf7qd7ryhjsboaga7cd6wo4dfnmylkid.onion

Descripción: Bloque para ejecutar cualquier comando en línea a través de SSH mediante comunicación de la red "Tor".

13. Uso del REst API con API-Key en Syncthing.

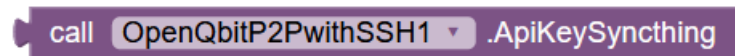
Syncthing cuenta con un REst API para poder usar GET / POST en la administración este es un apoyo adicional a la extensión (**OpenQbitP2PwithSSH**).

Para poder usar la Rest API primero debemos conocer la API-KEY esta nos dará el permiso para ejecutar las instrucciones de GET / POST.

Para saber la API-KEY del teléfono móvil local podemos obtenerla de dos formas.

La primera es usando la extensión (**OpenQbitP2PwithSSH**) con el bloque (**ApiKeySyncthing**).

Bloque para obtener el API-KEY del syncthing local que se esté ejecutando (**ApiKeySyncthing**)



Dependencia obligatoria: Bloque (**ConnectorSSHClient**), Bloque (**DisconnectSSHSession**).

Parámetros de entrada: **Ninguno – N/A**

Parámetros de salida: No entrega el contenido del archivo de configuración en donde se encuentra el API-KEY para ser utilizado con el Rest API de Syncthing.

Ejemplo:

```
[DEFAULT]
name = localhost

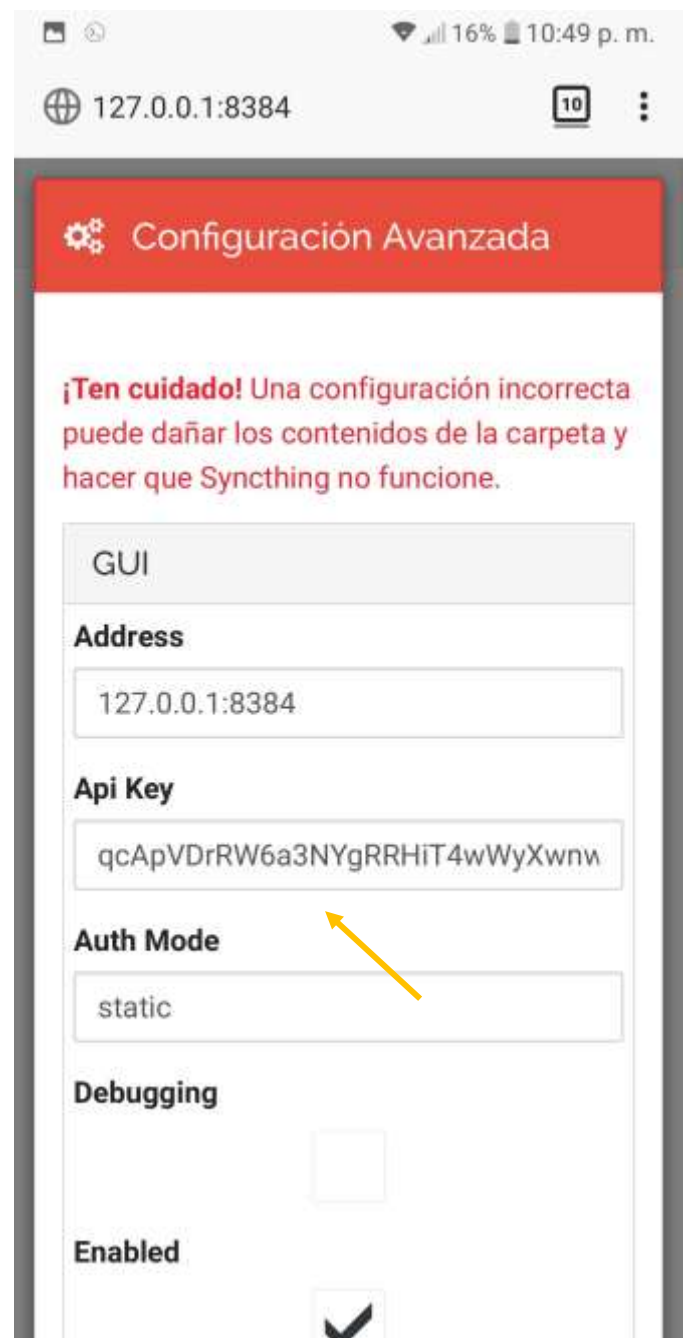
[localhost]
apikey = MafkDvpagX5J6oMzxm9HwDSXJPSQKPFS
hostname = localhost
port = 8384

[remote-device]
apikey = h9mifaKwDq3SSPPmgUuDjsrivFg3dtkK
hostname = some-host
port = 9001
```

Descripción: Bloque de comunicación obtener el contenido del archivo de configuración de syncthing que contiene el API-KEY.

