OpenQbit

Peer to Peer

Mobile phone communication through "Tor" network and synchronization of P2P files with "Syncthing

Installation, configuration & administration.

User manual

version 1.0 Beta

August 2020.

Content

1. Introduction.

A **peer-to-peer network or P2P network** is a network of computers in which all or some aspects work without fixed clients or servers, but a series of nodes that behave as equals to each other. That is, they act simultaneously as clients and servers with respect to the other nodes in the network.

P2P networks allow the direct exchange of information, in any format, between mobile phones, tablets, PCs and computers that are interconnected.

Typically these types of networks are implemented as overlay networks built on the application layer of public networks such as the Internet.

The fact that they are used to share and exchange information directly between two or more users has led some users to use them to exchange files whose content gives greater security in sensitive or copyrighted information.

Peer-to-peer networks leverage, manage and optimize the use of bandwidth of other network users through connectivity between them, and thus obtain more performance in connections and transfers than with some conventional centralized methods, where a relatively small number of servers provide the total bandwidth and shared resources for a service or application.

Such networks are useful for a variety of purposes. They are often used to share files of any kind (e.g. audio, video or software). This type of network is also often used in VoIP telephony to make real-time data transmission more efficient.

The efficiency of the nodes in the data link and transmission may vary depending on their local configuration (firewall, NAT, routers, etc.), processing speed, bandwidth availability of your network connection and disk storage capacity.

In our case we will be using the functionalities of two developments with maturity in their use and characteristics in communication, security and diversity for their use.

The first is the "**Tor**" network - Tor (acronym for The Onion Router) is a project whose main objective is the development of a low-latency distributed communications network superimposed on the Internet, in which the routing of messages exchanged between users does not reveal their identity, i.e. their IP address (anonymity at network level) and which, moreover, maintains the integrity and secrecy of the information that travels through it.

More reference in: https://www.torproject.org/

We will use this "Tor" network for any kind of communication between mobile phones whether they are in a Wifi environment or with a mobile Internet service from any phone company worldwide. Later on we will use it to create a secure channel through this network with an SSH (Secure Shell) service. With this combination we will be able to send, distribute, copy, obtain any kind of data.

Another tool or service that we will be for synchronization and / or data replication is Syncthing.

Syncthing. - It is an open source point-to-point file syncing application available for Windows, Mac, Linux, Android, Solaris, Darwin and BSD.

You can synchronize files between devices on a local network or between remote devices over the Internet. Data security is integrated into the software design.

How does it work?

Device discovery is achieved through publicly accessible discovery servers hosted by the project developers, local (LAN) discovery through broadcast messages, device history and static addressing/hosting. The project also provides the Syncthing Discovery Server program to host the discovery servers themselves, which can be used in conjunction with or as a replacement for public servers.

The network of community-contributed relay servers allows devices behind different IPv4 NAT firewalls to communicate by transmitting encrypted data through a third party. The retransmission performed is of a similar nature to the TURN protocol, with TLS-encrypted end-to-end traffic between devices (therefore, even the relay server cannot see the data, only the encrypted sequence). Private relays can also be configured and set up, with or without public relays, if desired. Synchronization will automatically switch from relay to direct device-to-device connections if it finds that a direct connection is available.

Synchronization can be used without any connection to the project or community servers: updates, optional usage data, discovery and relaying can be disabled and/or configured independently, so the mesh and its infrastructure can run in a closed system for privacy or confidentiality.

More information can be found in the reference: https://syncthing.net/

Let's start...

2. Installation and configuration of communication network for mobile phones.

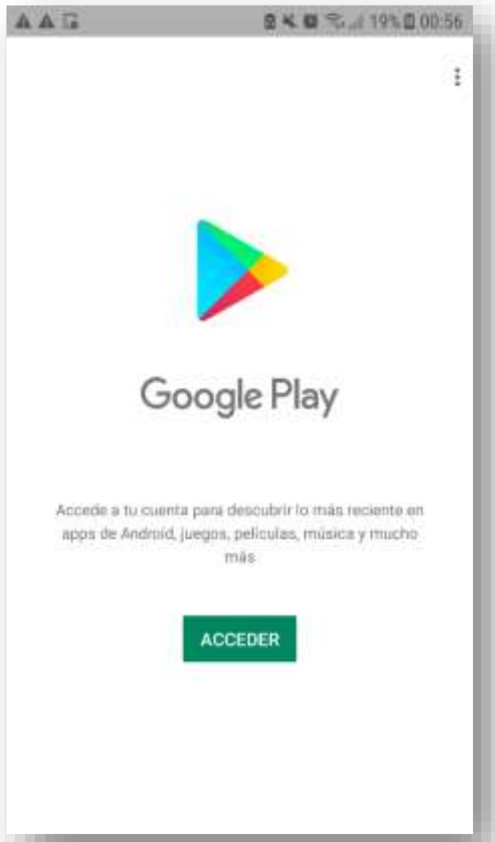
First we need a Linux environment since every Android system is based on Linux for security and flexibility in tools, we will use the "Termux" terminal that contains that environment where we will install the communications network.

Termux is a Linux emulator where we will install the necessary packages to create our communication network between nodes.

One of the main advantages of using Termux is that you can install programs without having to "rotate" the mobile phone (Smartphone). This ensures that no manufacturer's warranty is lost due to this installation.

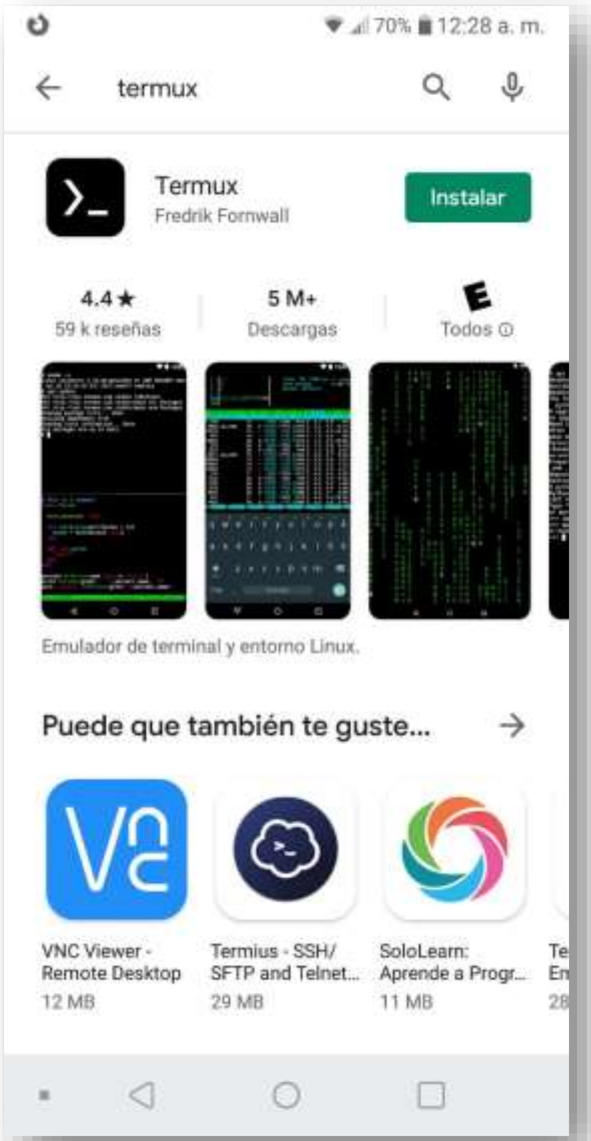
Termux installation.

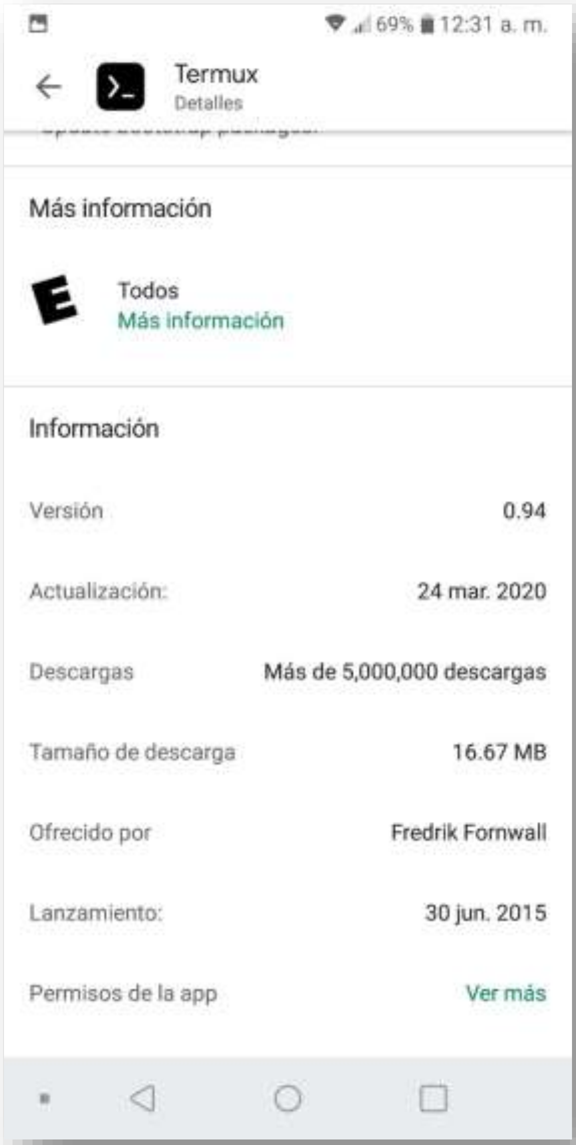
From your mobile, go to the Google Pay icon application (play.google.com).



Accede a tu cuenta para descubrir lo más reciente en apps de Android, juegos, películas, música y mucho más

Search by application "Termux", select it and start the installation process.


70% 12:28 a. m.
termux
Termux
Fredrik Fornwall
Instalar
4.4 ★
59 k reseñas
5 M+
Descargas
Todos ⓘ
Emulador de terminal y entorno Linux.
Puede que también te guste...
VNC Viewer - Remote Desktop
12 MB
Termius - SSH/ SFTP and Telnet...
29 MB
SoloLearn: Aprende a Progr...
11 MB



Información

Versión	0.94
Actualización:	24 mar. 2020
Descargas	Más de 5,000,000 descargas
Tamaño de descarga	16.67 MB
Ofrecido por	Fredrik Fornwall
Lanzamiento:	30 jun. 2015
Permisos de la app	Ver más

Start of the Termux application.

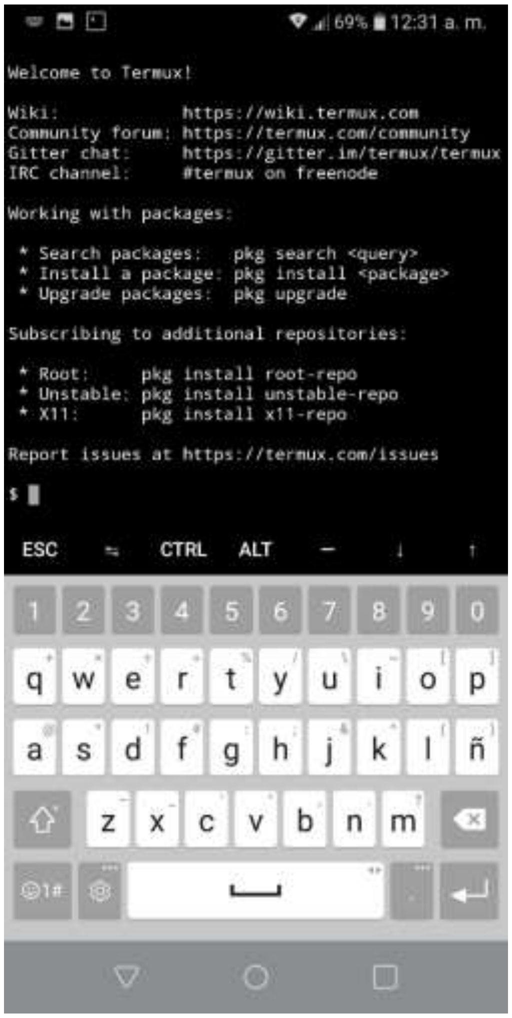
After starting we will have to execute the following two commands to perform updates of the Linux operating system emulator:

$ apt update

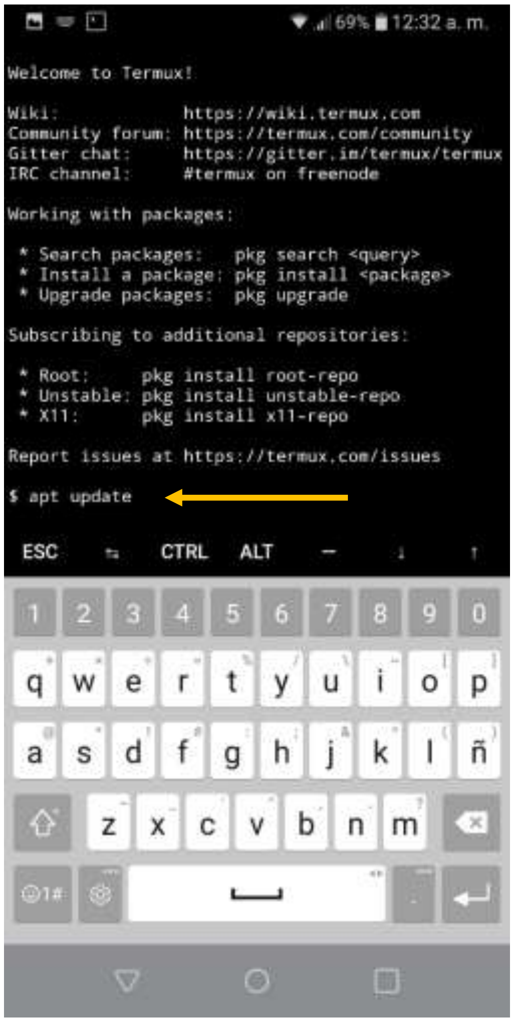
$ apt upgrade

Confirm all options Y(Yes)...

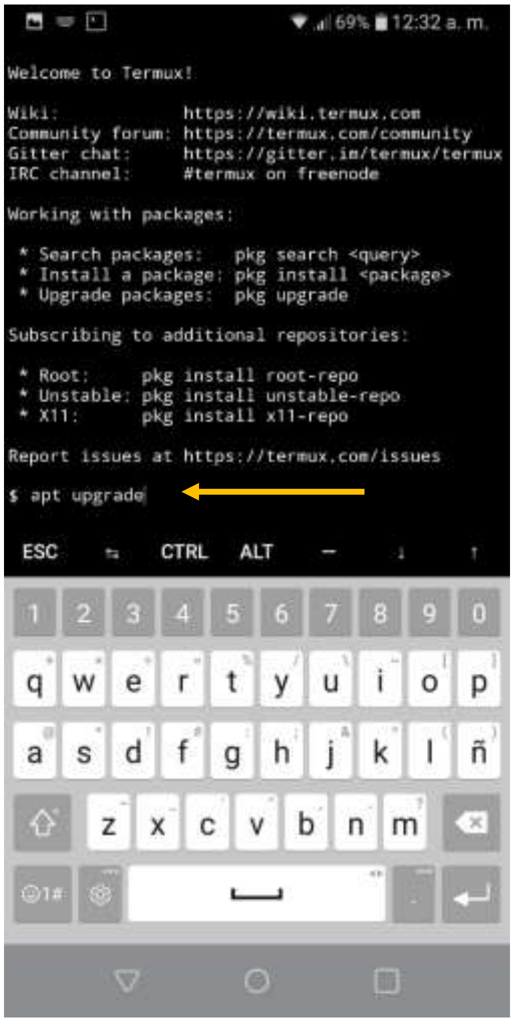
Termux   Home $ apt update $ apt upgrade



```
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

$
```



```
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

$ apt update
```



```
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

$ apt upgrade
```

3. Storage configuration within Termux.

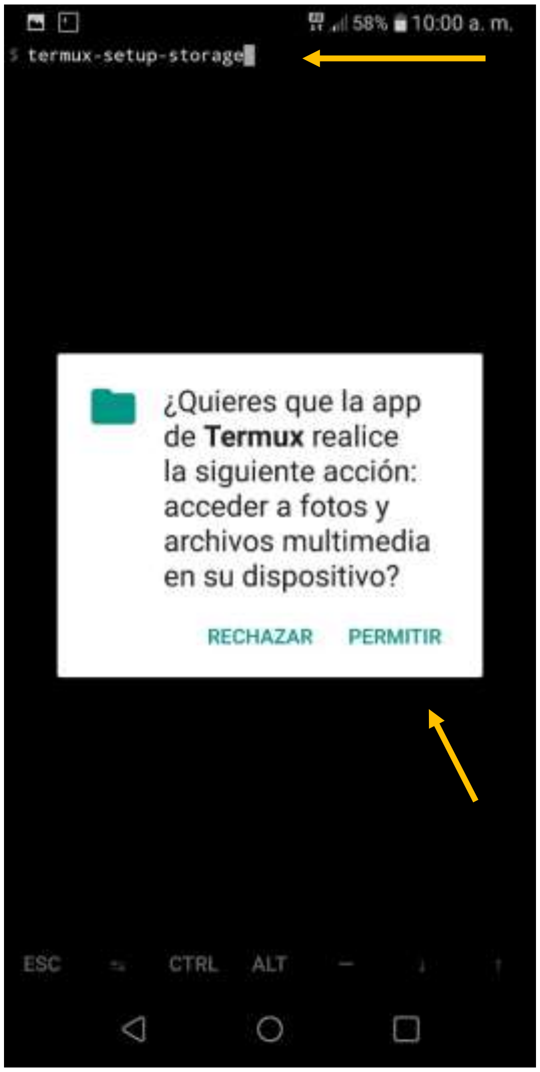
After you have updated and upgraded the Termux system, we will start configuring how to view the internal storage of the phone in the Termux system. This will help you to be able to exchange information between Termux and our information in the phone.

This can be done simply and quickly by running the following command on a Termux terminal.

$ termux-setup-storage

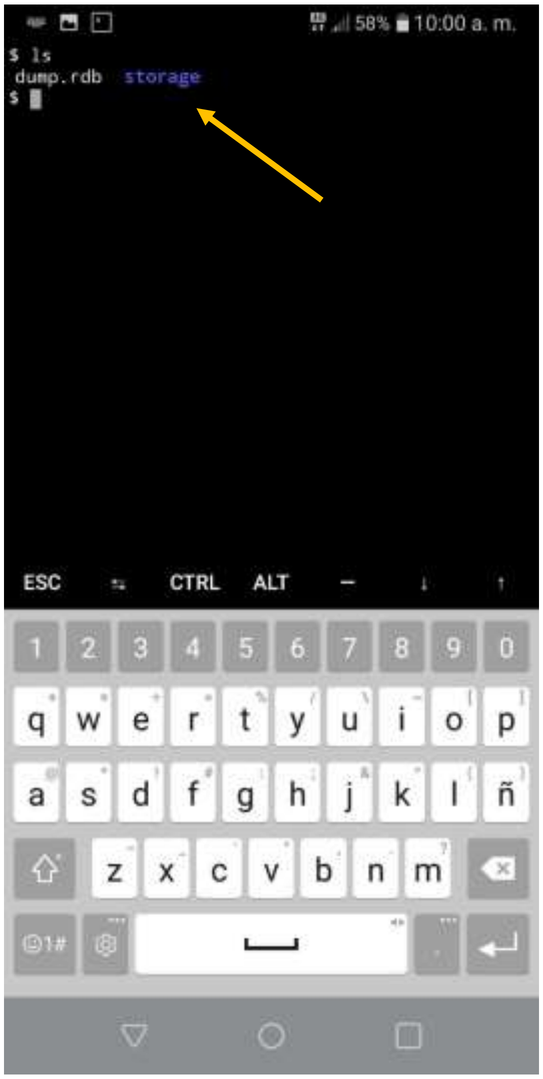
When you execute the previous command, a window appears asking you to confirm the creation of a virtual **storage** (directory) in Termux. We verify by giving the command:

$ ls



```
$ termux-setup-storage
```

¿Quieres que la app de **Termux** realice la siguiente acción: acceder a fotos y archivos multimedia en su dispositivo?



```
$ ls
dump.rdb  storage
$
```

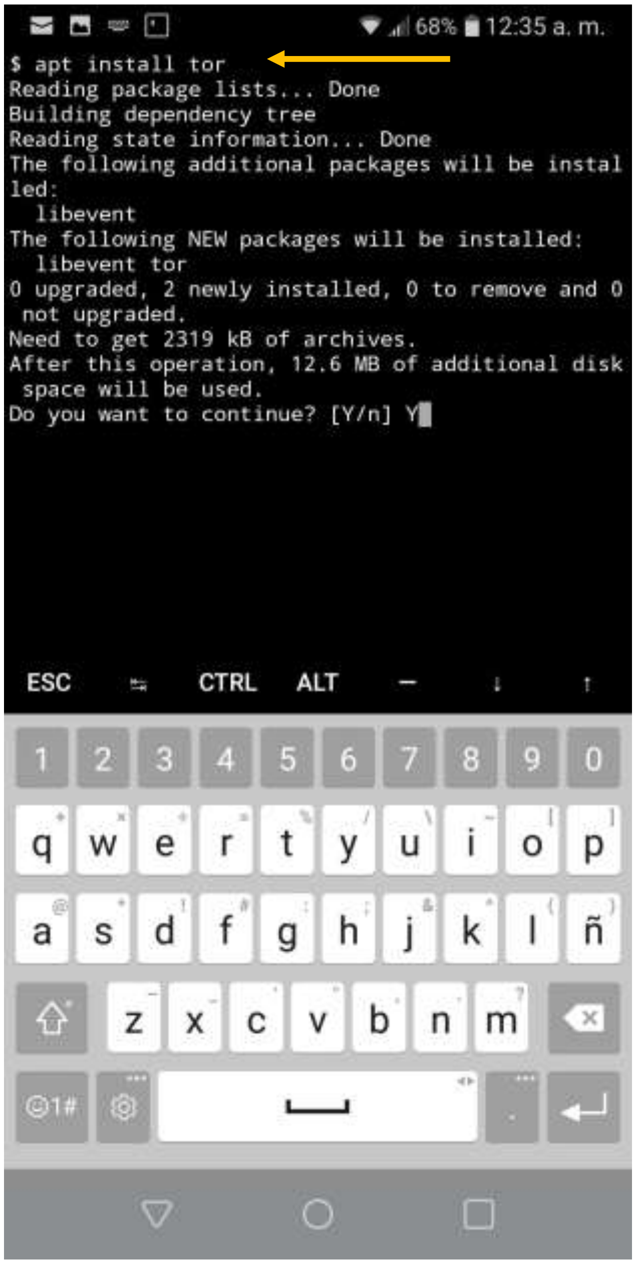
4. “Tor" network installation and "Syncthing" installation.

$ apt install tor

$ apt install synching

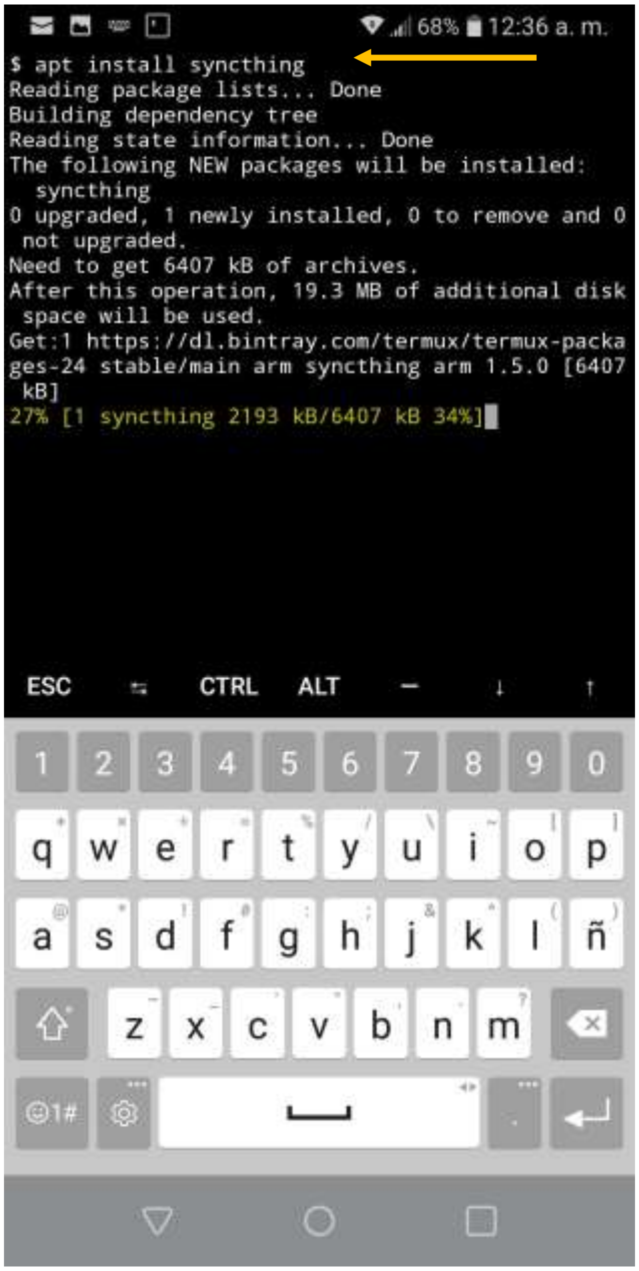
Accept installation by typing capital “Y” in both cases if requested...

$ apt install tor



```
$ apt install tor
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be instal
led:
  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 syncthing



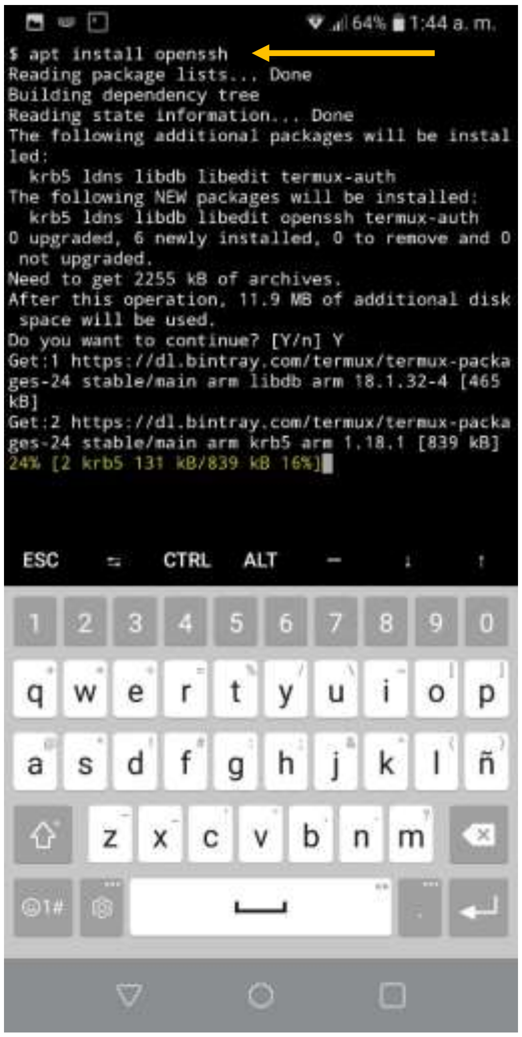
```
$ apt install syncthing
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  syncthing
0 upgraded, 1 newly installed, 0 to remove and 0
 not upgraded.
Need to get 6407 kB of archives.
After this operation, 19.3 MB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

5. SSH (Secure Shell) server.

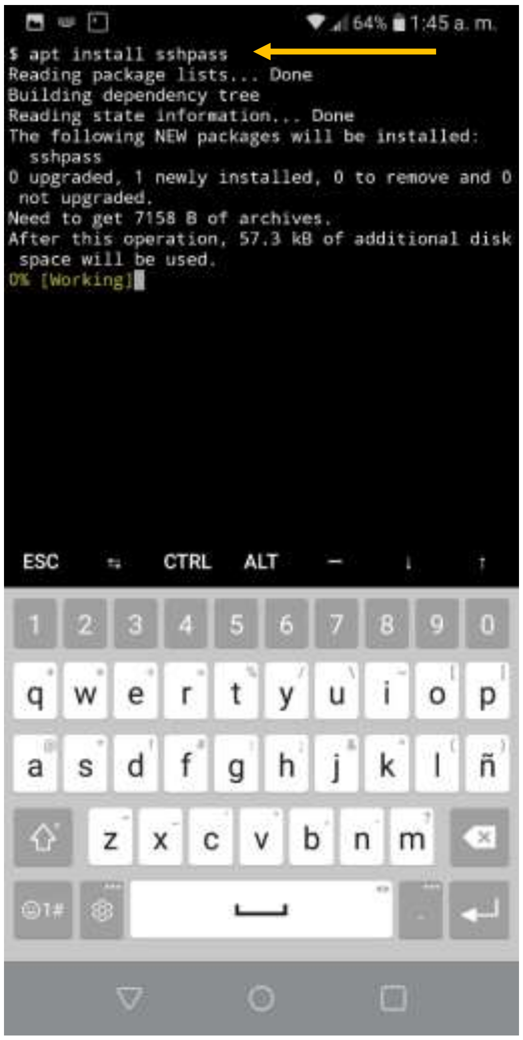
$ apt install openssh

$ apt install sshpass

$ apt install openssh $ apt install sshpass



```
$ apt install openssh
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be instal
led:
  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
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]
```

We have finished with the installation of the communication network, we continue with the configuration of the packages: Tor and Syncthing.

6. SSH server configuration on mobile phone (smartphone).

We will enable the SSH server in the mobile phone to be able to connect from our PC to the mobile phone and work in a faster and more comfortable way, it will also help us to check that the SSH server service in the mobile phone works correctly since we will use it in the communication network to use the P2P service of the Syncthing application.

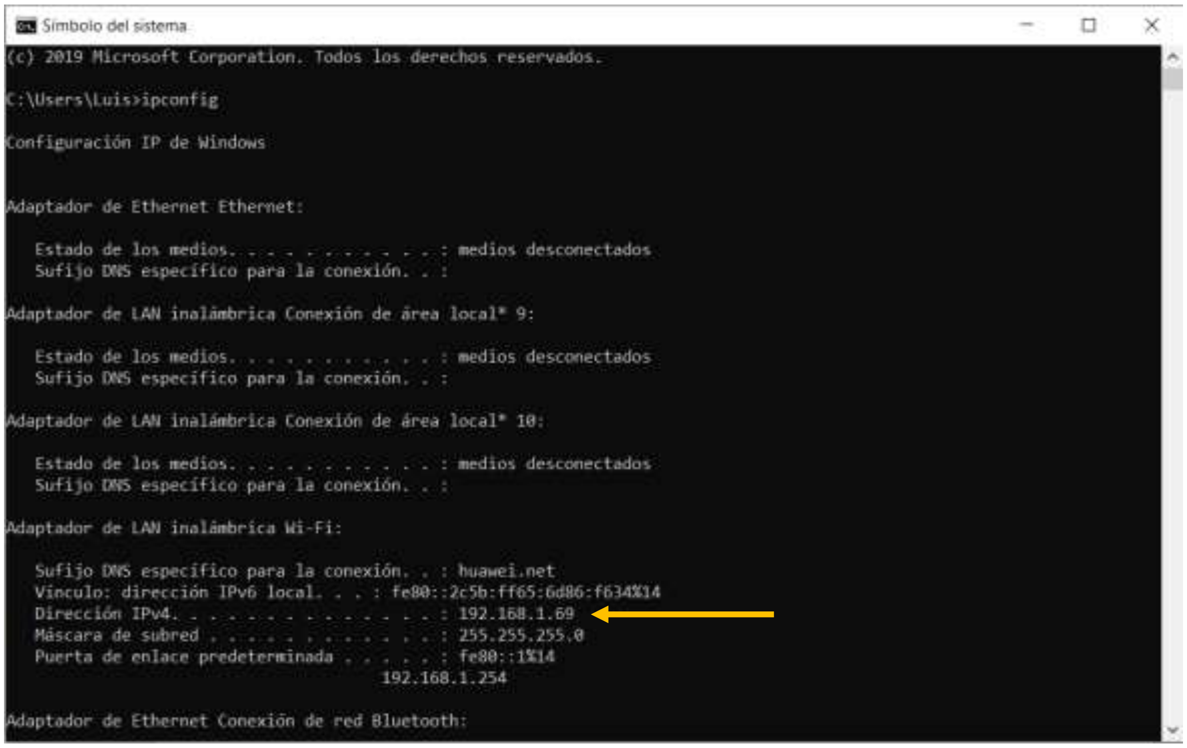
The first thing we have to do is connect the mobile and the PC to the **same WiFi network** so that they can see each other. The IPs or addresses must be similar to 192.168.XXX.XXX the XXX values are variable numbers that are assigned randomly in each computer.

This example was tested on an LG Q6 mobile phone and a PC with Windows 10 Home.

Check the IP or address that the PC has connected to the WiFi we must open a terminal in Windows.

In the bottom panel where the search magnifier is write cmd and press the Enter key. A terminal will open and in it we write the command:

C:User_Name> 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:
```

It will show us the IP assigned to the PC in this is 192.168.1.69 but this is most likely to be different in each case.

NOTE: The address where it says "IPv4 address" should be taken, not to be confused with the Gateway.

Now in the case of the mobile phone in the Termux terminal we must type the following command to know the name of our user that we will use to connect to the SSH server that has our phone, we execute the following command:

```
$ whoami
```

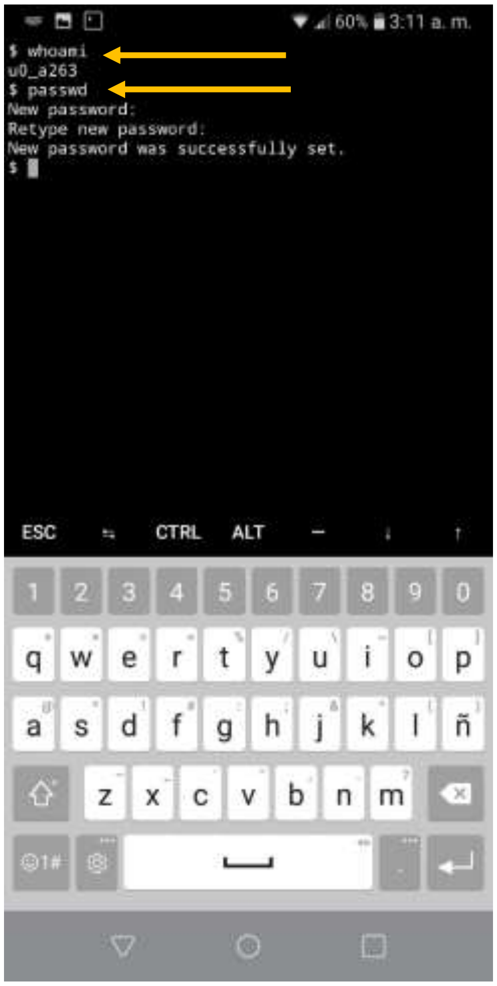
Later we must give a password to this user so we have to execute the following command:

```
$ passwd
```

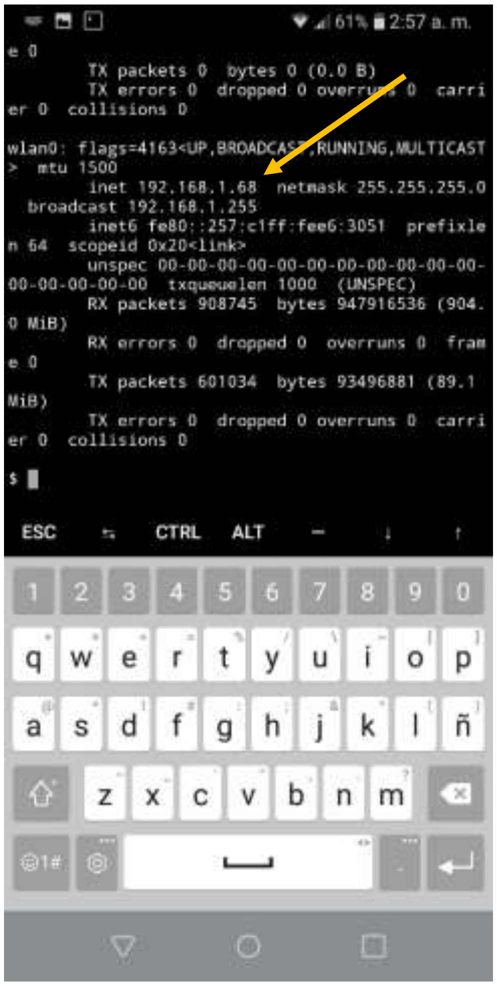
It will ask us to type a password and hit Enter, again it asks us for the password we confirm it and hit Enter, if it has been **successfully "New password was successfully set" in** case of marking an error is possible that the password has not been typed correctly. Perform the procedure again**.**

And then to know what IP we have in Termux we type the following command, the IP is after the word "**inet**":

```
$ ifconfig -a
```



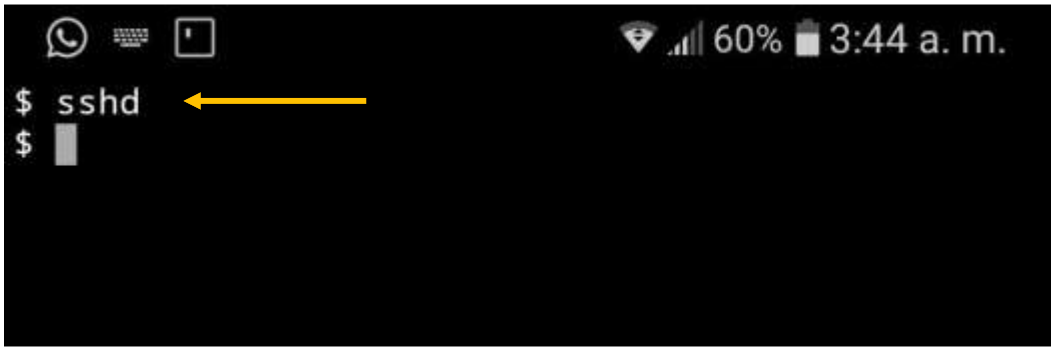
```
$ whoami
u0_a263
$ passwd
New password:
Retype new password:
New password was successfully set.
$
```



```
e 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carri
er 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  prefixle
n 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  fram
e 0
        TX packets 601034  bytes 93496881 (89.1
MiB)
        TX errors 0  dropped 0 overruns 0  carri
er 0  collisions 0

$
```

Now it's time to start the SSH server service on your phone so you can receive sessions from your PC. We execute the following command in the Termux terminal, this command does not give any result.

$ sshd

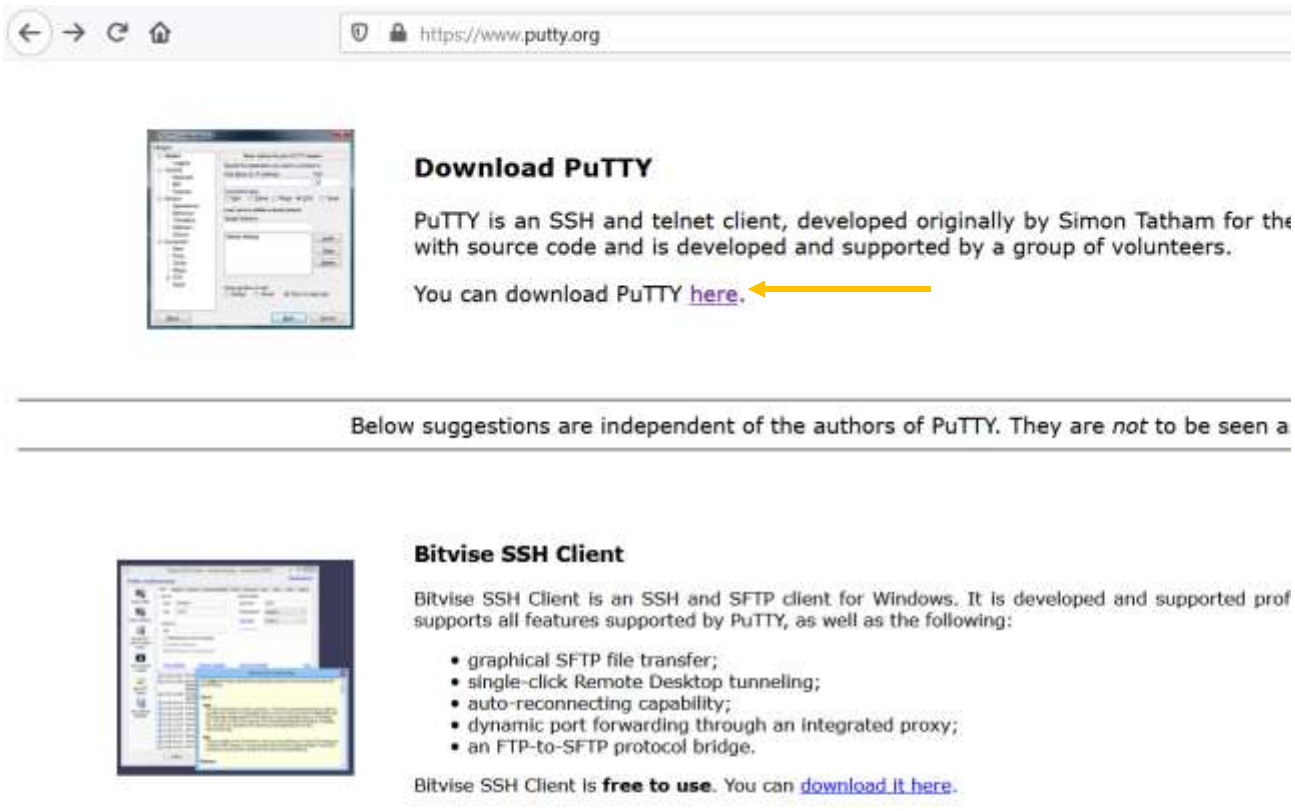