La segunda opción es poder consultarla desde el browser local que tengas en tu teléfono móvil en la dirección por default:

<http://127.0.0.1:8384/>



Ya que se tenga la API-KEY podemos empezar a usar el Rest API de Syncthing nos apoyaremos con una extensión del siguiente sitio web para ejecutar el comando Curl.

<https://puravidaapps.com/extensions.php>

Y Podemos usar la extensión:

[Terminal / Shell Extension](#) by Juan Antonio

Por ejemplo, para obtener el ID del teléfono local podemos ejecutar el siguiente comando de Curl.

```
curl -H "X-API-Key: qcApVDrRW6a3NYgRRHiT4wWyXwnwwqsG"  
http://localhost:8384/rest/stats/device
```



Esta nospermitira ejecutar comandos Curl para usar el Rest API.

Para ver todas las opciones de Syncthing ir a la siguiente referencia de su Rest API.

<https://docs.syncthing.net/dev/rest.html>

Otra herramienta que nos ayudara a la administración es la línea de comandos directos de Syncthing esta a podemos usar con el bloque para ejecutar comandos en línea a través de SSH con el bloque (**CommandLineExecuteSSH**).

Las opciones para la línea de comandos de Syncthing son las siguientes:

```
syncthing [-audit] [-auditfile=<file|-|-->] [-browser-only] [device-id]  
          [-generate=<dir>] [-gui-address=<address>] [-gui-apikey=<key>]  
          [-home=<dir>] [-logfile=<filename>] [-logflags=<flags>]  
          [-no-browser] [-no-console] [-no-restart] [-paths] [-paused]  
          [-reset-database] [-reset-deltas] [-unpaused] [-upgrade]  
          [-upgrade-check] [-upgrade-to=<url>] [-verbose] [-version]
```

Para ver detalles de cada opcion se puede hacer referencia a su sitio:

<https://docs.syncthing.net/users/syncthing.html>

Por ultimo hay que tomar en cuenta que el Rest API y la línea de comandos de Syncthing solo son de apoyo ya que estos dos no cuentan con funciones importantes como dar de alta un ID de un nuevo dispositivo o remover algun ID ya registrado. Es por ese motivo que se creo la extensión (**OpenQbitP2PwithSSH**).

14. Licenciamiento y uso de software.

Android

<https://source.android.com/setup/start/licenses>

Termux

<https://github.com/termux/termux-app/blob/master/LICENSE.md>

Git

<https://git-scm.com/about/free-and-open-source>

Tor Network

<https://github.com/torproject/tor/blob/master/LICENSE>

Syncthing Network

<https://forum.syncthing.net/t/syncthing-is-now-mplv2-licensed/2133>

OpenSSH

<https://www.openssh.com/features.html>

Putty SSH

<https://www.chiark.greenend.org.uk/~sgtatham/putty/licence.html>

MIT App Inventor 2 Companion y App Inventor Blockly

<https://appinventor.mit.edu/about/termsofservice>

Apache Ant

<https://ant.apache.org/license.html>

WGET

<https://www.gnu.org/software/wget/>

Extensiones externas:

JSOINTOOLS

<https://thunkableblocks.blogspot.com/2017/07/jsontools-extension.html>

Licenciamiento versiones opensource y comercial de sistema OpenQbitP2Pconsultar la página oficial <http://www.openqbit.com>

Mini BlocklyChain, MiniBlockly, BlocklyCode, MiniBlockMiniChain, QBlockly son marcas registradas por OpenQbit.

OpenQbitP2P es dominio público.

Todo el código y la documentación en OpenQbitP2P ha sido dedicado al dominio público por los autores. Todos los autores de códigos y representantes de las empresas para las que trabajan han firmado declaraciones juradas dedicando sus contribuciones al dominio público y los originales de esas declaraciones juradas se almacenan en una caja fuerte en las oficinas principales de OpenQbit México.

Cualquier persona es libre de publicar, usar o distribuir las extensiones de OpenQbitP2P (OpenQbit) original, ya sea en forma de código fuente o como binario compilado, para cualquier propósito, comercial o no comercial, y por cualquier medio.

El párrafo anterior se aplica al código y la documentación entregables en OpenQbitP2P, aquellas partes de la biblioteca de OpenQbitP2P que realmente agrupa y envía con una aplicación más grande. Algunos scripts utilizados como parte del proceso de compilación (por ejemplo, los scripts de "configuración" generados por autoconf) pueden estar incluidos en otras licencias de código abierto. Sin embargo, nada de estos scripts de compilación llega a la biblioteca entregable final de OpenQbitP2P, por lo que las licencias asociadas con esos scripts no deberían ser un factor en la evaluación de sus derechos para copiar y usar la biblioteca OpenQbitP2P.