```
$ sshd
$
```

Now we will have to install a program on the PC that will communicate with the phone's SSH server from the PC.

We have to go to https://www.putty.org

Select where the link "You can download PuTTY here" is



Download PuTTY

PuTTY is an SSH and telnet client, developed originally by Simon Tatham for the
with source code and is developed and supported by a group of volunteers.

You can download PuTTY here.

Below suggestions are independent of the authors of PuTTY. They are *not* to be seen a

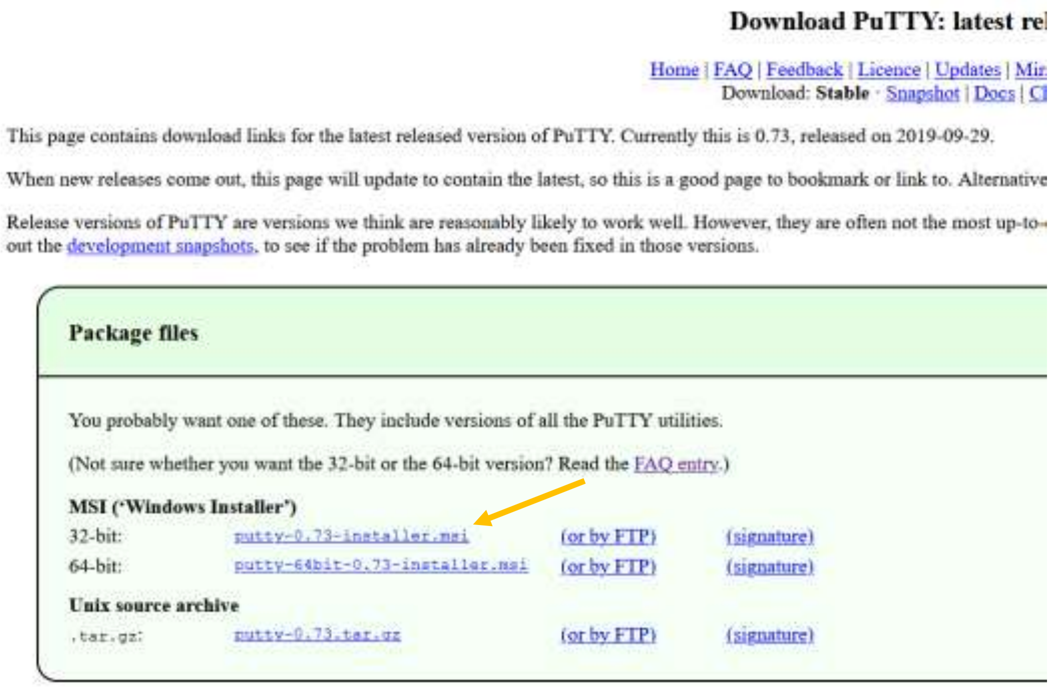
Bitvise SSH Client

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.

Choose the 32-bit version, it doesn't matter if your system is 64-bit will work fine.



Download PuTTY: latest rel

Home | FAQ | Feedback | Licence | Updates | Mir
Download: **Stable** · Snapshot | Docs | Cl

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-
out 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.)

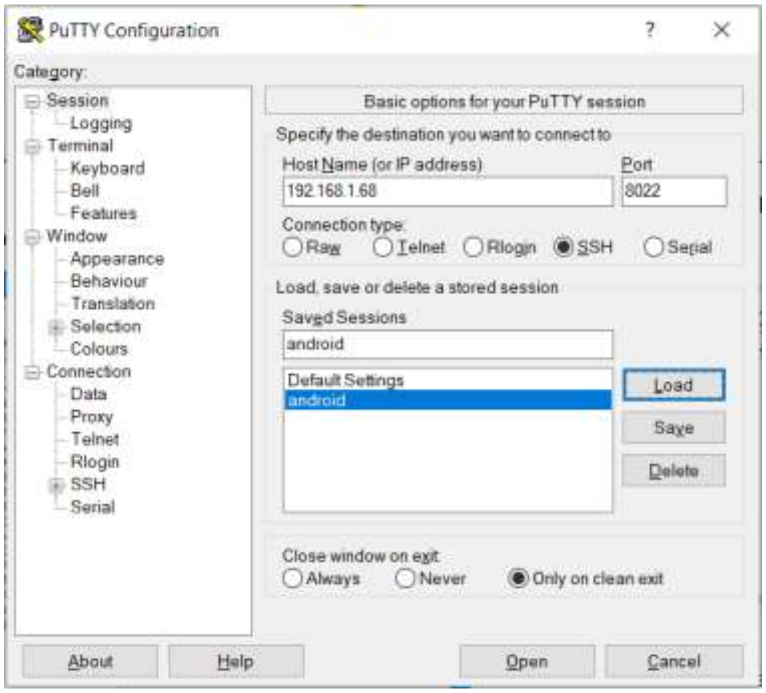
MSI ('Windows Installer')

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)

Once it has been downloaded to your PC, run it and install it with the default options. Then start the PuTTY application.



Host Name (or IP address): 192.168.1.68
Port: 8022
Connection type: Raw Telnet Rlogin [✓ SSH] Serial
Saved Sessions: android
Saved sessions list: Default Settings, android
Close window on exit: Always Never [✓ Only on clean exit]

In this session we will enter the data from our Openssh server that we installed in the mobile phone.

Enter the IP of the mobile phone.

HostName or IP address:

192.168.1.68 (IP example)

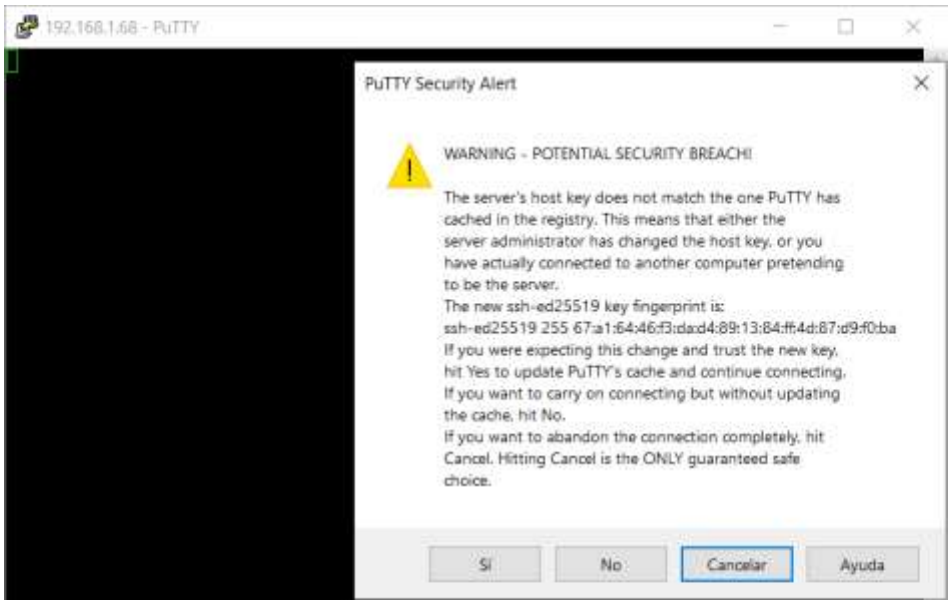
Port:

8022 (Default port of the mobile SSH server).

We can give a name to the session in "Saved Sessions" and click on the Save button.

Later on in the lower part we press to open a connection to the server giving the button "Open".

On the PC when you connect for the first **time you will be asked for confirmation of** the information encryption key by clicking on the "Yes" button.

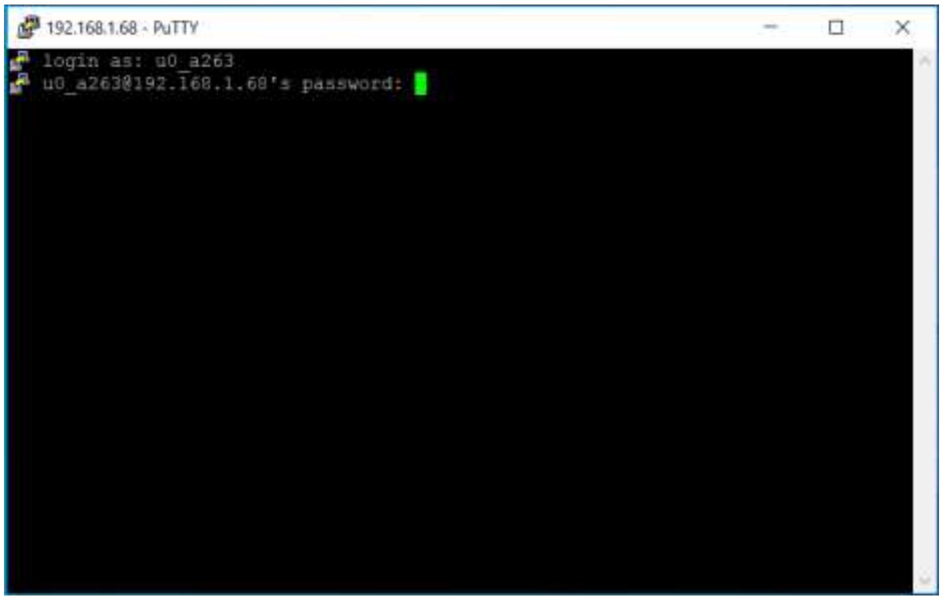


WARNING - POTENTIAL SECURITY BREACH!

The server's host key does not match the one PuTTY has cached in the registry. This means that either the server administrator has changed the host key, or you have actually connected to another computer pretending to be the server.
The new ssh-ed25519 key fingerprint is:
ssh-ed25519 255 67:a1:64:46:f3:da:d4:89:13:84:ff:4d:87:d9:f0:ba
If you were expecting this change and trust the new key, hit Yes to update PuTTY's cache and continue connecting.
If you want to carry on connecting but without updating the cache, hit No.
If you want to abandon the connection completely, hit Cancel. Hitting Cancel is the ONLY guaranteed safe choice.

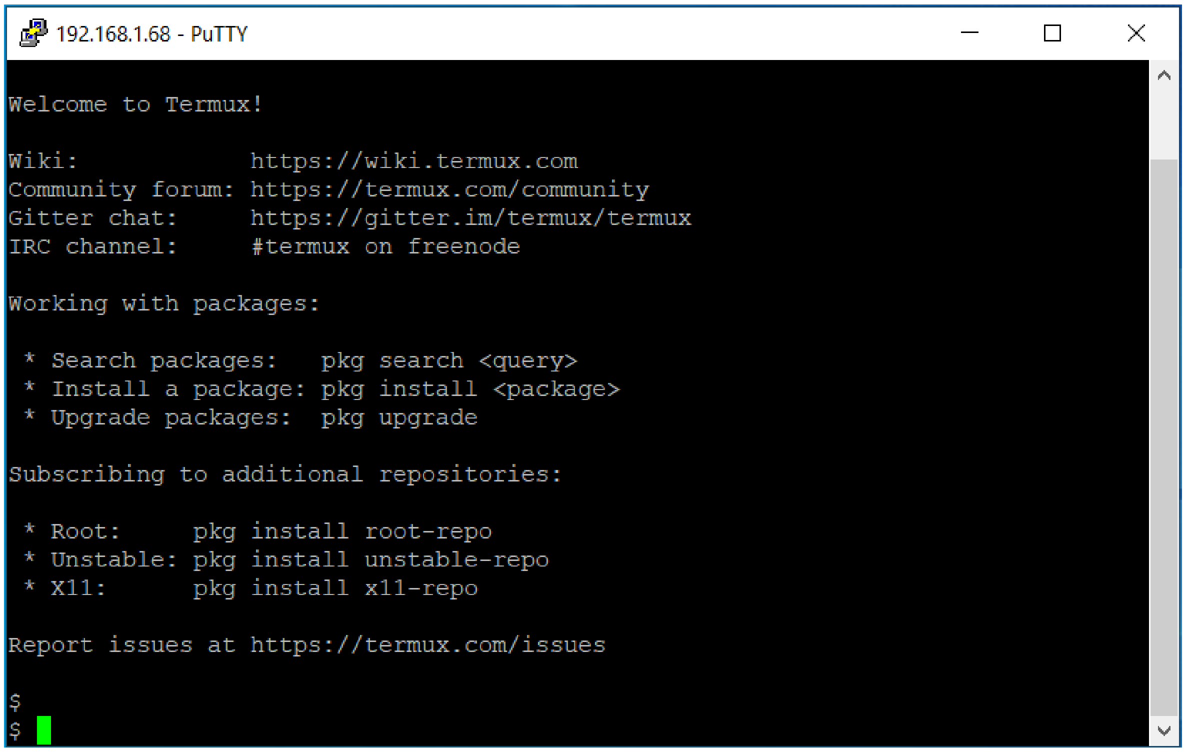
Later on, we will be asked for the user we are going to connect with. We will use the information that we previously obtained (user and password).

In the **Login as:** we must enter our user and give Enter, then we will ask for the password again give the Enter button.



```
login as: u0_a263
u0_a263@192.168.1.68's password:
```

If the data was correct, we will be in an SSH (Secure Shell) session performed from the PC (Client) on the phone (SSH Server).



```
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

$
$ 
```

IMPORTANT NOTE: Remember that the IP (address) of the PC and the IP (address) of the mobile phone connected to the same WiFi will probably be changing every time we disconnect and reconnect so we have to double check what addresses each device has, this will ensure the success of the connection between devices through the phone's SSH server and PC (Client).

So far we've only been able to connect to the same WiFi network, but if we move our phone outside the same network as the PC, we won't be able to connect because there are different networks involved in going through other more complicated communication devices. We'll solve this when we set up the "Tor" network.

Remember that this connection is made only to verify the service of the server we installed on the phone and to have a more comfortable working environment with a remote session from a PC to the phone.

7. “Tor" network configuration with SSH (Secure Shell) service.

With the remote session from the PC we will start configuring the "Tor" network with the SSH service enabled.

The importance of having the "Tor" network is to give the devices the ability to communicate anywhere in the world over the Internet without being on the same WiFi network, no matter where we are we can connect and form the network between nodes.

The "Tor" network has a lot of flexibility in its configuration since it has several parameters in its configuration file "torrc" this file is in the path (`$PREFIX/etc/tor/torrc`) in our case with the Termux session from our PC we will know by typing the following command:

$ echo $PREFIX

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

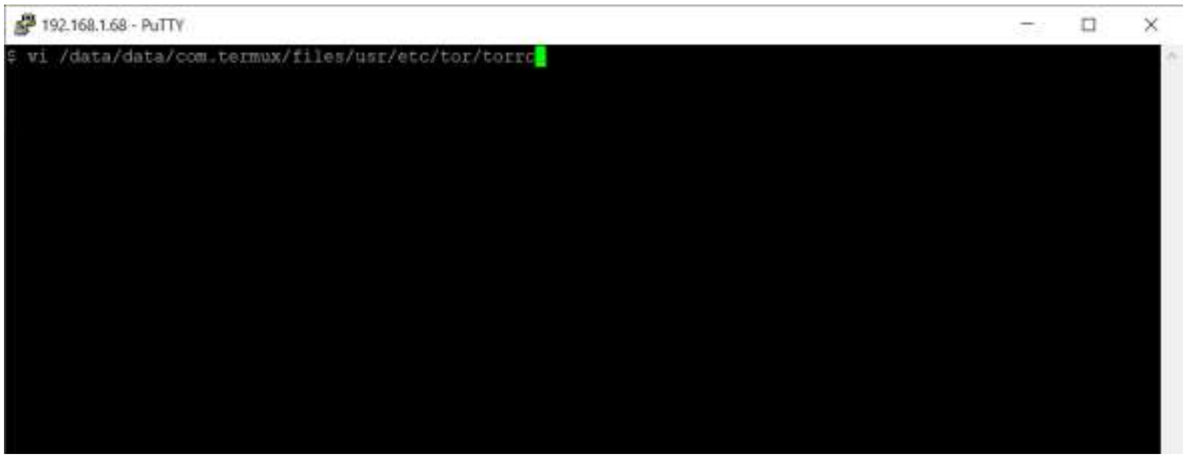
Therefore the file we need to edit will be in the path:

PREFIX/etc/tor/torrc which is equal to /data/data/com.termux/files/usr/etc/tor/torrc

We proceed to edit the configuration file using the command line editor "vi" by executing the following command:

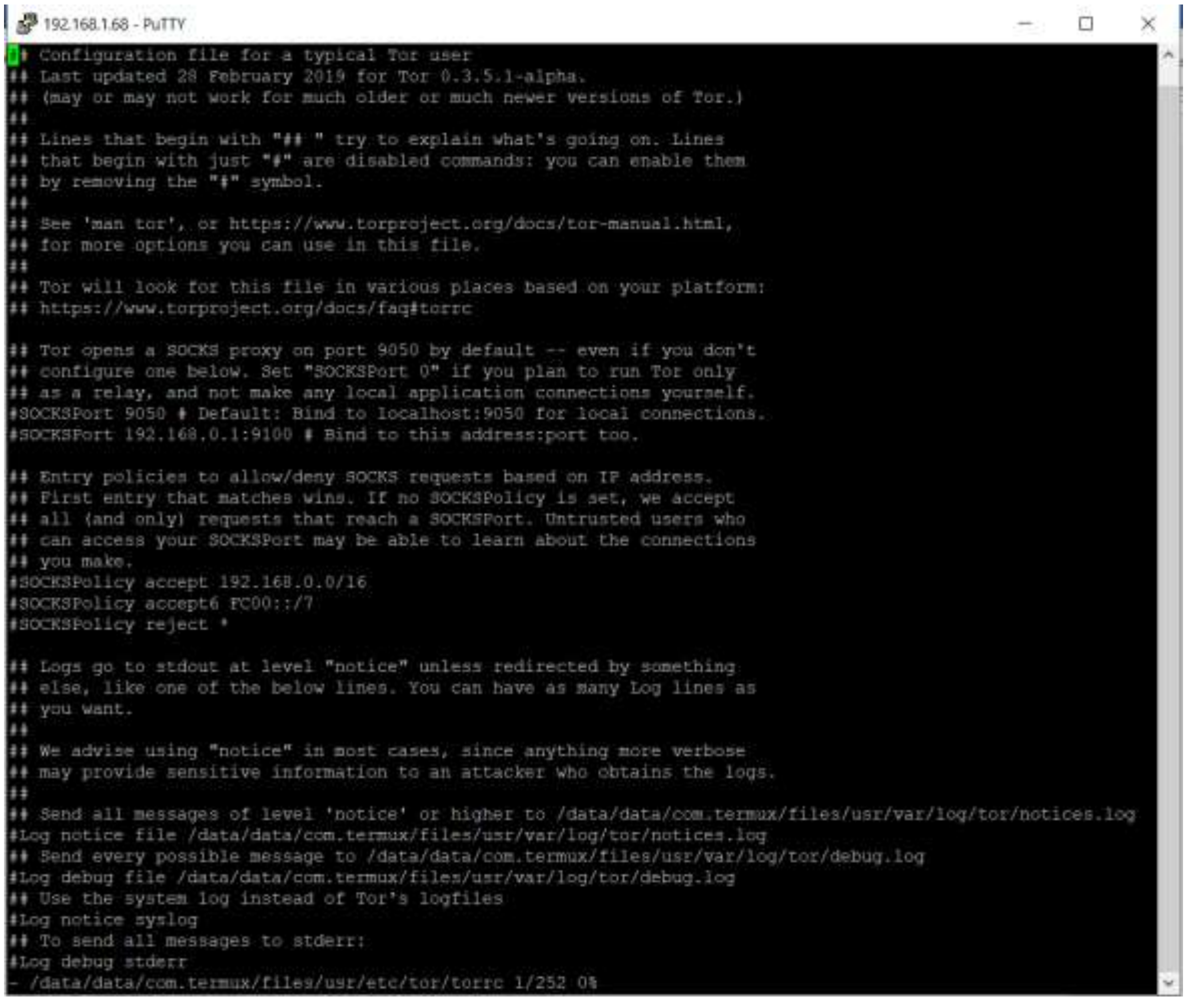
vi /data/data/com.termux/files/usr/etc/tor/torrc

He will edit that file for us, as an example:



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

Edited **"torrc"** file:



```
## 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%
```

In this "torrc" file we will have to add or use the lines that the file has by making the following changes, three lines that are the following:

Syntax: SOCKSPort <application port number>

Example: SOCKSPort 9050

The SOCKSPort variable tells us that this communication socket over the TCP-IP protocol will use the mobile device (phone) by default and port 9050 will be opened or in use.

Syntax: HiddenServiceDir <Directory where the application's configuration will be saved>

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

The HiddenServiceDir variable tells us that this will be the directory where the configuration of the service that will be used through the Tor network will be stored. In this directory you will find the configuration and security file for the service, and in this directory you will find a file called **hostname.**

We can see the address assigned by the Tor network for the specific service created in our case is creating an SSH service that will be implemented on the Tor network, the directory we're naming as **hidden_ssh** will contain the **hostname** file. This directory does not need to be created, because when we start the Tor network service it will be created automatically.

To see the address provided by the Tor network, we can use the "more" command. Before we use this command, we must finish configuring the "torrc" file and register the Tor network service so that the **hidden_ssh** directory and the files inside it are created, as is the case with the **hostname** file.

$ more ~/.tor/hidden_ssh/hostname (note that the tor home directory is hidden and a "." must be used before the tor directory name). The Tor service should already be running and configured as seen above.

The above command will result in an address with an alphanumeric string with an extension **. onion** similar to :

uqwthf6ojdmoybyzvudoq3x2weq7k7rjitblhba3pjbawk2nvjmx3wer.onion

Finally we need to add the **HiddenServicePort** variable to the "torrc" configuration file. This parameter tells the Tor network which port the application we're signing up for will use over the Tor network.

```
Syntax: HiddenServicePort <Port of departure> <Local or public IP>: <Port of departure>

        O

        HiddenServicePort <Port of departure>
```

Examples:

```
       HiddenServicePort 22 127.0.0.1:8022

      O

        HiddenServicePort 8022
```

With the above we have completed the configuration of the Tor network for generic services and SSH (Secure Shell) service. This service will be used to communicate and send any type of data or files such as updating a database such as SQLite or Mysql or Progress or another one installed on our mobile phone.

We continue with the configuration of the P2P communications network for synchronization and data replication with Syncthing.

8. Peer to Peer system configuration with manual syncthing.

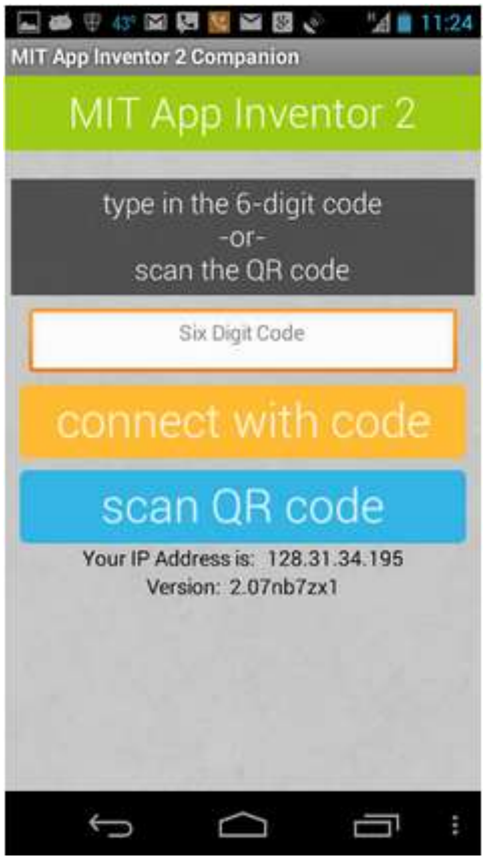
A "Peer to Peer" or P2P architecture is fundamental in a point to point technology system since this architecture gives two central points in the system communication processes, one is to provide the same updated (synchronized) information at any time in all the nodes no matter if the nodes are in a private network (Wifi) or a public network (internet) or a hybrid of these two and a second point of this type of architecture is that they do not depend on an intermediary (server) to transfer, update or consult information between nodes. All this is due to the fact that communication is carried out directly between the nodes, in our case the P2P network is carried out directly between the telephones that form the network without an intermediary.

For this task we will use the opensource Syncthing tool that works based on a "Peer to Peer" architecture.

The configuration of Syncthing for devices (mobile phones) will be seen manually and automatically. The manual form is applied in synchronization with a maximum of 5 nodes, the nodes can be mobile phones, PCs, servers or tablets, in case of having a large number of volume the optimal is the automatic configuration, the process is focused on the registration of the nodes with which we want to perform the synchronization of the selected information.

The requirements to start with are:

1. Have Tor network service started (optional).
2. (optional - recommended) Have installed an application that can read QR codes, we suggest the App inventor Android application that is easy and simple to install from Google Play.
3. To have initiated the service of Synthing.



We show the App Inventor application that we will use for reading QR codes that will serve us in the future for the registration of nodes in Syncthing.

The following example was made between two mobile devices (phones) using the following models:

Mobile device #1: Model LG Q6

Mobile device #2: Sansumg model

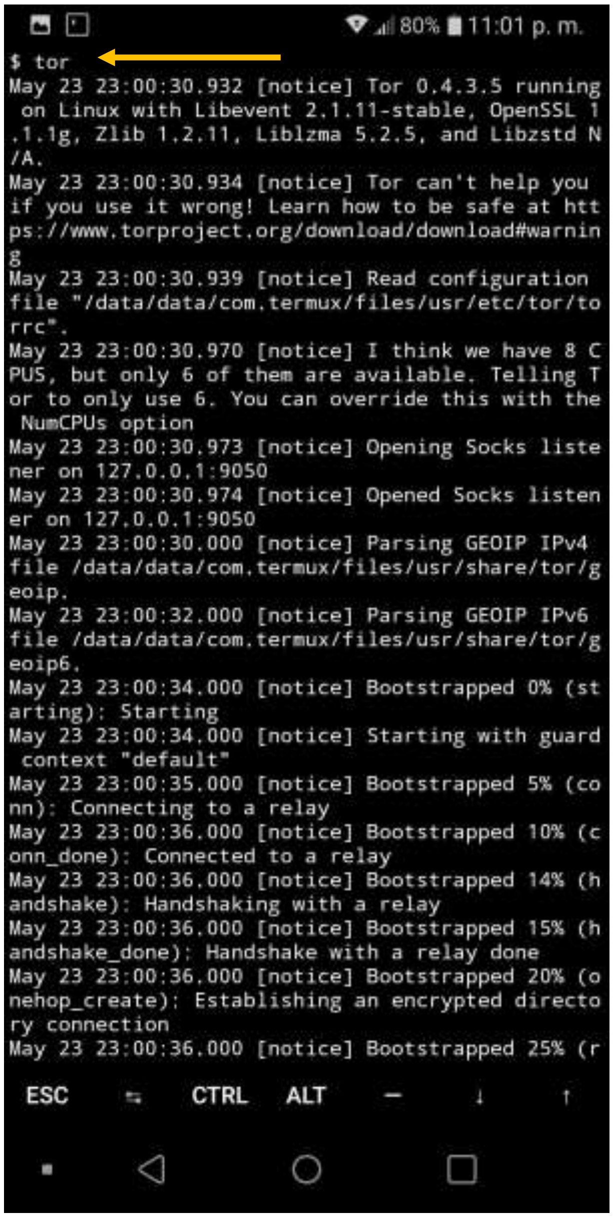
We'll start with starting Tor network services and syncthing services using the following commands, open two terminals inside Termux, and run each command separately:

```
$ tor
```

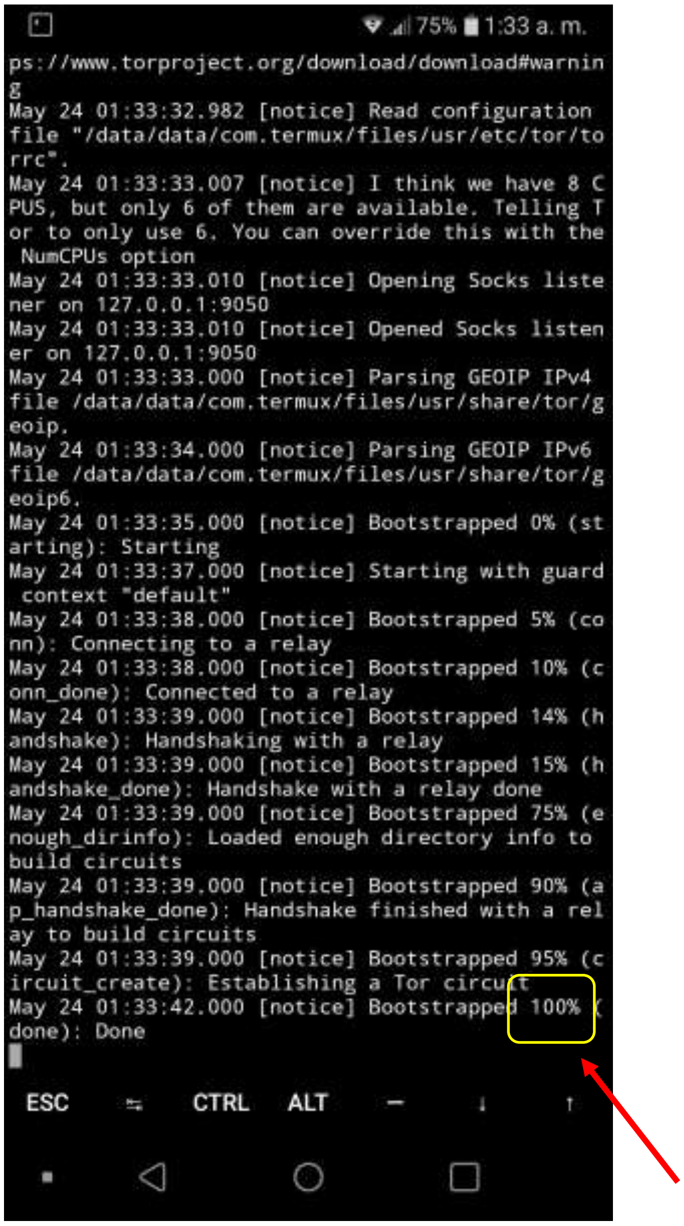
After you have typed the Tor network program startup command, you should check that the execution was successful, by checking that it was done 100%.

Running the Tor program on a Termux terminal, we verify that it is running at 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
```

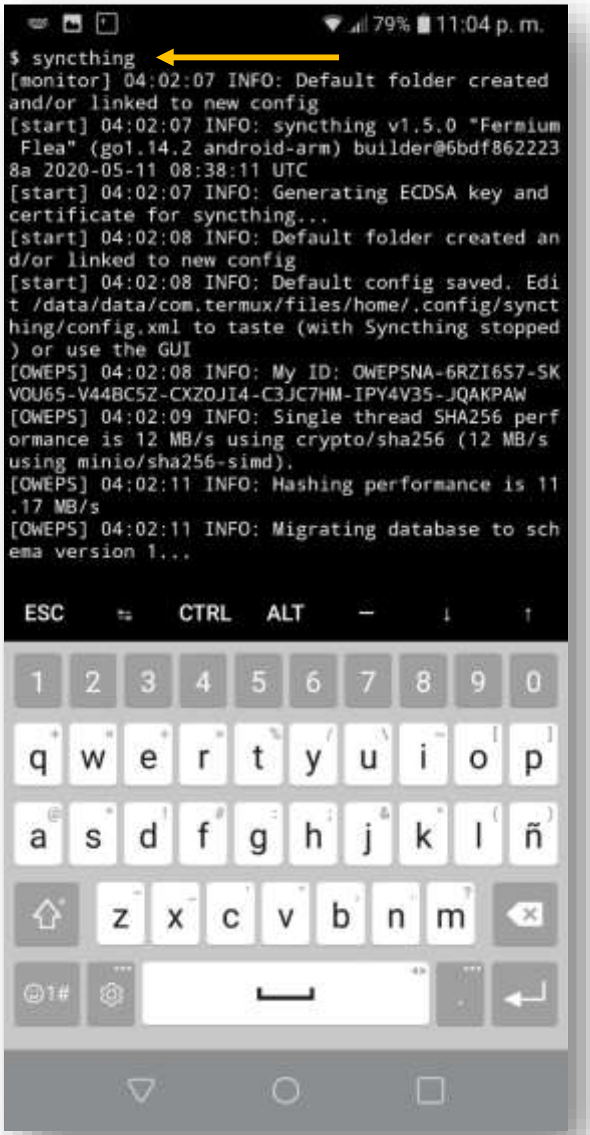
Then we execute the syncthing command:

$ syncthing

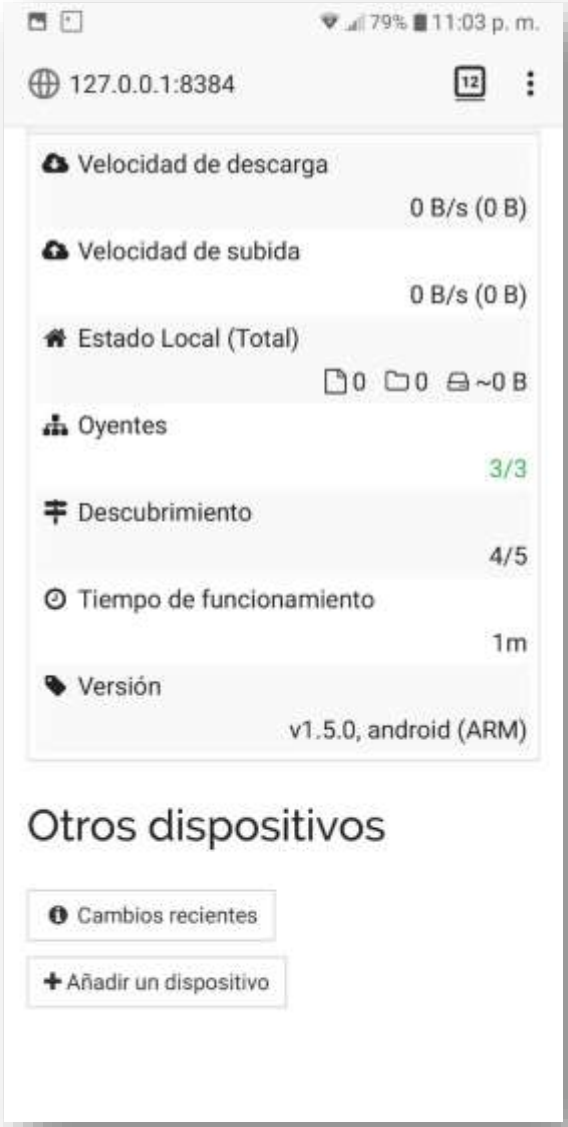
After executing this command it will open an administration page in the browser of our phone if it doesn't open automatically, we can go to any browser that we have installed where we normally navigate on the internet and we can put the next one:

http://127.0.0.1:8384

It will open the administration screen of the tool that will help us to synchronize our information between all the nodes (phones) of the system.



```
$ syncthing
[monitor] 04:02:07 INFO: Default folder created
and/or linked to new config
[start] 04:02:07 INFO: syncthing v1.5.0 "Fermium
 Flea" (go1.14.2 android-arm) builder@6bdf862223
8a 2020-05-11 08:38:11 UTC
[start] 04:02:07 INFO: Generating ECDSA key and
certificate for syncthing...
[start] 04:02:08 INFO: Default folder created an
d/or linked to new config
[start] 04:02:08 INFO: Default config saved. Edi
t /data/data/com.termux/files/home/.config/synct
hing/config.xml to taste (with Syncthing stopped
) or use the GUI
[OWEPS] 04:02:08 INFO: My ID: OWEPSNA-6RZI6S7-SK
VOU6S-V44BC5Z-CXZOJI4-C3JC7HM-IPY4V35-JQAKPAW
[OWEPS] 04:02:09 INFO: Single thread SHA256 perf
ormance is 12 MB/s using crypto/sha256 (12 MB/s
using minio/sha256-simd).
[OWEPS] 04:02:11 INFO: Hashing performance is 11
.17 MB/s
[OWEPS] 04:02:11 INFO: Migrating database to sch
ema version 1...
```



127.0.0.1:8384

Velocidad de descarga 0 B/s (0 B)
Velocidad de subida 0 B/s (0 B)
Estado Local (Total) 0 0 ~0 B
Oyentes 3/3
Descubrimiento 4/5
Tiempo de funcionamiento 1m
Versión v1.5.0, android (ARM)

Otros dispositivos

Since we have the "syncthing" administration screen open, we proceed to register the node or nodes that we want to synchronize the information. At this point is where we will occupy the program that reads QR codes.

The syncthing program when it starts for the first time creates a unique identifier of the phone that is composed of a group of eight sets of alphanumeric characters in capital letters, this identifier (ID) is the one that we will register in the node or nodes that we want to synchronize the information.

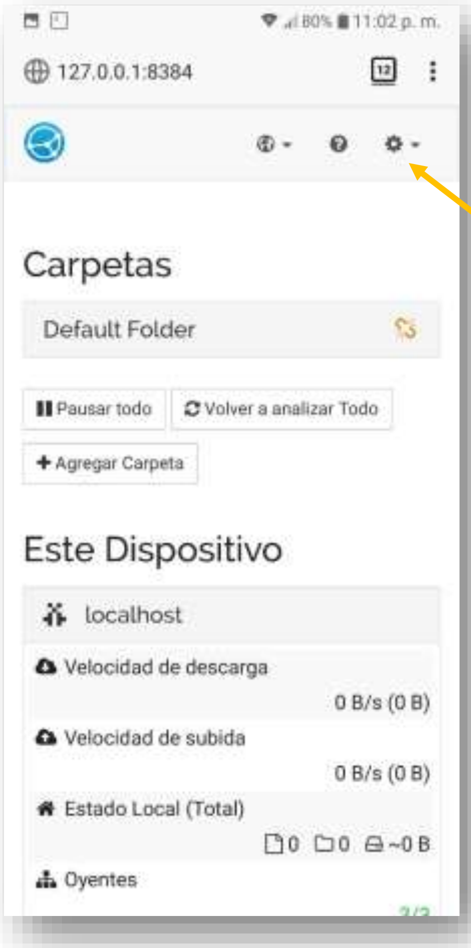
In our case the LG Q6 phone ID will have to be registered to the Sansumg phone and the Sansumg phone ID will have to be registered to the LG Q6. They must be in both phones in order to work properly.

We will perform the steps of the Sansumg mobile phone registration on the LG Q6 phone.

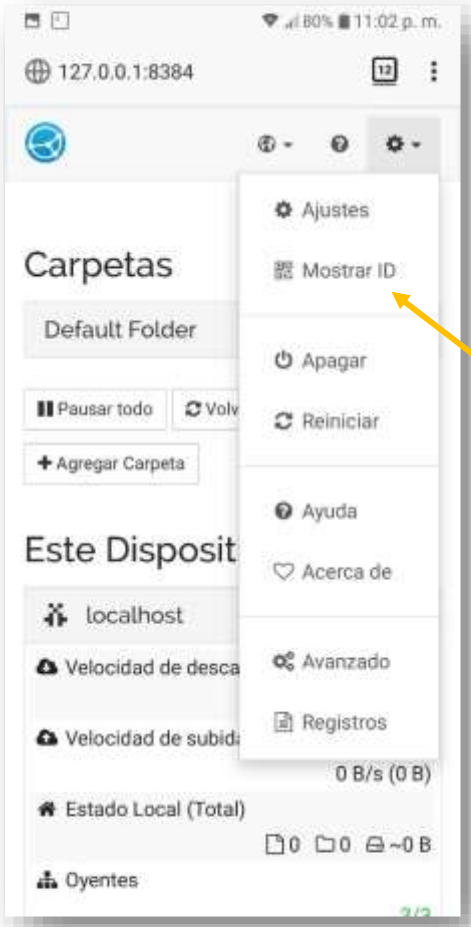
First (1) on the top of the administration screen (internet browser) of the Sansumg phone with syncthing we will click on the menu tab on the top right side.

Second (2) step we will see a menu, in this we click on "Show ID" of the Sansumg.

(1)



(2)

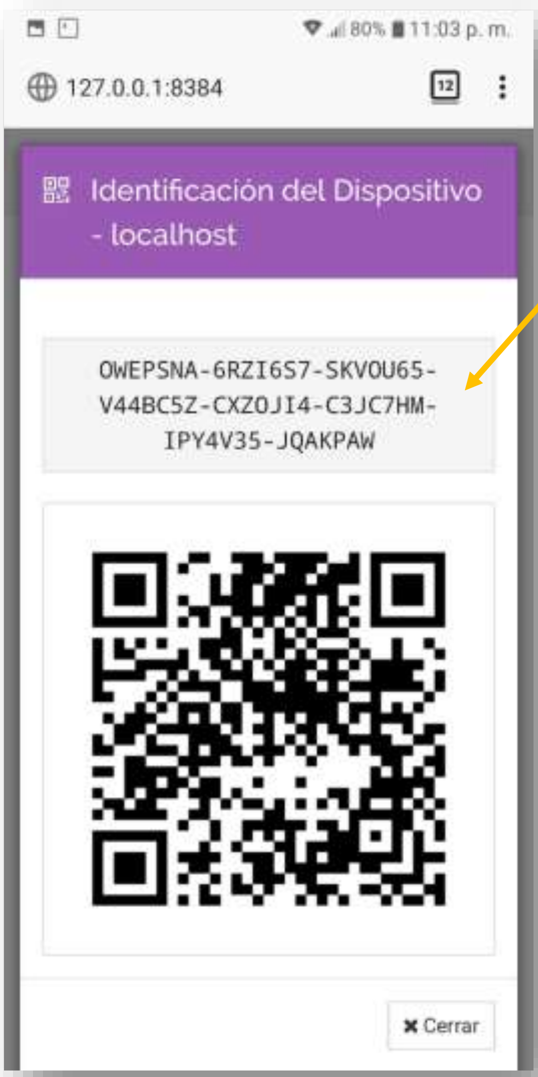


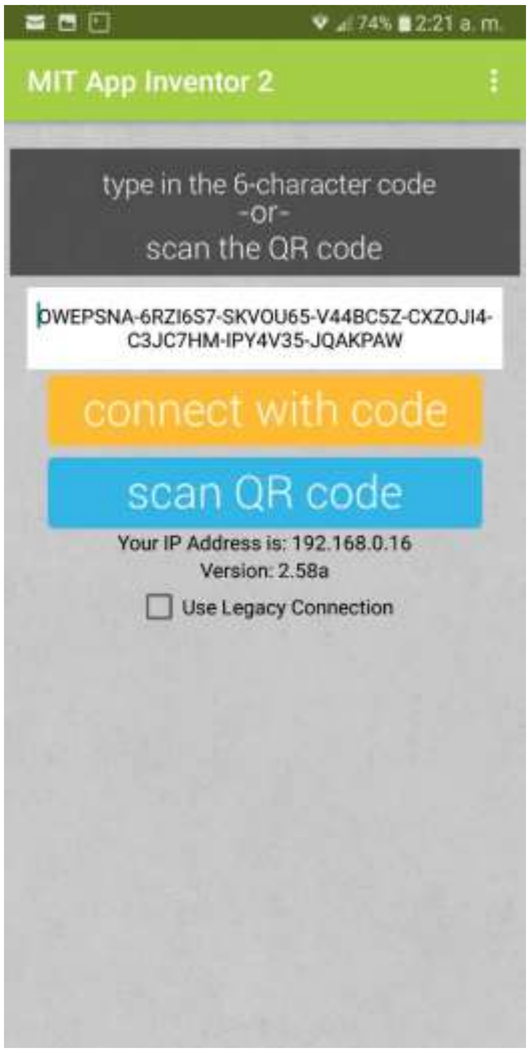
When you click on "Show ID" the following screen will appear which is a QR code of the Sansumg phone in our case the ID that identifies the phone is

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

Then in the LG Q6 phone, the program that will help us to capture this Sansumg's QR code is the one we suggest from the App Inventor application (optional), although in case we don't have it we can also simply introduce it manually when we start the registration in the LG Q6, however, to avoid any error we suggest to use it.

Using App program inventor installed in the LG Q6 mobile to capture QR code from the remote Sansumg phone we want to synchronize.


80% 11:03 p. m.
127.0.0.1:8384
Identificación del Dispositivo - localhost
OWEPSNA-6RZI6S7-SKVOU65-V44BC5Z-CXZOJI4-C3JC7HM-IPY4V35-JQAKPAW
✖ Cerrar


74% 2:21 a. m.
MIT App Inventor 2
type in the 6-character code -or- scan the QR code
OWEPSNA-6RZI6S7-SKVOU65-V44BC5Z-CXZOJI4-C3JC7HM-IPY4V35-JQAKPAW
connect with code
scan QR code
Your IP Address is: 192.168.0.16
Version: 2.58a
Use Legacy Connection

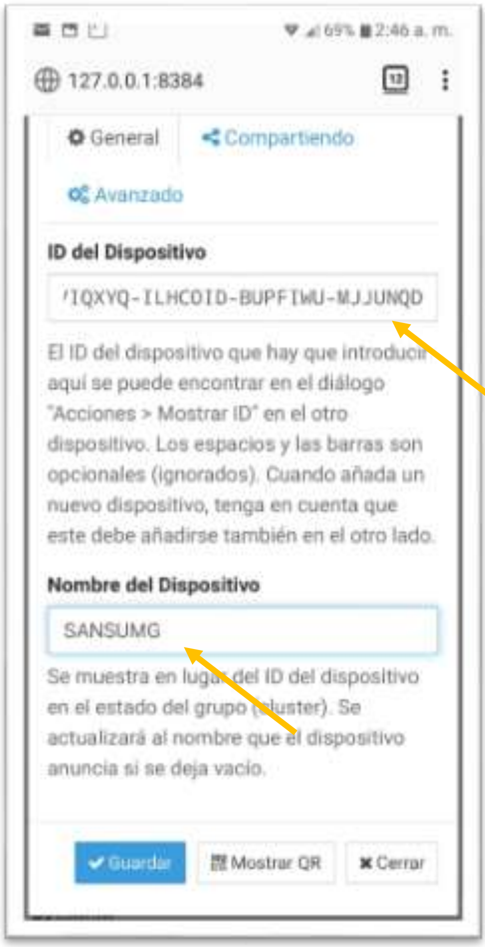
Then we copy to the clipboard the QR code in the App inventor program by clicking on the captured code, we choose "SELECT ALL" →"COPY".

With this information we proceed to register the Sansumg in the LG Q6. In the administration screen we move to the lower part where it says "Other devices" and click on the button "Add a device", we introduce the Sansumg's ID and give it a registration name.

Then in the upper tab "Share" we click and in this we mark the lower option in folder "Default" with this we will be sharing a directory that was created by Default called Sync in our device.

Then at the top we click on the "Advanced" tab and select the data compression option under "All Data".

Finally we click on the lower save button, we finish the registration in a node, this same process must be done in the remote phone or phones that we want to synchronize.



ID del Dispositivo

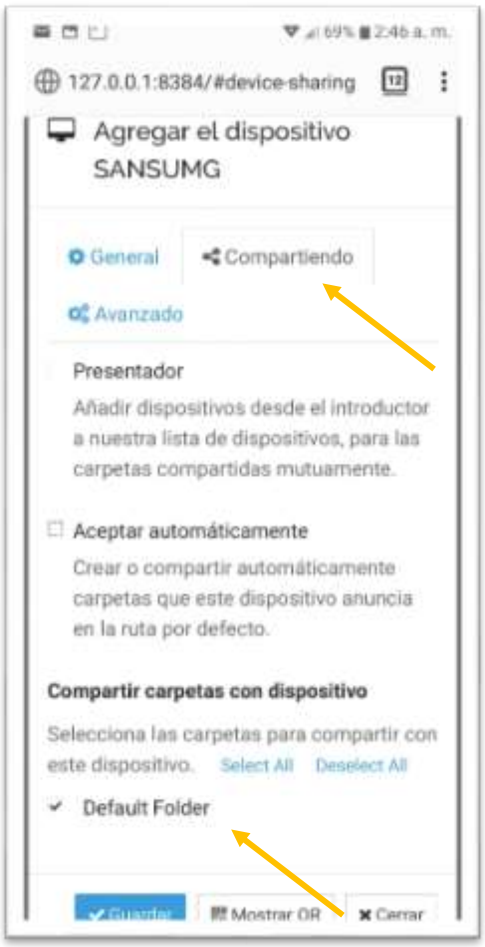
'IQXYQ-ILHCOID-BUPFIWU-MJJUNQD

El ID del dispositivo que hay que introducir aquí se puede encontrar en el diálogo "Acciones > Mostrar ID" en el otro dispositivo. Los espacios y las barras son opcionales (ignorados). Cuando añada un nuevo dispositivo, tenga en cuenta que este debe añadirse también en el otro lado.

Nombre del Dispositivo

SANSUMG

Se muestra en lugar del ID del dispositivo en el estado del grupo (cluster). Se actualizará al nombre que el dispositivo anuncia si se deja vacío.



Agregar el dispositivo SANSUMG

Presentador

Añadir dispositivos desde el introductor a nuestra lista de dispositivos, para las carpetas compartidas mutuamente.

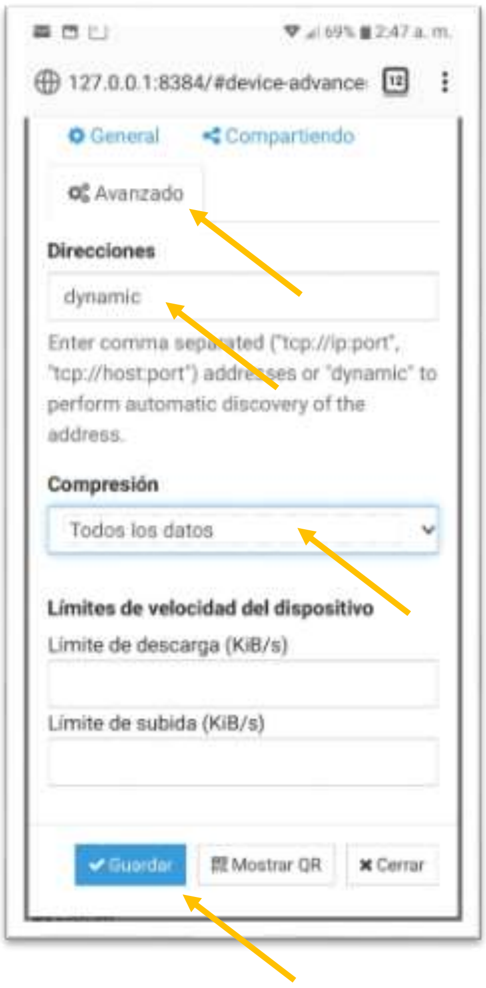
☐ **Aceptar automáticamente**

Crear o compartir automáticamente carpetas que este dispositivo anuncia en la ruta por defecto.