Todo el código entregable en OpenQbitP2P ha sido escrito desde cero. No se ha tomado ningún código de otros proyectos o de Internet abierto. Cada línea de código puede rastrearse hasta su autor original, y todos esos autores tienen dedicates de dominio público en el archivo. Por lo tanto, la base de código OpenQbitP2P es limpia y no está contaminada con código con licencia de otros proyectos de código abierto, no contribución abierta

OpenQbitP2P es de código abierto, lo que significa que puede hacer tantas copias como desee y hacer lo que quiera con esas copias, sin limitación. Pero OpenQbitP2P no es de contribución abierta. Para mantener OpenQbitP2P en el dominio público y garantizar que el código no se contamine con contenido patentado o con licencia, el proyecto no acepta parches de personas desconocidas. Todo el código en OpenQbitP2P es original, ya que ha sido escrito específicamente para su uso por OpenQbitP2P. No se ha copiado ningún código de fuentes desconocidas en Internet.

OpenQbitP2P es de dominio público y no requiere una licencia. Aun así, algunas organizaciones quieren pruebas legales de su derecho a usar OpenQbitP2P. Las circunstancias donde esto ocurre incluyen lo siguiente:

- Su empresa desea indemnización por reclamos de infracción de derechos de autor.
- Está utilizando OpenQbitP2P en una jurisdicción que no reconoce el dominio público.
- Está utilizando de OpenQbitP2P en una jurisdicción que no reconoce el derecho de un autor a dedicar su trabajo al dominio público.
- Desea tener un documento legal tangible como evidencia de que tiene el derecho legal de usar y distribuir de OpenQbitP2P.
- Su departamento legal le dice que debe comprar una licencia.

Si alguna de las circunstancias anteriores se aplica a usted, OpenQbit, la compañía que emplea a todos los desarrolladores de OpenQbitP2P, le venderá una Garantía de título para OpenQbitP2P. Una garantía de título es un documento legal que afirma que los autores

reclamados de OpenQbitP2P son los verdaderos autores, y que los autores tienen el derecho legal de dedicar el OpenQbitP2P al dominio público, y que OpenQbit se defenderá enérgicamente contra los reclamos de licenciamiento. Todos los ingresos de la venta de las garantías de título de OpenQbitP2P se utilizan para financiar la mejora continua y el soporte de OpenQbitP2P.

Código contribuido

Para mantener OpenQbitP2P completamente libre y libre de derechos de autor, el proyecto no acepta parches. Si desea hacer un cambio sugerido e incluir un parche como prueba de concepto, sería genial. Sin embargo, no se ofenda si reescribimos su parche desde cero. El tipo de licenciamiento no-comercial u opensource quien lo use en esta modalidad y alguna similar sin compra de soporte ya sea uso individual o corporativo no importando el tamaño de empresa se registrará por las siguientes premisas legales.

Renuncia de garantía. A menos que lo exija la ley aplicable o acordado por escrito, el Licenciante proporciona el Trabajo (y cada El Colaborador proporciona sus Contribuciones) "TAL CUAL", **SIN GARANTÍAS O CONDICIONES DE NINGÚN TIPO**, ya sea expreso o implícito, incluidas, entre otras, cualquier garantía o condición de TÍTULO, NO INFRACCIÓN, COMERCIALIZACIÓN O APTITUD PARA UN PROPÓSITO PARTICULAR. Usted es el único responsable de determinar el correcto uso o redistribuir el Trabajo y asumir cualquier riesgo asociado con su ejercicio de permisos bajo esta Licencia.

Cualquier pérdida económica o de algún tipo por el uso de este software el afectado no tendrá opción de pago de ningún tipo. Todo litigio legal las partes se someterán a tribunales únicamente en la jurisdicción de la Ciudad de México, país México.

Para soporte, uso y licenciamiento comercial se tendrá que realizar un acuerdo o contrato establecido entre OpenQbit o su corporativo y la parte interesada.

Los términos y condiciones de distribución comercialización pueden cambiar sin previo aviso, dirigirse al portal oficial de www.openqbit.com para ver cualquier modificación de cláusulas de soporte y licenciamiento no-comercial y comercial.

Cualquier persona, usuario, entidad privada, pública de cualquier índole legal o de cualquier parte del mundo quien simplemente use el software acepta sin condicionantes las cláusulas establecidas en este documento y las que se puedan modificar en cualquier momento en el portal de www.openqbit.com sin previo aviso pudiendo ser aplicadas a discreción de OpenQbit en uso no-comercial o comercial.

Cualquier duda e información de OpenQbitP2P dirigirse a la comunidad de App Inventor o a las comunidades de diversos sistemas Blockly como son: AppBuilder, Trunkable, etc. y/o al correo opensource@openqbit.com por la demanda de preguntas puede tardar la respuesta de 3 a 5 días hábiles.

Soporte con uso comercial.

support@openqbit.com

Ventas uso comercial.

sales@openqbit.com

Información legal y preguntas o dudas de licenciamiento.

legal@openqbit.com