Compartir carpetas con dispositivo

Selecciona las carpetas para compartir con este dispositivo. Select All Deselect All

✓ Default Folder



Direcciones

dynamic

Enter comma separated ("tcp://ip:port", "tcp://host:port") addresses or "dynamic" to perform automatic discovery of the address.

Compresión

Todos los datos

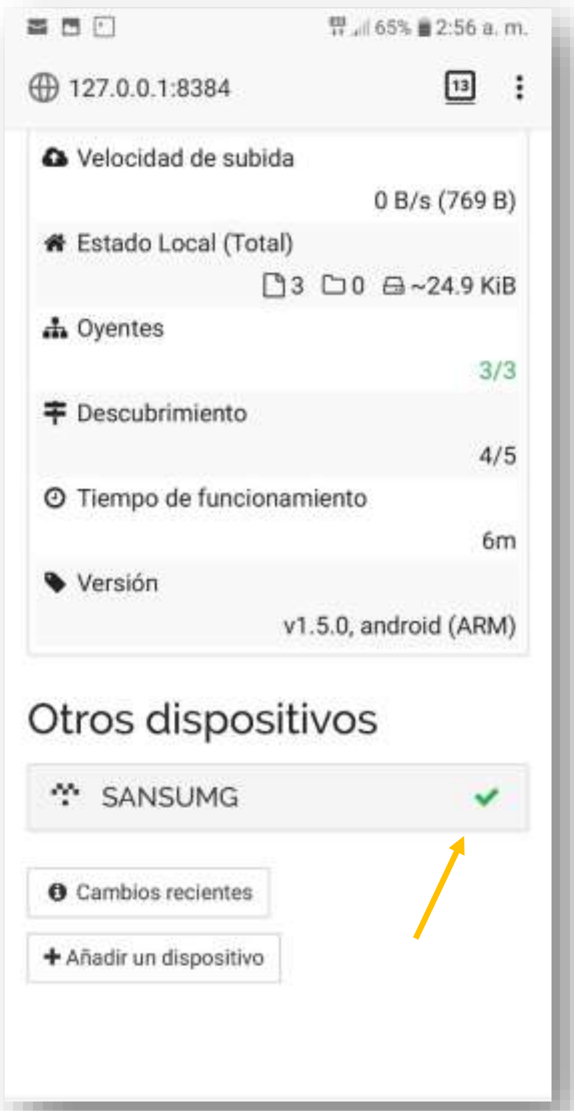
Límites de velocidad del dispositivo

Límite de descarga (KiB/s)

Límite de subida (KiB/s)

When saving and registering in both phones we will wait a maximum period of 30 seconds in which the devices are located and the connection between devices will appear as good giving a confirmation in green.

Device #1 LG Q6



65% 2:56 a. m.

127.0.0.1:8384

Velocidad de subida
0 B/s (769 B)

Estado Local (Total)
3 0 ~24.9 KiB

Oyentes
3/3

Descubrimiento
4/5

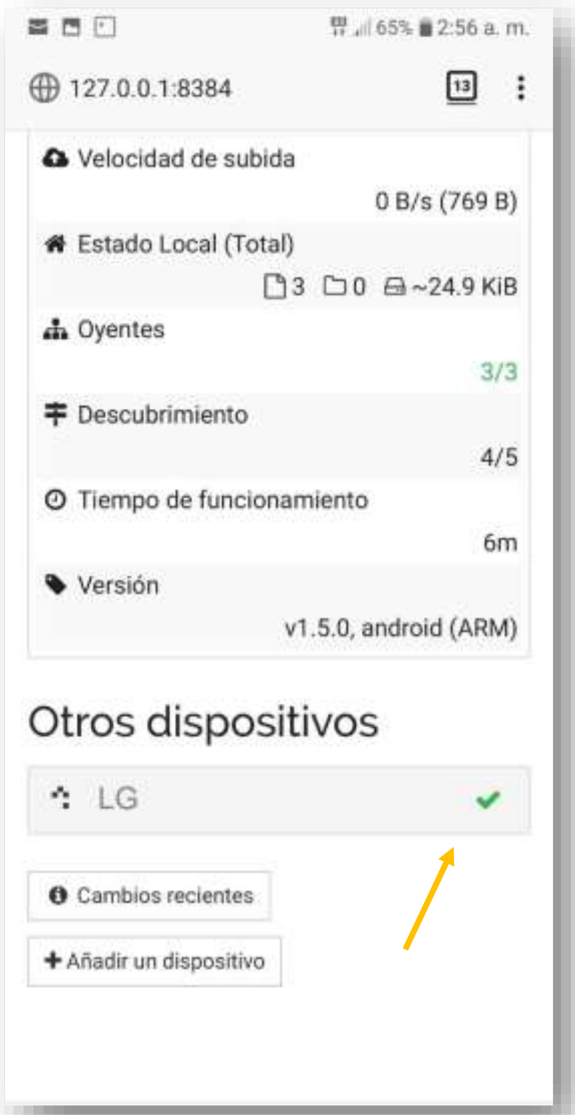
Tiempo de funcionamiento
6m

Versión
v1.5.0, android (ARM)

Otros dispositivos

SANSUMG ✓

Device #2 Sansumg



65% 2:56 a. m.

127.0.0.1:8384

Velocidad de subida
0 B/s (769 B)

Estado Local (Total)
3 0 ~24.9 KiB

Oyentes
3/3

Descubrimiento
4/5

Tiempo de funcionamiento
6m

Versión
v1.5.0, android (ARM)

Otros dispositivos

LG ✓

All information in the Sync directory located in the path /data/data/com.termux/file/home/Sync will be synchronized, any changes will be copied and synchronized.

Installation of syncthing management tool for nodes. We proceed to the installation of **SyncthingManager.**

First we install what you will need for SyncthingManager to work properly.

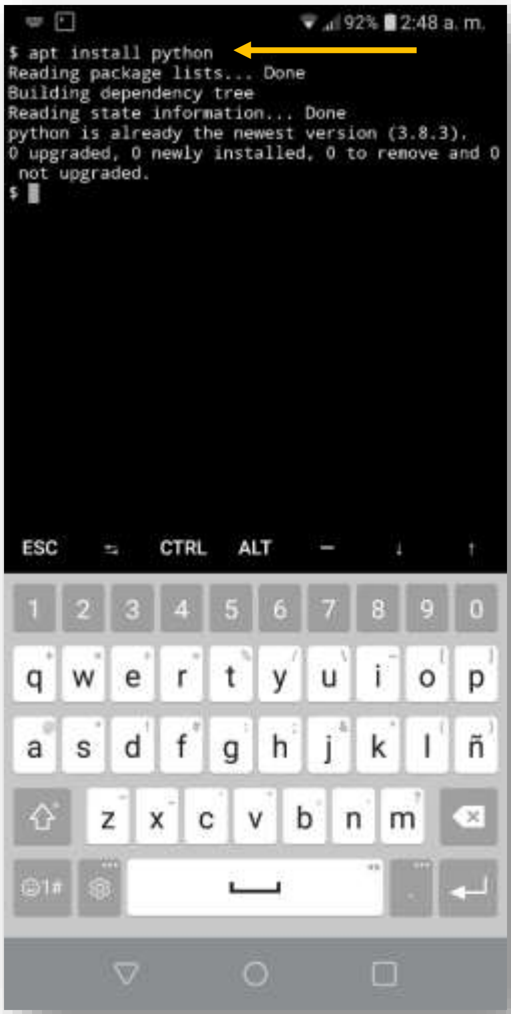
$ apt install Python

$pip3 install -upgrade pip

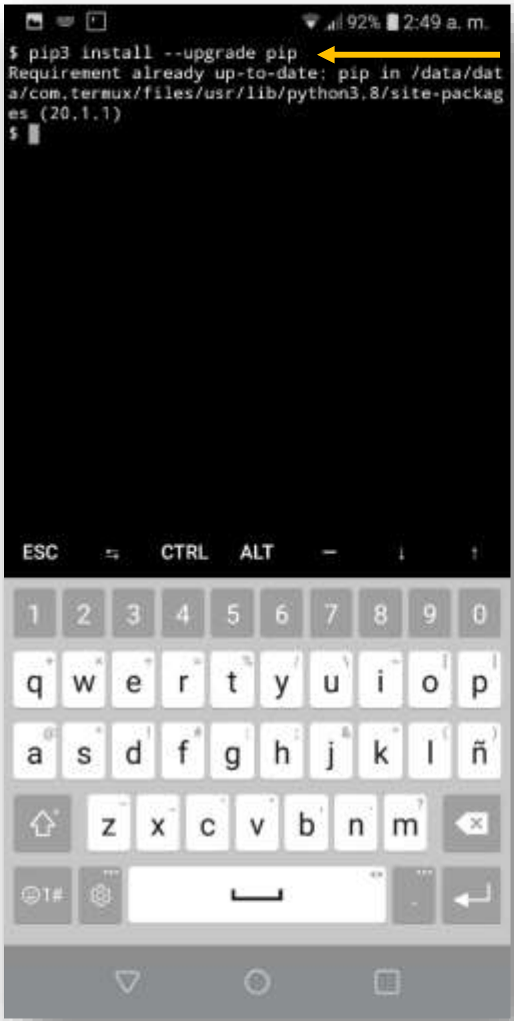
$pip install syncthing

pip3 install syncthingmanager

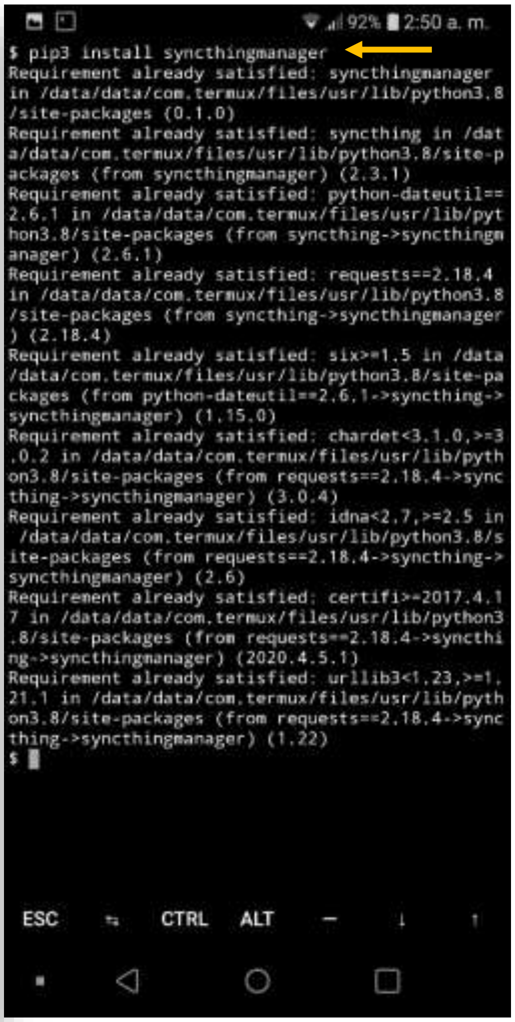
The SyncthingManager tool will help us to synchronize "peer to peer" in an automatic way and not manually for new and existing nodes.



```
$ 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.
$
```

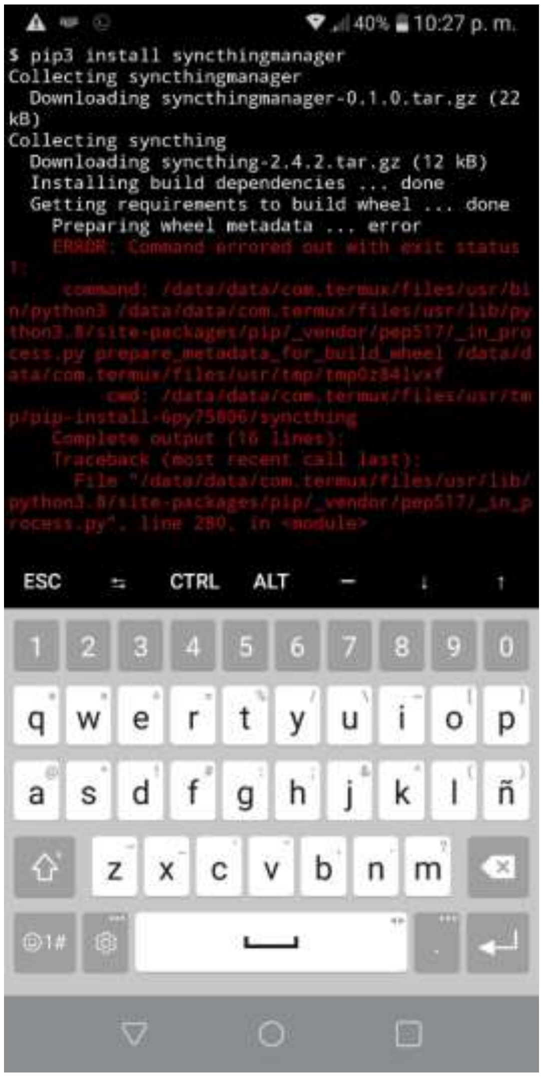


```
$ pip3 install --upgrade pip
Requirement already up-to-date: pip in /data/dat
a/com.termux/files/usr/lib/python3.8/site-packag
es (20.1.1)
$
```

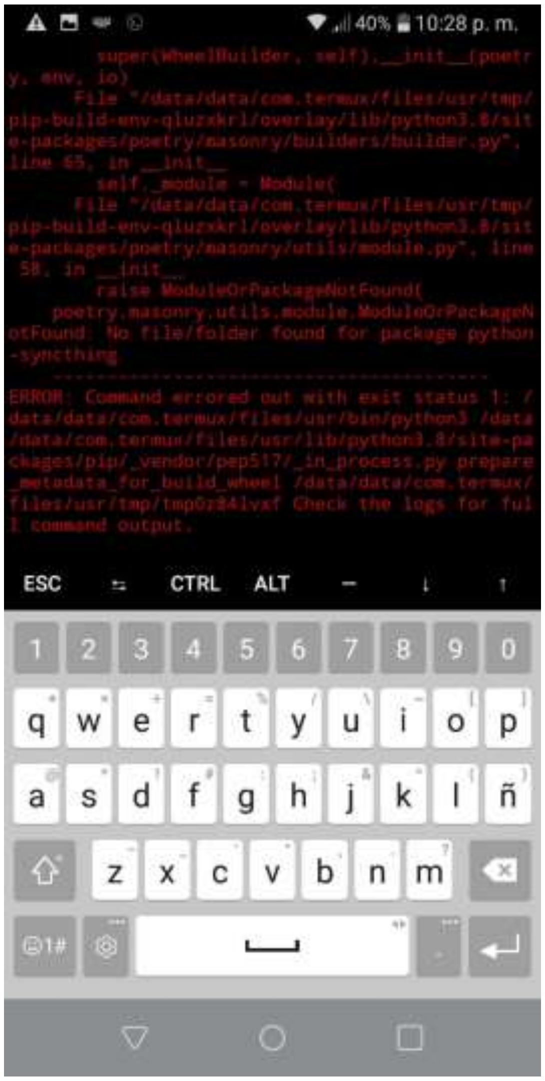


```
$ pip3 install syncthingmanager
Requirement already satisfied: syncthingmanager
in /data/data/com.termux/files/usr/lib/python3.8
/site-packages (0.1.0)
Requirement already satisfied: syncthing in /dat
a/data/com.termux/files/usr/lib/python3.8/site-p
ackages (from syncthingmanager) (2.3.1)
Requirement already satisfied: python-dateutil==
2.6.1 in /data/data/com.termux/files/usr/lib/pyt
hon3.8/site-packages (from syncthing->syncthingm
anager) (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-pa
ckages (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/pyth
on3.8/site-packages (from requests==2.18.4->sync
thing->syncthingmanager) (3.0.4)
Requirement already satisfied: idna<2.7,>=2.5 in
 /data/data/com.termux/files/usr/lib/python3.8/s
ite-packages (from requests==2.18.4->syncthing->
syncthingmanager) (2.6)
Requirement already satisfied: certifi>=2017.4.1
7 in /data/data/com.termux/files/usr/lib/python3
.8/site-packages (from requests==2.18.4->syncthi
ng->syncthingmanager) (2020.4.5.1)
Requirement already satisfied: urllib3<1.23,>=1.
21.1 in /data/data/com.termux/files/usr/lib/pyth
on3.8/site-packages (from requests==2.18.4->sync
thing->syncthingmanager) (1.22)
$
```

IMPORTANT NOTE: In the installation process of syncthingmanager version 0.1.0, in the version with Python **3.8.X** when installed in the Termux terminal it can give installation errors this is because the version of Python 3.8.X contains some new processes to install syncthingmanager and has errors with the version of Python syncthing 2.4.2, if the following installation problem occurs:



```
$ 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/tmp0z841vxf
         cwd: /data/data/com.termux/files/usr/tm
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>
```



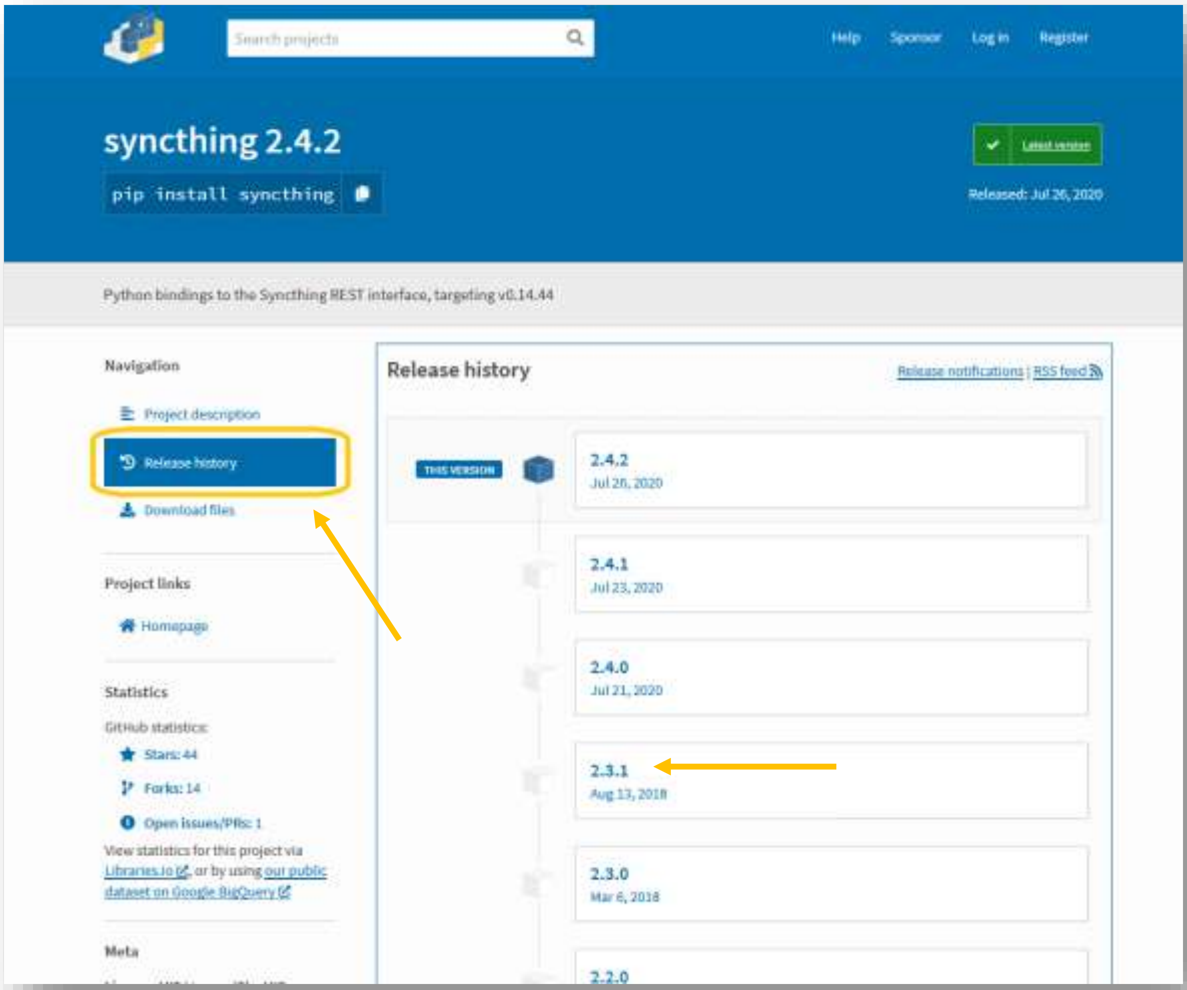
```
        super(WheelBuilder, self).__init__(poetr
y, env, io)
      File "/data/data/com.termux/files/usr/tmp/
pip-build-env-qluzxkrl/overlay/lib/python3.8/sit
e-packages/poetry/masonry/builders/builder.py",
line 65, in __init__
        self._module = Module(
      File "/data/data/com.termux/files/usr/tmp/
pip-build-env-qluzxkrl/overlay/lib/python3.8/sit
e-packages/poetry/masonry/utils/module.py", line
 58, in __init__
        raise ModuleOrPackageNotFound(
    poetry.masonry.utils.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/tmp0z841vxf Check the logs for ful
l command output.
```

To solve this problem we have the following option.

1.- Do an installation downgrade or take a previous version to syncthing for Python and then do the installation of syncmanager this would be to check the versions available in Python for syncthing, go to the site:

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

There we can check all the previous versions in the "Release history" in this case we tried syncthing version 2.3.1 and it can be installed without complications.



Search projects

Help Sponsor Log in Register

syncthing 2.4.2

pip install syncthing

Latest version

Released: Jul 26, 2020

Python bindings to the Syncthing REST interface, targeting v0.14.44

Navigation

Project description

Release history

Download files

Project links

Homepage

Statistics

GitHub statistics:

Stars: 44

Forks: 14

Open issues/PRs: 1

View statistics for this project via Libraries.io, or by using our public dataset on Google BigQuery

Meta

Release history

Release notifications | RSS feed

THIS VERSION

2.4.2
Jul 26, 2020

2.4.1
Jul 23, 2020

2.4.0
Jul 21, 2020

2.3.1
Aug 13, 2018

2.3.0
Mar 6, 2018

2.2.0

To install an older version in Python just run the following command using the following syntax $ pip install <package == version>:

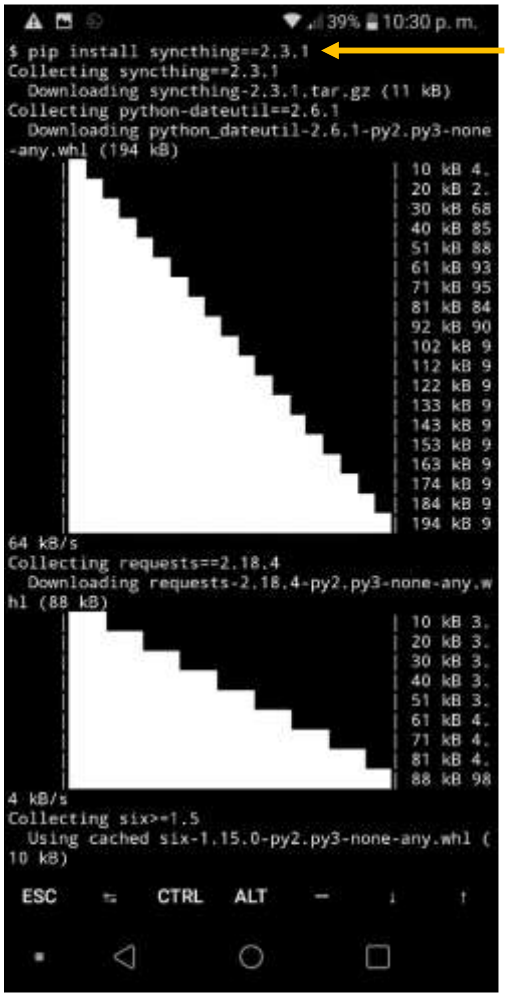
pip install syncthing==2.3.1

Then perform the installation of syncthingmanager as normal using pip version 3 with:

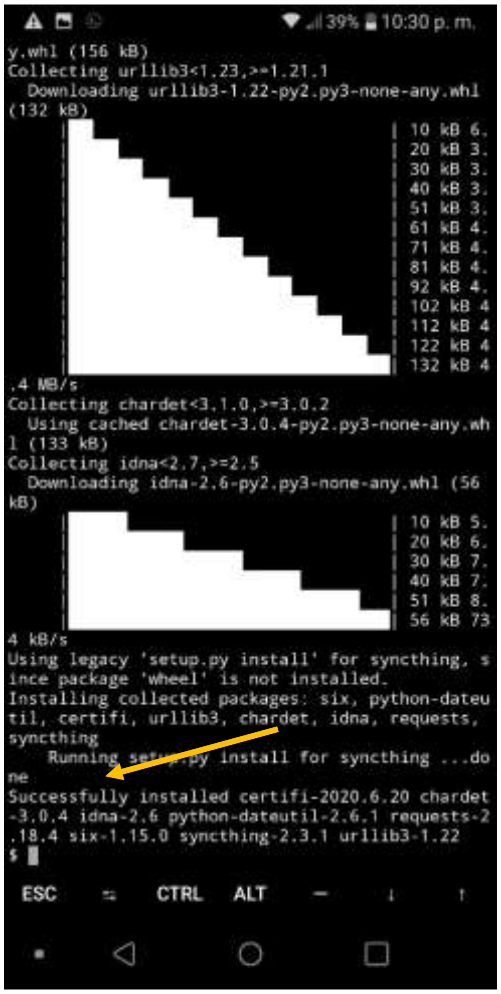
pip3 install syncthingmanager

After the installation we must execute a previous configuration for the syncthingmanager environment variabes by executing the following command in the Termux terminal:

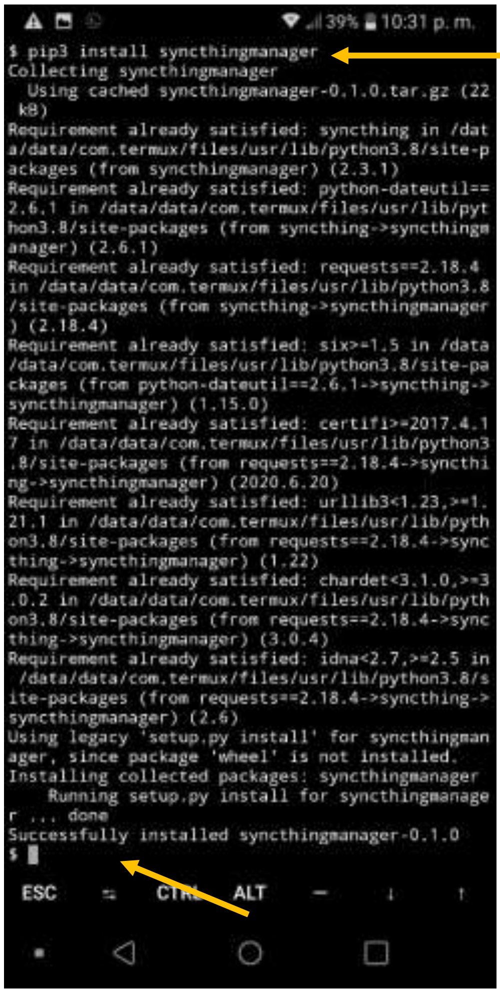
$ stman configure



```
$ 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.w
hl (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)
```



```
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.wh
l (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, s
ince package 'wheel' is not installed.
Installing collected packages: six, python-dateu
til, certifi, urllib3, chardet, idna, requests,
syncthing
    Running setup.py install for syncthing ...do
ne
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
$
```

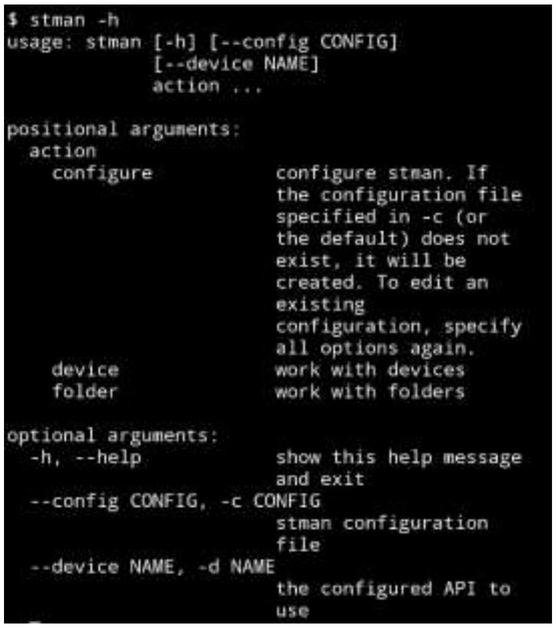


```
$ pip3 install syncthingmanager
Collecting syncthingmanager
  Using cached syncthingmanager-0.1.0.tar.gz (22
 kB)
Requirement already satisfied: syncthing in /dat
a/data/com.termux/files/usr/lib/python3.8/site-p
ackages (from syncthingmanager) (2.3.1)
Requirement already satisfied: python-dateutil==
2.6.1 in /data/data/com.termux/files/usr/lib/pyt
hon3.8/site-packages (from syncthing->syncthingm
anager) (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-pa
ckages (from python-dateutil==2.6.1->syncthing->
syncthingmanager) (1.15.0)
Requirement already satisfied: certifi>=2017.4.1
7 in /data/data/com.termux/files/usr/lib/python3
.8/site-packages (from requests==2.18.4->syncthi
ng->syncthingmanager) (2020.6.20)
Requirement already satisfied: urllib3<1.23,>=1.
21.1 in /data/data/com.termux/files/usr/lib/pyth
on3.8/site-packages (from requests==2.18.4->sync
thing->syncthingmanager) (1.22)
Requirement already satisfied: chardet<3.1.0,>=3
.0.2 in /data/data/com.termux/files/usr/lib/pyth
on3.8/site-packages (from requests==2.18.4->sync
thing->syncthingmanager) (3.0.4)
Requirement already satisfied: idna<2.7,>=2.5 in
 /data/data/com.termux/files/usr/lib/python3.8/s
ite-packages (from requests==2.18.4->syncthing->
syncthingmanager) (2.6)
Using legacy 'setup.py install' for syncthingman
ager, since package 'wheel' is not installed.
Installing collected packages: syncthingmanager
    Running setup.py install for syncthingmanage
r ... done
Successfully installed syncthingmanager-0.1.0
$
```

To check if the installation is correct we can use the command:

$ stman -h

This will give us the following result:



```
$ 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             work with devices
    folder             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
```

To use syncthingmanager we can support ourselves using the extension (OpenQbitP2PwithSSH) with this extension we can execute all the **"stman"** online commands of the syncthingmanager as a requirement we must have running syncthing in the Termux terminal to be able to execute the online commands examples below and/or use the extension (OpenQbitP2PwithSSH) that contains all the syncthingmanager commands:

To run syncthing on the termux terminal we only have to use the following command:

$ syncthing

Examples of online commands for managing syncthing through syncthingmanager from a Termux terminal

```
# 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-EKSIIKF-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-EKSIIKF-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
```

More information on the use of syncthingmanager can be found at:
https://github.com/classicsc/syncthingmanager

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

App Inventor is a software development environment created by Google Labs to build applications for the Android operating system. The user can, visually and from a set of basic tools, link a series of blocks to create the application. The system is free and can be easily downloaded from the web. Applications created with App Inventor are very easy to create because no knowledge of any programming language is required.

All the current environments that use Blockly's technology such as AppyBuilder and Thunkable among others have their free version, their way of use can be through the internet in their different sites or it can also be installed at home.

The blocks that make up the Peer to Peer architecture have been tested in App inventor and AppyBuilder but because of their code optimization they should work on the other platforms.

Online versions:

App Inventor.

https://appinventor.mit.edu/

AppyBuilder.

http://appybuilder.com/

Thunkable.

https://thunkable.com/

Version to be installed on your computer (PC):

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

Environment for developers of Blockly blocks.

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

10. What is an IdDevice or device identifier in Syncthing?

Understanding device identifiers

Each device is identified by a device ID. The device ID is used for address resolution, authentication, and authorization. The term "Device ID" could have been interchangeable with "Key ID" since the Device ID is a direct property of the public key in use.

Keys

To understand the device IDs, we must look at the underlying mechanisms. On the first start, Syncthing will create a public/private key pair.

It is currently a 384-bit ECDSA key (3072-bit RSA prior to v0.12.5, which is used as an example in this article). The keys are stored in the form of a private key (key.pem) and a self-signed certificate (cert.pem). The self-signing part does not add any security or functionality as far as Syncthing is concerned, but allows the use of the keys in a standard TLS exchange.

Device ID

To form a device ID, the SHA-256 hash of the certificate data is calculated in DER form. This means that the hash covers all the information in the Certificate: previous section.

The hash results in a 256-bit hash that we encode using base32. Base32 encodes five bits per character, so we need 256/5 = 51.2 characters to encode the device ID. This becomes 52 characters in practice, but 52 base32 characters would be decoded to 260 bits, which is not an integer number of bytes. The base32 encoding adds padding to 280 bits (the next multiple of 5 and 8 bits) so that the resulting ID looks more or less like this:

MFZWI3DBONSGYYLTMRWGC43ENRQXGZDMMFZWI3DBONSGYYLTMRWA ====

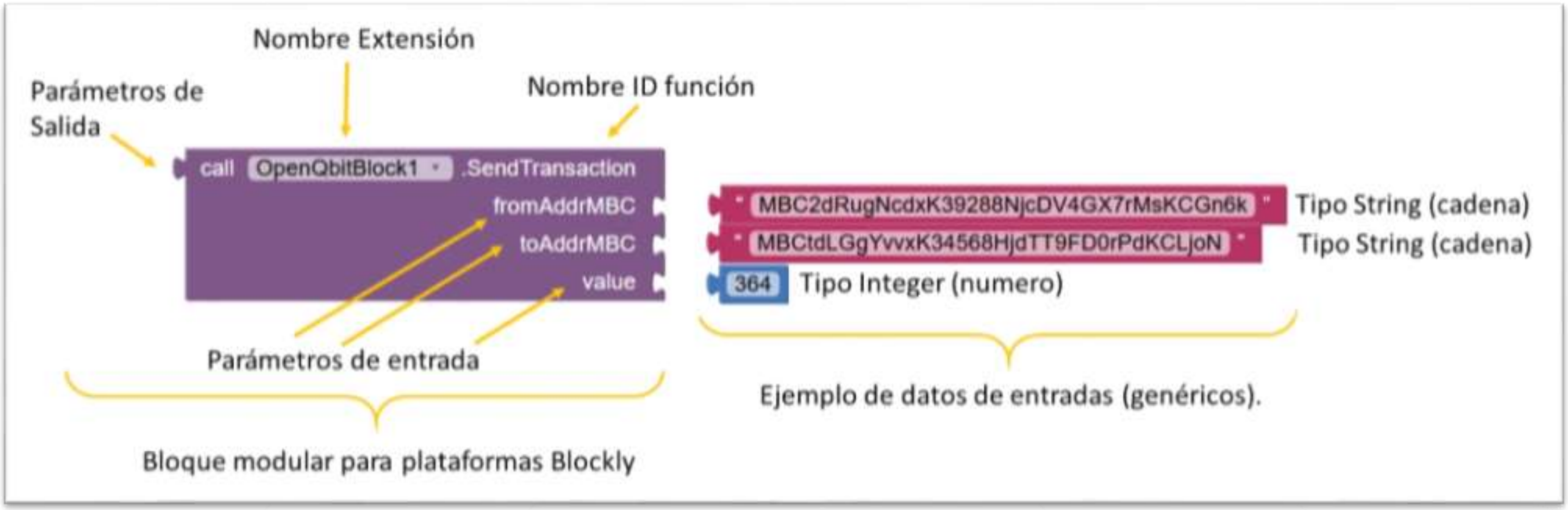
The padding (====) is removed, the device ID is divided into four groups and check digits are added for each group. For display purposes, the device ID is grouped with dashes, resulting in the final sample value:

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

11. Definition and use of blocks in OpenQbitP2PwithSSH

We will start by explaining the distribution of the data that all the blocks will have, their syntax of use and configuration.

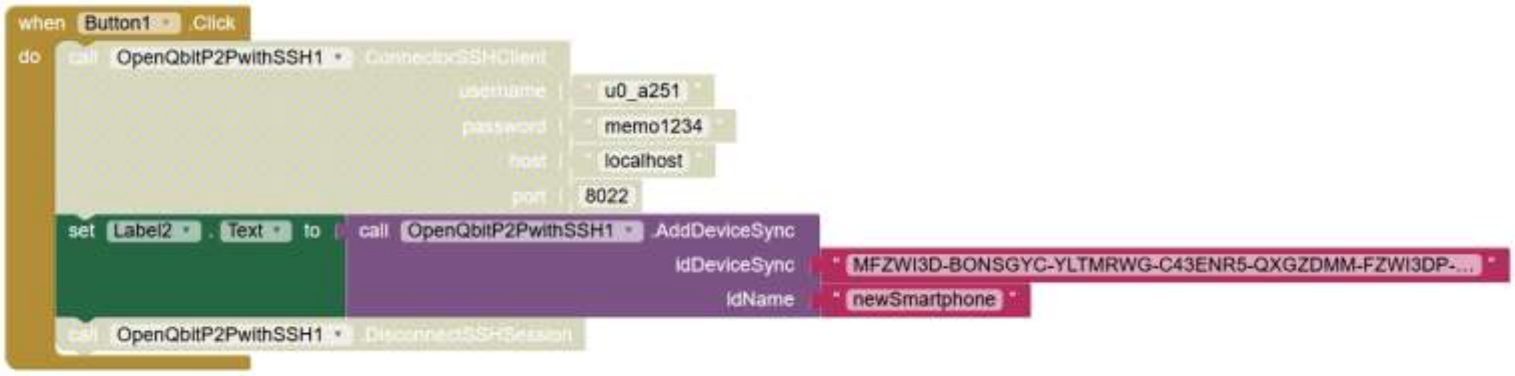
In the following generic example we can see a modular block and its input and output parameters, as well as the types of input data, these data can be of String (string) or Integer (integer or decimal) type. We show how it is used and configure it for its proper functioning.


Nombre Extensión
Parámetros de Salida
Nombre ID función
call OpenQbitBlock1 .SendTransaction
fromAddrMBC
toAddrMBC
value
MBC2dRugNcdxK39288NjcDV4GX7rMsKCGn6k
MBCtdLGgYvvxK34568HjdTT9FD0rPdKCLjoN
364
Tipo String (cadena)
Tipo String (cadena)
Tipo Integer (numero)
Parámetros de entrada
Ejemplo de datos de entradas (genéricos).
Bloque modular para plataformas Blockly

Each module block will have its description and will be named in case it has any mandatory or optional dependency(s) of other blocks used as input parameters and the integration process will be announced.

12.OpenQbitP2PwithSSH extension.

Block to add a new device (**AddDeviceSync**)


when Button1 .Click
do call OpenQbitP2PwithSSH1 .ConnectorSSHClient
username "u0_a251"
password "memo1234"
host "localhost"
port 8022
set Label2 . Text to call OpenQbitP2PwithSSH1 .AddDeviceSync
idDeviceSync "MFZWI3D-BONSGYC-YLTMRWG-C43ENR5-QXGZDMM-FZWI3DP-..."
idName "newSmartphone"
call OpenQbitP2PwithSSH1 .DisconnectSSHSession

Mandatory dependency: Block (**ConnectorSSHClient**), Block (**DisconnectSSHSession**).

Input parameters: **idDeviceSyn** <string>, idName **<string>**

idDevice. - is an arrangement of capital letters divided into 8 sets, each set consisting of 7 elements. The string must measure a total of 56 characters.

Example:

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

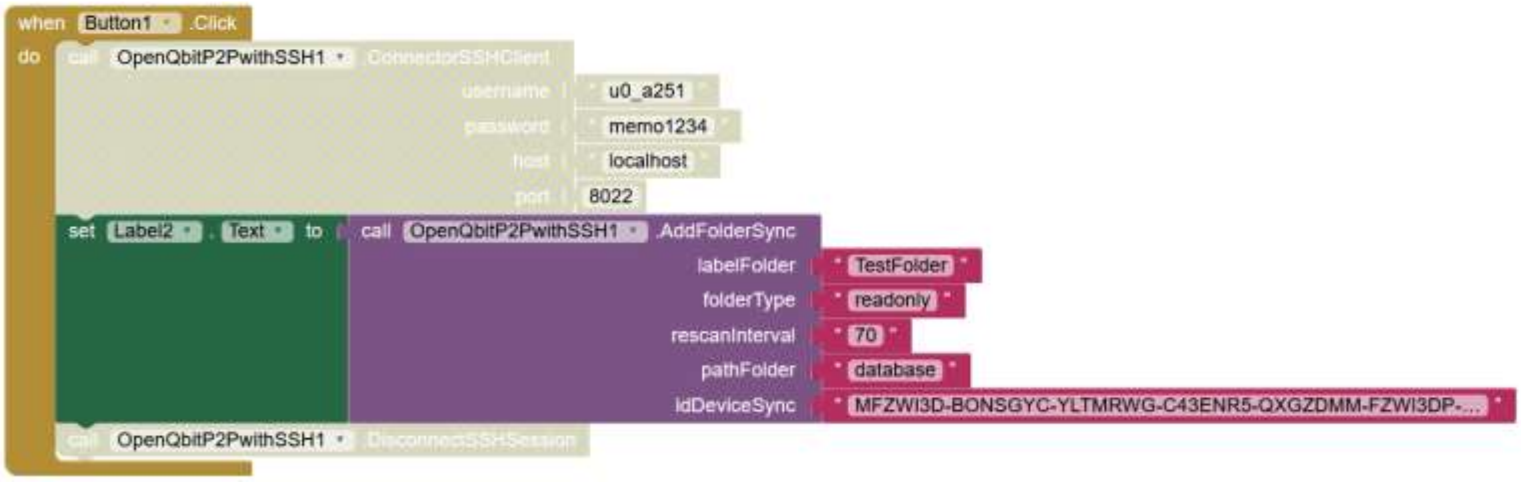
***idName*. -** is an identifier or label that the user chooses to identify the new device.

Output parameters: Delivers a message. **"Device already configured: idDevice".**

Then use the block **(GetDeviceList)** to check that the device ID has been added.

Description: Communication block to add a new device to connect to syncthing running in the Termux terminal, via SSH (Secure Shell) communication protocol.

Block to add a new folder or directory to synchronize (**AddFolderSync)**



```
when Button1 .Click
do  call OpenQbitP2PwithSSH1 .ConnectorSSHClient
                         username  "u0_a251"
                         password  "memo1234"
                             host  "localhost"
                             port  8022
    set Label2 . Text to  call OpenQbitP2PwithSSH1 .AddFolderSync
                                        labelFolder  "TestFolder"
                                         folderType  "readonly"
                                     rescanInterval  "70"
                                         pathFolder  "database"
                                       idDeviceSync  "MFZWI3D-BONSGYC-YLTMRWG-C43ENR5-QXGZDMM-FZWI3DP-..."
    call OpenQbitP2PwithSSH1 .DisconnectSSHSession
```

Mandatory dependency: Block (**ConnectorSSHClient)**, Block (**DisconnectSSHSession**).

Input parameters: **labelFolder** <string>, folderType **<string** - options: readonly or readwrite>, rescanInterval <string>, pathFolder <string>, idDevice<string>

labelFolder. - Label that identifies the directory that will be shared among the devices.

FolderType. - This will be the permissions that the directory will have if it only reads "readonly" or reads and writes "readwrite".

rescanInterval. - This is the time interval in seconds for you to review the source change deltas and perform the synchronization to the local directory.

pathFolder. - This is the path of the directory or folder that will be shared.

idDevice. - It is an arrangement of characters in capital letters divided into 8 sets, each set consisting of 5 elements. The string must measure a total of 40 characters.

Example:

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

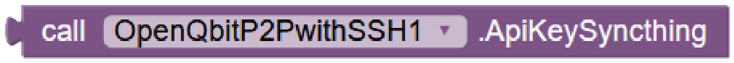
***idName*. -** It is an identifier or label that the user chooses to identify the new device.

Output parameters: Delivers a message. **"The folder IDDevice is already in use."** In case of success the execution of the command does not deliver a result.

Then use the block **(InfoFolderSync)** to check that the directory was configured.

Description: Communication block to add a new directory configuration to connect to syncthing running on the Termux terminal, via SSH (Secure Shell) communication protocol.

Block to get the API-KEY of the local syncthing that is running (**ApyKeySyncthing)**



Mandatory dependency: Block (**ConnectorSSHClient)**, Block (**DisconnectSSHSession**).

Input parameters: **None - N/A**

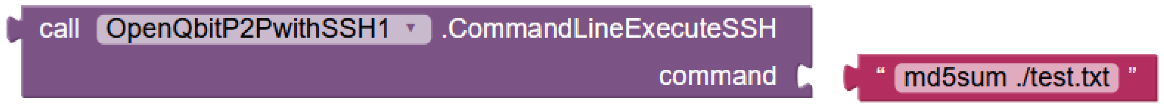
Output parameters: It does not deliver the content of the configuration file where the API-KEY is located to be used with the Rest API of Syncthing.

Example:

```
[DEFAULT]
name = localhost

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

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

Description: Communication block obtains the content of the syncthing configuration file containing the API-KEY.

Block for Executing Commands in Termux Linux Terminal (**CommandLineExecuteSSH**).



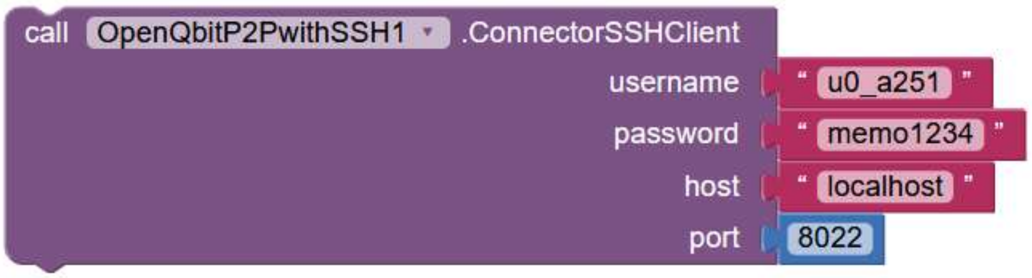
Mandatory dependency: Block (**ConnectorSSHClient)**, Block (**DisconnectSSHSession**).

Input parameters: **command** <string>

Output parameters: Variable data, depending on the executed command or program.

Description: Command execution block in Termux terminal pre-requisite make a connection with the block (**ConnectorSSHClient)** can execute all kinds of commands online and/or get specific execution data from scripts or programs that have a CLI (Command-Line Interface) online.

Connector Block SSH Client (**ConnectorSSHClient**)



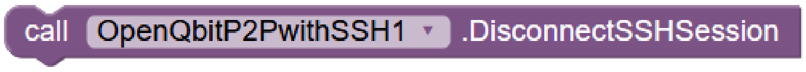
```
call OpenQbitP2PwithSSH1 .ConnectorSSHClient
                    username  " u0_a251 "
                    password  " memo1234 "
                    host      " localhost "
                    port      8022
```

Input parameters: **username** <string>, **password <string>,** host <string>, port<integer>

Output parameters: Not applicable - N/A

Description: Communication block to connect to the SSH server of the Termux terminal, via SSH (Secure Shell) communication protocol.

Block to close the SSH session (**DisconnectSSHSession**)

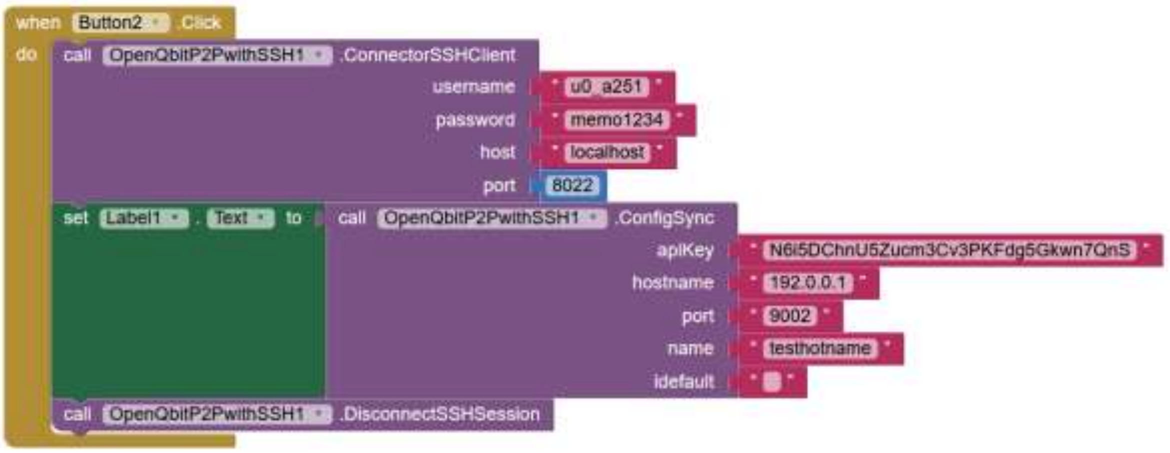


```
call OpenQbitP2PwithSSH1 .DisconnectSSHSession
```

Input and output parameters: Not applicable (none)

Description: Block to close SSH session.

Block to modify the configuration file (**ConfigSync**)



```
when Button2 .Click
do  call OpenQbitP2PwithSSH1 .ConnectorSSHClient
                        username  " u0_a251 "
                        password  " memo1234 "
                        host      " localhost "
                        port      8022
    set Label1 . Text to  call OpenQbitP2PwithSSH1 .ConfigSync
                                          apiKey    " N6i5DChnU5Zucm3Cv3PKFdg5Gkwn7QnS "
                                          hostname  " 192.0.0.1 "
                                          port      " 9002 "
                                          name      " testhotname "
                                          idefault  " "
    call OpenQbitP2PwithSSH1 .DisconnectSSHSession
```

Mandatory dependency: Use before with Block (**ConnectorSSHClient)**, Block (**DisconnectSSHSession**).

Input parameters: **to piKey** <string> , hostname<string>, port<string>, name<string>, pdefault< string>.

apiKey. - It is the API-KEY of the local or remote syncthing that you want to change parameters of the syncthingmanager configuration file.

hostname. - This is the name of the device to which the changes will be referenced. To change the local file of the device the parameter to use is "localhost".

port. - This is the port on which the requests will be connected to the syncthing.

name. - This is the identifier or label within the syncthingmanager configuration file.

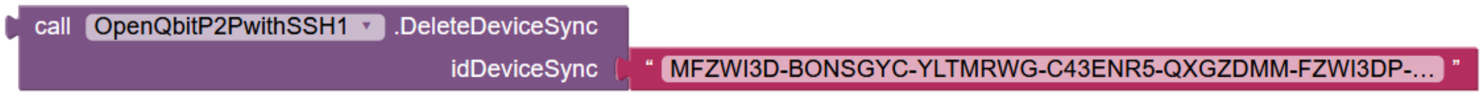
idefault.- Is the parameter to define the local device as default if the option is left empty it will not be the default, in case you want this configuration to be the default you have to use the parameter **- - default ,** the parameter starts with two minus "-" signs and then the word default everything must go together without spaces.

Output parameters: Write the parameters to the synthingmanager configuration file in the path: **~/.config/syncthingmanager/syncthingmanager.conf**

Description: Execution block parameters to modify the syncthingmanager configuration file.

Block to delete a device that is registered in the local syncthing (**DeleteDeviceSync**)

Mandatory dependency: Use before with Block (**ConnectorSSHClient)**, Block



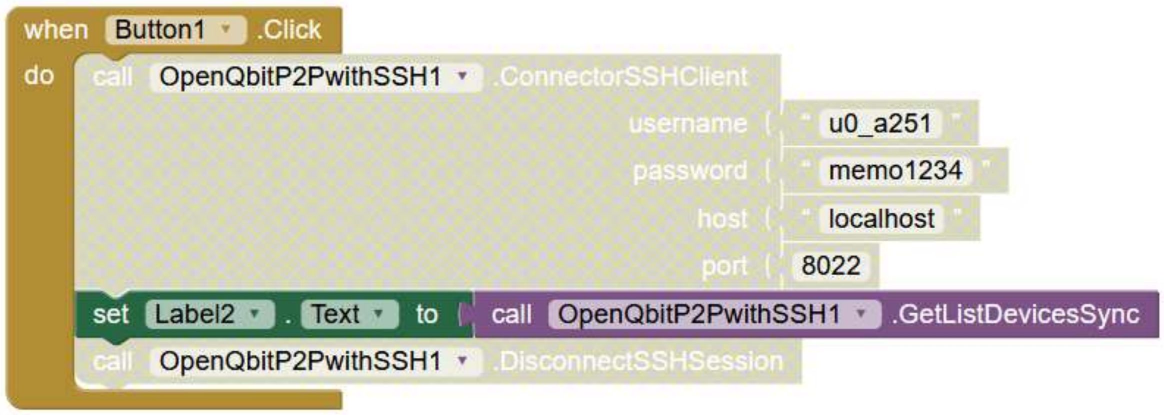
(**DisconnectSSHSession**).

Input parameters: Not applicable - N/A

Output parameters: It delivers the message "Successfully delete id = **idDevideSync**". After executing this block you can run the block (GetDeviceList) to verify that the desired idDevice has been deleted.

Description: Block to delete a device from the local syncthing.

Block to list all idDevice in local syncthing (**GetListDeviceSync**)


when Button1 .Click
do call OpenQbitP2PwithSSH1 .ConnectorSSHClient
username " u0_a251 "
password " memo1234 "
host " localhost "
port 8022
set Label2 . Text to call OpenQbitP2PwithSSH1 .GetListDevicesSync
call OpenQbitP2PwithSSH1 .DisconnectSSHSession

Mandatory dependency: Use before with Block (**ConnectorSSHClient)**, Block (**DisconnectSSHSession**).

Input parameters: Not applicable (none)

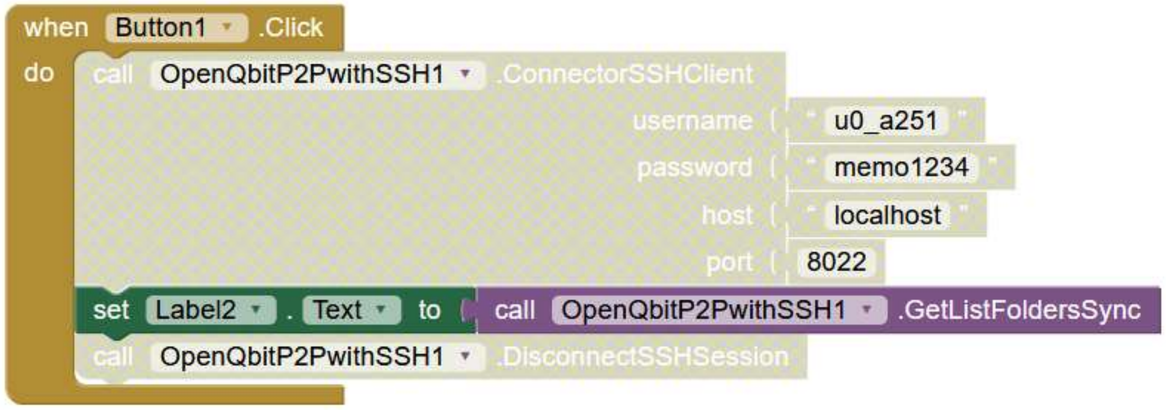
Outbound parameters: Delivers the list of IdDevice or identifiers of the devices registered in the local syncthing.

Example:

```
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
```

Description: Block for generating the list of IDs.

Block to generate list of shared directories (**GetListFoldersSync**)


when Button1 .Click
do call OpenQbitP2PwithSSH1 .ConnectorSSHClient
username " u0_a251 "
password " memo1234 "
host " localhost "
port 8022
set Label2 . Text to call OpenQbitP2PwithSSH1 .GetListFoldersSync
call OpenQbitP2PwithSSH1 .DisconnectSSHSession

Mandatory dependency: Use before with Block (**ConnectorSSHClient)**, Block (**DisconnectSSHSession**).

Input parameters: Not applicable (none)

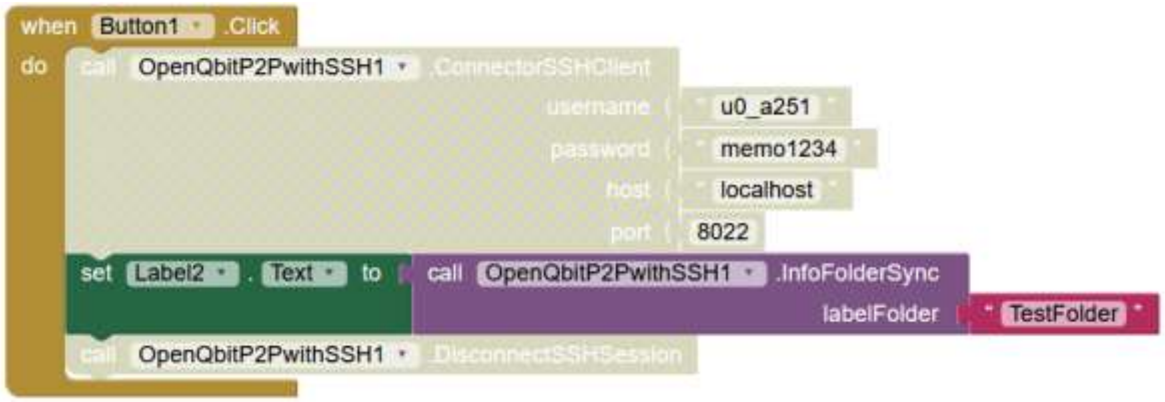
Outbound parameters: Delivers the list of forlders or shared directories.

Example:

```
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/
```

Description: Block to generate the list of local and remote shared folders or directories.

Block to obtain the configuration of a folder or directory (InfoFolderSync)


when Button1 .Click
do call OpenQbitP2PwithSSH1 .ConnectorSSHClient
username " u0_a251 "
password " memo1234 "
host " localhost "
port 8022
set Label2 . Text to call OpenQbitP2PwithSSH1 .InfoFolderSync
labelFolder " TestFolder "
call OpenQbitP2PwithSSH1 .DisconnectSSHSession

Mandatory dependency: Use before with Block (ConnectorSSHClient), Block (DisconnectSSHSession).

Input parameters: Not applicable (none)

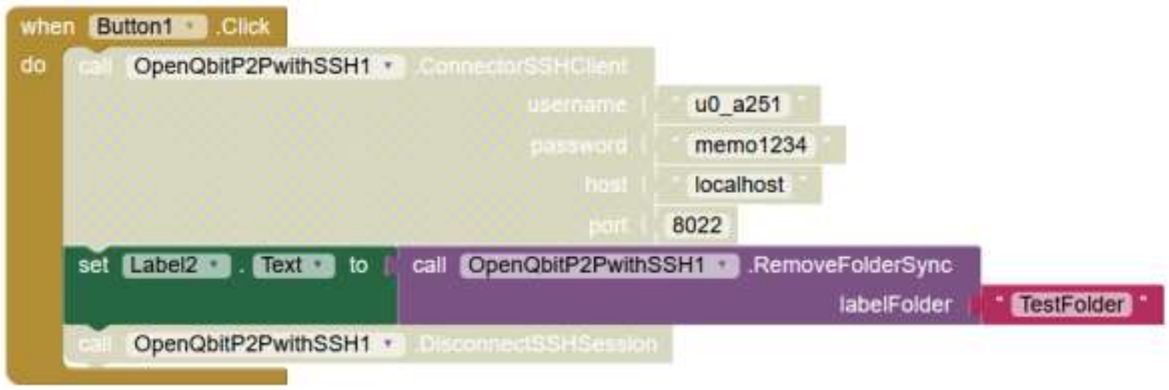
Output Parameters: Delivers the configuration of the shared folder or directory.

Example:

```
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
```

Description: Block to see the parameter configuration of a selected folder or directory, both local and remote.

Block to delete a shared folder or directory (RemoveFoldersSync)


when Button1 .Click
do call OpenQbitP2PwithSSH1 .ConnectorSSHClient
username " u0_a251 "
password " memo1234 "
host " localhost "
port 8022
set Label2 . Text to call OpenQbitP2PwithSSH1 .RemoveFolderSync
labelFolder " TestFolder "
call OpenQbitP2PwithSSH1 .DisconnectSSHSession

Mandatory dependency: Use before with Block (**ConnectorSSHClient)**, Block (**DisconnectSSHSession**).

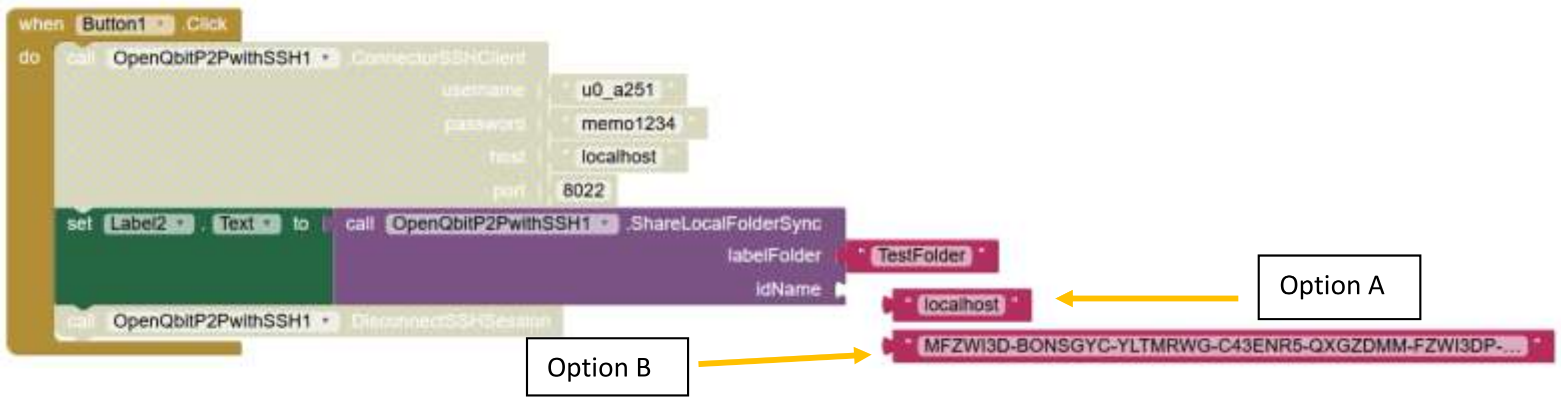
Input parameters: Not applicable (none)

Output parameters: Not applicable - N/A

To check if the deletion in the list of shared directories has been done correctly use the block (**GetListFoldersSync**).

Description: Remove or delete a folder or directory that is in the syncthing sharing list.

Block to share folder or directory (**ShareLocalFolderSync**)


when Button1 .Click
do call OpenQbitP2PwithSSH1 .ConnectorSSHClient
username u0_a251
password memo1234
host localhost
port 8022
set Label2 . Text to call OpenQbitP2PwithSSH1 .ShareLocalFolderSync
labelFolder TestFolder
idName
localhost
MFZWI3D-BONSGYC-YLTMRWG-C43ENR5-QXGZDMM-FZWI3DP-...
call OpenQbitP2PwithSSH1 .DisconnectSSHSession
Option A
Option B

Required dependency: Use before with Block (**ConnectorSSHClient)**, Block (**DisconnectSSHSession), Block** (**AddFoldersSync**)

Input parameters: **labelfolder** <string>, idName<string>.

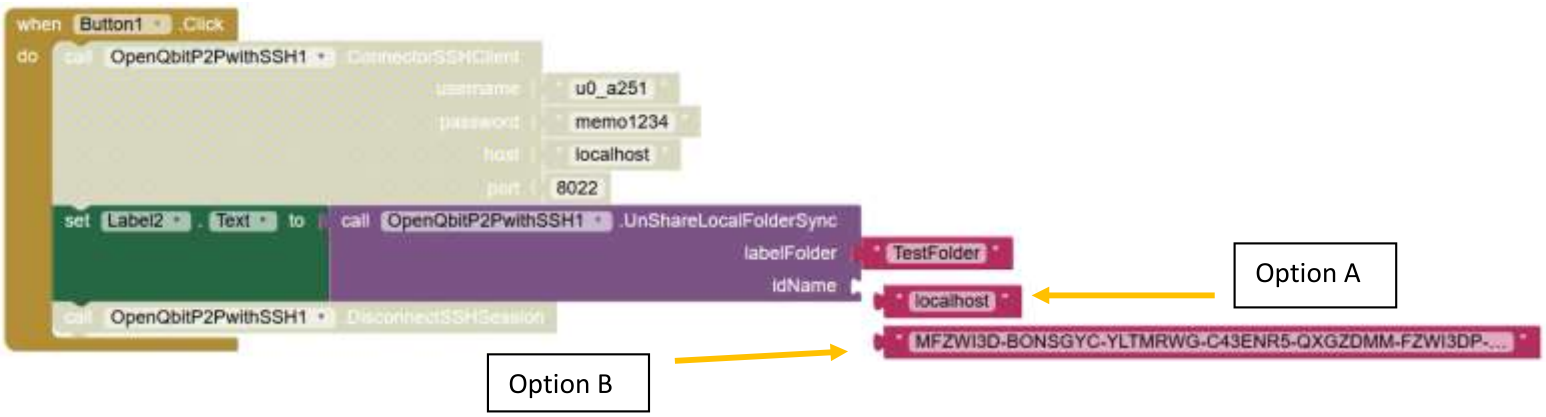
labelfolder. - Label that identifies the directory you gave when you registered the folder or directory with the block (**AddFoldersSync)** this label can be seen by running the block (**GetListFoldersSync**).

Example of the block information (**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. - There are two options for using option A is to use for the local device "localhost" or as option B you can also enter the **idDevice** Identifier for local or remote device.

Block to remove the folder or directory sharing service (**UnShareLocalFoldersSync**)


when Button1 .Click
do call OpenQbitP2PwithSSH1 .ConnectorSSHClient
username u0_a251
password memo1234
host localhost
port 8022
set Label2 . Text to call OpenQbitP2PwithSSH1 .UnShareLocalFolderSync
labelFolder TestFolder
idName localhost
MFZWI3D-BONSGYC-YLTMRWG-C43ENR5-QXGZDMM-FZWI3DP-...
call OpenQbitP2PwithSSH1 .DisconnectSSHSession
Option A
Option B

Required dependency: Use before with Block (**ConnectorSSHClient), Block** (**DisconnectSSHSession), Block** (**ShareLocalFoldersSync**)

Input parameters: **labelfolder** <string>, idName<string>.

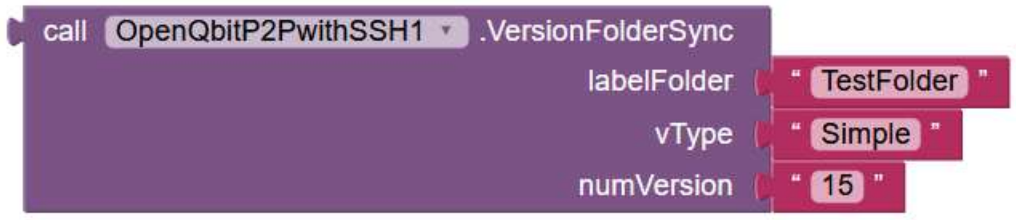
labelfolder. - Label that identifies the directory you gave when you registered the folder or directory with the block (**AddFoldersSync)** this label can be seen by running the block (**GetListFoldersSync**).

Example of the block information (**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. - There are two options for using option A is to use for the local device "localhost" or as option B you can also enter the **idDevice** Identifier for local or remote device.

Block to configure the folder or directory version property (**VersioningFoldersSyn**)



Mandatory dependency: Use before with Block (**ConnectorSSHClient)**, Block (**DisconnectSSHSession**).

Input parameters: labelfolder<string>, vType<string>, numVersion<string>

labelfolder. - Label that identifies the directory you gave when you registered the folder or directory with the block (**AddFoldersSync).** This label can be seen by running the block (**GetListFoldersSync**)**.**

Three types are handled (Simple, Staggered and External) to see details refer to the site: https://docs.syncthing.net/users/versioning.html?highlight=version

Version number: This is the version number specified in the folder or directory.

Output parameters: Not applicable - N/A

To check if the version parameters have been done correctly use the block (**InfoFoldersSync**).

Example:

```
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
```

Description: Places the parameters for version control in directories.

How to use the blocks to use the "Tor" network?

Before we must install the command to use it online from the network "Tor" with the execution of the following command in the termux terminal or in its case in the Linux of the installed PC or server.

For Termux Terminal:

$ pkg install torsocks

For Linux:

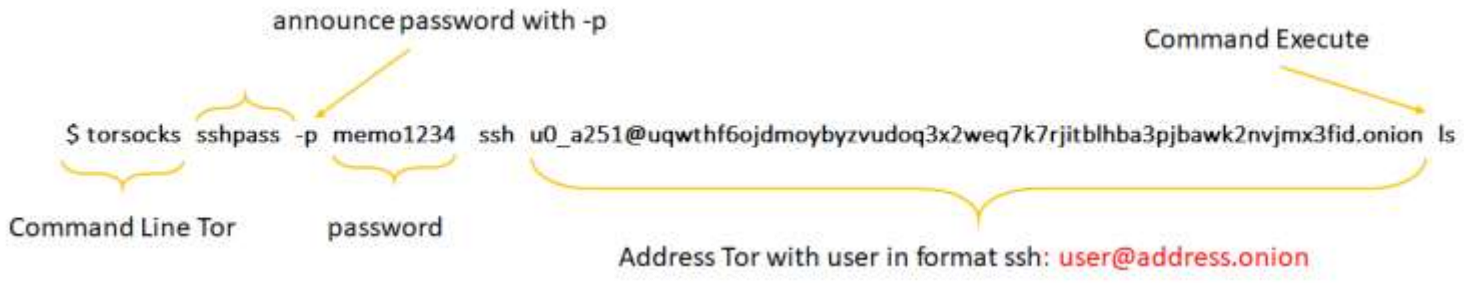
$ apt install torsocks

To use the online torsocks command we need to have the Tor address of the remote device. We have two ways of knowing this, one is on the remote device we give the following command on the termux or Linux terminal:

$ more ~/.tor/hidden_ssh/hostname (note that the tor home directory is hidden and a "." must be used before the tor directory name). The Tor service should already be running and configured as seen above.

The second option is to use the block (GetAddressTor) on the device that we want to have the Tor address on, before using this block it is considered to have already configured and started the Tor service for SSH as seen earlier in the Tor service for SSH configuration section.

We start using the "Tor" network; the first step to consider is that we will be using the "Tor" network through an SSH (Secure Shell) tunnel, so we must first run the following command online at a shell terminal on either the mobile phone (Termux), PC, tablet or server for the first and only time from the source device to the target (remote) device as follows:



```
$ torsocks  sshpass  -p  memo1234   ssh  u0_a251@uqwthf6ojdmoybyzvudoq3x2weq7k7rjitblhba3pjbawk2nvjmx3fid.onion  ls
```

announce password with -p
Command Execute
Command Line Tor
password
Address Tor with user in format ssh: user@address.onion

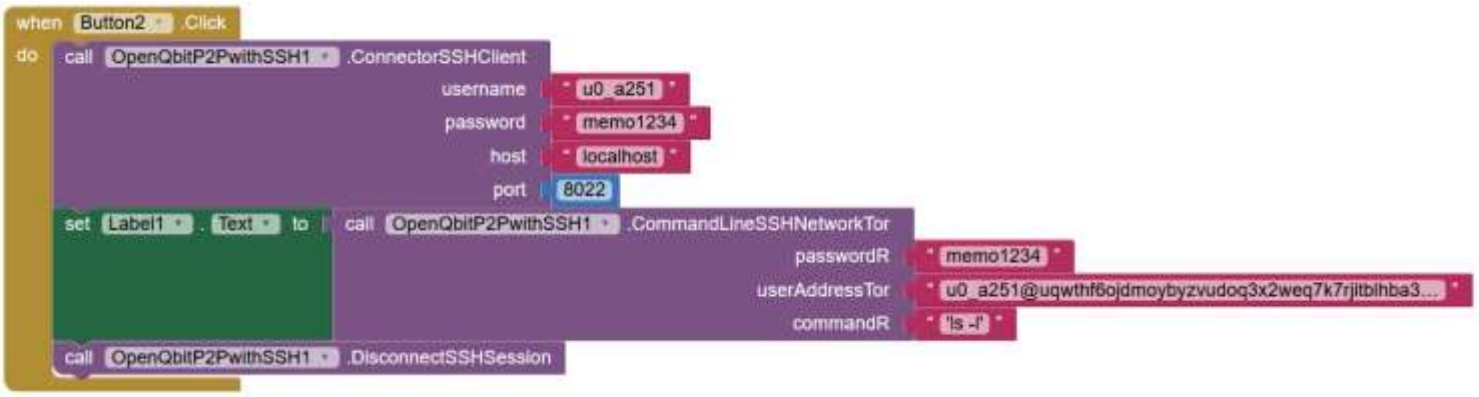
The previous command will create the certificates of authorization of the destination device after executing it will give us the following a message that we must validate and confirm the keys of the certificate.

Example:

```
The authenticity of host
'u0_a251@uqwthf6ojdmoybyzvudoq3x2weq7k7rjitblhba3pjbawk2nvjmx3fid.onion
(127.42.42.0)' can't be established. ECDSA key fingerprint is
SHA256:f22LX7WJfLGOiKxP+0+cA/l5Q1GsJLFA30ZyMyGLMl4. Are you sure you want
to continue connecting (yes/no)? yes
```

Now we can use the "Tor" network blocks.

Block to execute SSH command through network "Tor" (**CommandSSHNetworkTor**)



```
when Button2 .Click
do  call OpenQbitP2PwithSSH1 .ConnectorSSHClient
                              username  "u0_a251"
                              password  "memo1234"
                              host      "localhost"
                              port      8022
    set Label1 . Text to  call OpenQbitP2PwithSSH1 .CommandLineSSHNetworkTor
                                                    passwordR       "memo1234"
                                                    userAddressTor  "u0_a251@uqwthf6ojdmoybyzvudoq3x2weq7k7rjitblhba3..."
                                                    commandR        "ls -l"
    call OpenQbitP2PwithSSH1 .DisconnectSSHSession
```

Mandatory dependency: Use before with Block (**ConnectorSSHClient**), Block (**DisconnectSSHSession**).

Input parameters: **passwordR** <string>, userAddressTor<string>, commandR<string>

passwordR. - -This is the password of the user you will connect to at the remote device (destination).

userAddressTor. - This is the user's address in SSH format:

Example:

$$\underbrace{\text{u0\_a251@uqwthf6ojdmoybyzvudoq3x2weq7k7rjitblhba3pjbawk2nvjmx3fid.onion}}_{\text{Address Tor with user in format ssh: user@address.onion}}$$

commandR. - Command to be executed on the remote device (destination).

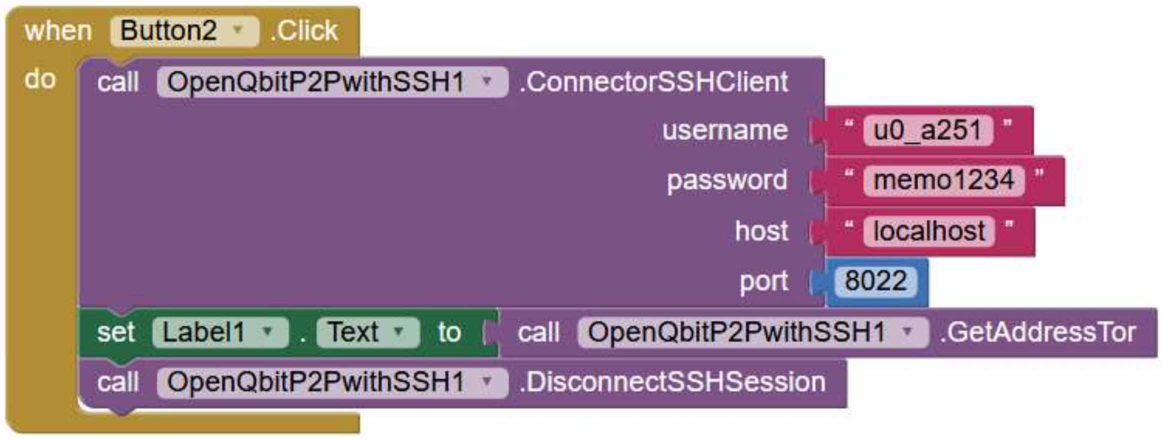
IMPORTANT NOTE: if the command has several parameters, it must be entered with the following syntax between single quotes:

Example:

'cat HellofromMexico.txt**'**

Description: Block to execute any online command over SSH through "Tor" network communication.

Block to **get the** local "Tor" network address (**GetAddressTor**)



when Button2 .Click
do call OpenQbitP2PwithSSH1 .ConnectorSSHClient
username " u0_a251 "
password " memo1234 "
host " localhost "
port 8022
set Label1 . Text to call OpenQbitP2PwithSSH1 .GetAddressTor
call OpenQbitP2PwithSSH1 .DisconnectSSHSession

Mandatory dependency: Use before with Block (**ConnectorSSHClient)**, Block (**DisconnectSSHSession**).

Input parameters: Not applicable - N/A

Output parameters: Will give us the address of the network "Tor" with extension . oinion

Example:

7yfbhadahgvz25fxysojxd6tf7qd7ryhjsboaga7cd6wo4dfnmylkid.onion

Description: Block to execute any online command over SSH through "Tor" network communication.

13. Using the REst API with API-Key in Syncthing

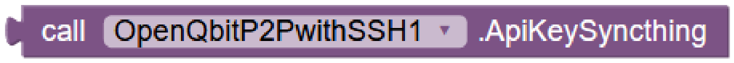
Syncthing has a REst API to use GET / POST in the administration this is an additional support to the extension (**OpenQbitP2PwithSSH**).

In order to use the Rest API we must first know the API-KEY which will give us permission to execute the GET / POST instructions.

To know the API-KEY of the local mobile phone we can obtain it in two ways.

The first one is using the extension (**OpenQbitP2PwithSSH)** with the block (**ApiKeySyncthing**).

Block to get the API-KEY of the local syncthing that is running (**ApyKeySyncthing)**



Mandatory dependency: Block (**ConnectorSSHClient)**, Block (**DisconnectSSHSession**).

Input parameters: **None - N/A**

Output parameters: It does not deliver the content of the configuration file where the API-KEY is located to be used with the Rest API of Syncthing.

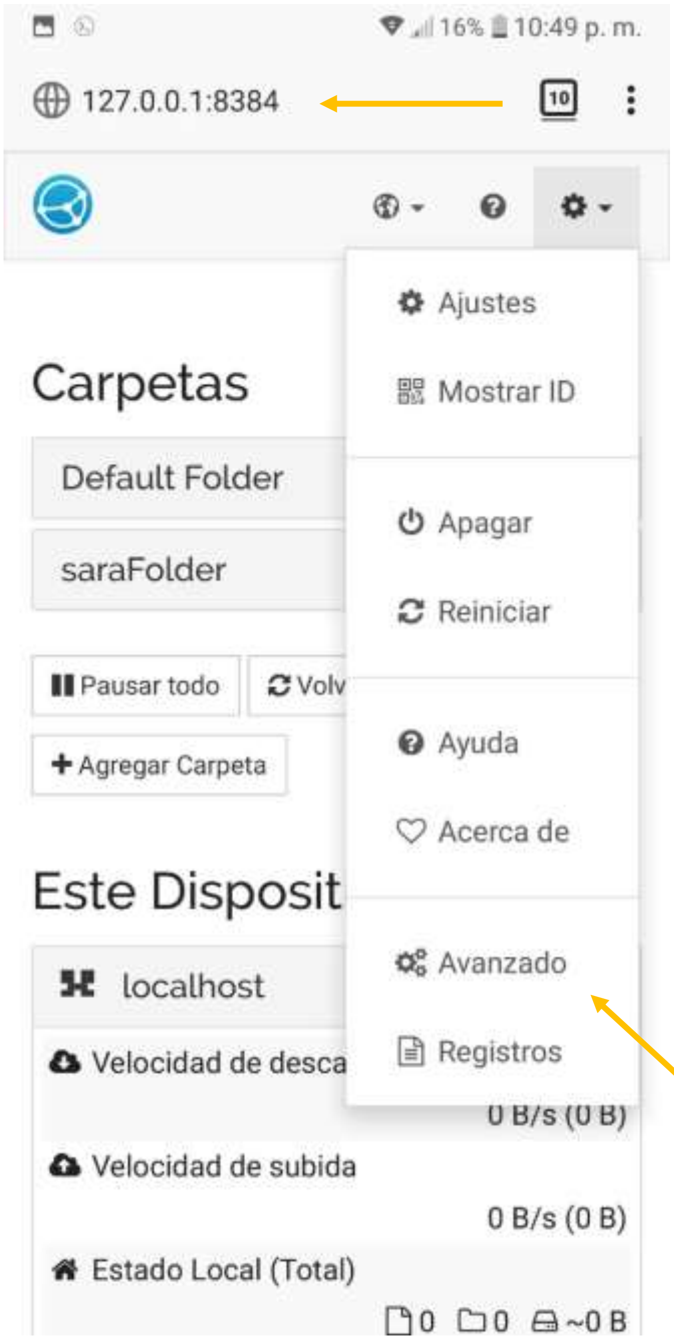
Example:

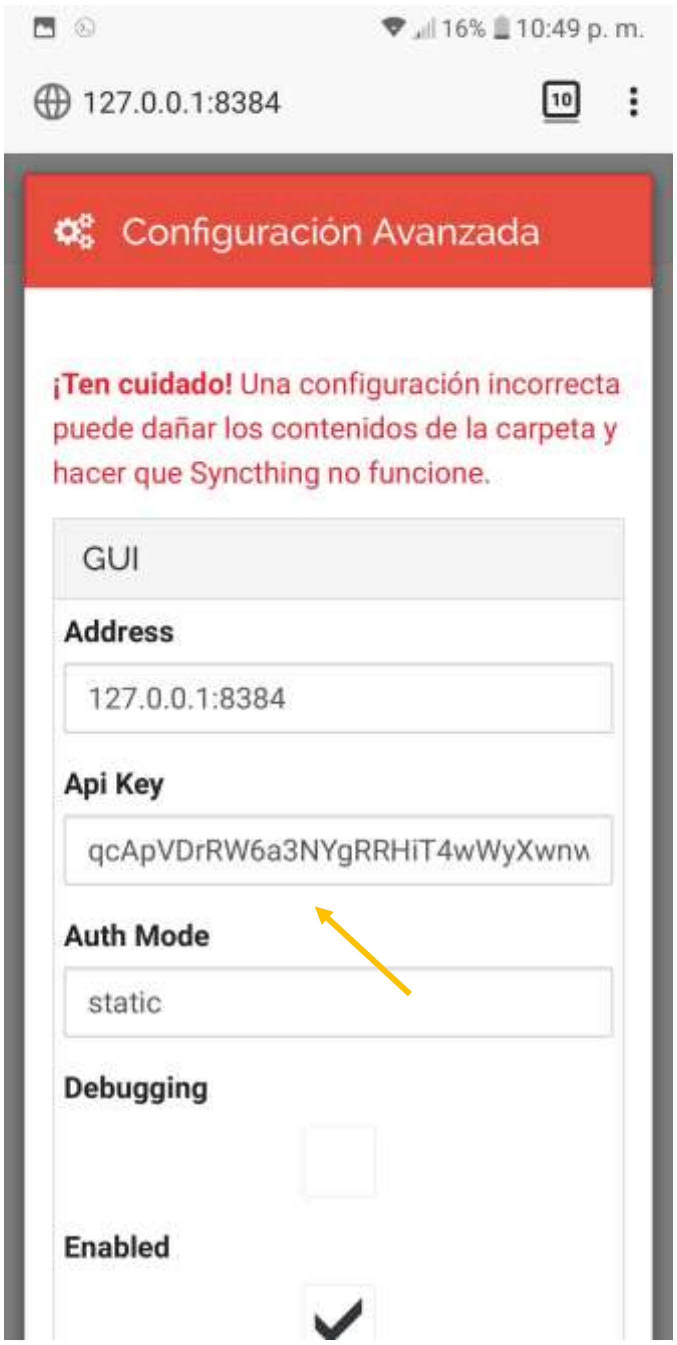
```
[DEFAULT]
name = localhost

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

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

Description: Communication block obtains the content of the syncthing configuration file containing the API-KEY.

The second option is to be able to consult it from the local browser you have in your mobile phone at the default address:

http://127.0.0.1:8384/




16% 10:49 p. m.

127.0.0.1:8384

Configuración Avanzada

¡Ten cuidado! Una configuración incorrecta puede dañar los contenidos de la carpeta y hacer que Syncthing no funcione.

GUI

Address

127.0.0.1:8384

Api Key

qcApVDrRW6a3NYgRRHiT4wWyXwnw

Auth Mode

static

Debugging

☐

Enabled

✓

Once we have the API-KEY we can start using the Rest API from Syncthing we will use an extension from the following website to execute the Curl command.

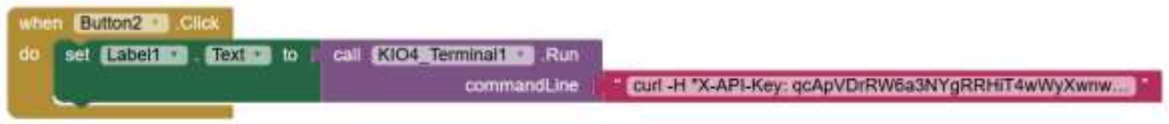
https://puravidaapps.com/extensions.php

And we can use the extension:

Terminal / Shell Extension by Juan Antonio

For example, to get the local phone ID we can execute the following Curl command.

curl -H "X-API-Key: qcApVDrRW6a3NYgRHiT4wWyXwnwwqsG" http://localhost:8384/rest/stats/device



```
when Button2 .Click
do  set Label1 . Text to  call KIO4_Terminal1 .Run
                                commandLine  " curl -H "X-API-Key: qcApVDrRW6a3NYgRRHiT4wWyXwnw... "
```

This will not allow you to execute Curl commands to use the Rest API.

To see all the options of Syncthing go to the next reference of your Rest API.
https://docs.syncthing.net/dev/rest.html

Another tool that will help us with the administration is the direct command line of Syncthing this to we can use with the block to execute commands online through SSH with the block **(CommandLineExecuteSSH).**

The command line options for Syncthing are as follows:

```
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]
```

To see details of each option you can reference your site:

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

Finally, it is important to note that the Rest API and the Syncthing command line are only for support, since these two do not have important functions such as registering a new device ID or removing an already registered ID. That's why we created the extension (**OpenQbitP2PwithSSH**).

14. Licensing and use of 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 and App Inventor Blockly
https://appinventor.mit.edu/about/termsofservice
Apache Ant
https://ant.apache.org/license.html
WGET
https://www.gnu.org/software/wget/

External extensions:
JSONTOOLs
https://thunkableblocks.blogspot.com/2017/07/jsontools-extension.html

Licensing opensource and commercial versions of OpenQbitP2P system consult the official website http://www.openqbit.com

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

OpenQbitP2P is in the public domain.

All code and documentation in OpenQbitP2P has been dedicated to the public domain by the authors. All code authors and representatives of the companies they work for have signed affidavits dedicating their contributions to the public domain and the originals of these affidavits are stored in a safe at OpenQbit Mexico's main offices.

Any person is free to publish, use or distribute the original OpenQbitP2P (OpenQbit) extensions, either as source code or as a compiled binary, for any purpose, commercial or non-commercial, and by any means.

The above paragraph applies to the code and documentation deliverable in OpenQbitP2P, those parts of the OpenQbitP2P library that actually bundle and ship with a larger application. Some scripts used as part of the compilation process (e.g., "configuration" scripts generated by autoconf) may be included in other open source licenses. However, none of these compilation scripts make it into the final OpenQbitP2P deliverable library, so the licenses associated with those scripts should not be a factor in evaluating your rights to copy and use the OpenQbitP2P library.

All deliverable code in OpenQbitP2P has been written from scratch. No code has been taken from other projects or from the open internet. Each line of code can be traced back to its original author, and all those authors have public domain dedications on file. Therefore, the OpenQbitP2P code base is clean and not contaminated with code licensed from other open source projects, not open contribution

OpenQbitP2P is open source, which means you can make as many copies as you want and do what you want with those copies, without limitation. But OpenQbitP2P is not open contribution. To keep OpenQbitP2P in the public domain and ensure that the code is not contaminated with proprietary or licensed content, the project does not accept patches from unknown people. All code in OpenQbitP2P is original, as it has been written specifically for use by OpenQbitP2P. No code has been copied from unknown sources on the Internet.

OpenQbitP2P is in the public domain and does not require a license. Still, some organizations want legal proof of their right to use OpenQbitP2P. The circumstances where this occurs include the following:

- Your company wants compensation for claims of copyright infringement.
- You are using OpenQbitP2P in a jurisdiction that does not recognize the public domain.
- You are using OpenQbitP2P in a jurisdiction that does not recognize an author's right to place his or her work in the public domain.
- You want to have a tangible legal document as evidence that you have the legal right to use and distribute OpenQbitP2P.
- Your legal department tells you that you must buy a license.

If any of the above circumstances apply to you, OpenQbit, the company that employs all OpenQbitP2P developers, will sell you an OpenQbitP2P Title Guarantee. A Title Warranty is a legal document that states that the claimed authors of OpenQbitP2P are the true authors, and that the authors have the legal right to dedicate OpenQbitP2P to the public domain, and that OpenQbit will vigorously defend itself against licensing claims. All proceeds from the sale of the OpenQbitP2P title warranties are used to fund the continuous improvement and support of OpenQbitP2P.

Contributed Code

To keep OpenQbitP2P completely free and royalty free, the project does not accept patches. If you want to make a suggested change and include a patch as a proof of concept, that would be great. However, don't be offended if we rewrite your patch from scratch. The type of non-commercial or opensource license who uses it in this modality and some similar without purchase of support either individual or corporate use no matter the size of the company will be governed by the following legal premises.

Warranty disclaimer. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) "AS IS", **WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either** express or implied, including, without limitation, any warranties or conditions of TITLE, NONINFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the correct use or redistribution of the Work and for assuming any risks associated with your exercise of permissions under this License.

Any financial or other losses incurred by the use of this software will be borne by the affected party. All legal disputes the parties will submit to courts only in the jurisdiction of Mexico City, country Mexico.

For commercial support, use and licensing an agreement or contract must be established between OpenQbit or its corporate and the interested party.

The terms and conditions of distribution marketing may change without notice, please go to the official website www.openqbit.com see any changes to support and licensing clauses non-commercial and commercial.

Any person, user, private or public entity of any legal nature or from any part of the world who simply uses the software accepts without conditions the clauses established in this document and those that can be modified at any time in the portal of www.openqbit.com without previous notice and can be applied at the discretion of OpenQbit in non-commercial or commercial use.

Any questions and information about OpenQbitP2P can be addressed to the App Inventor community or to the various Blockly systems communities as they are: AppBuilder, Trunkable, etc. and/or to the email opensource@openqbit.com for the demand of questions can take 3 to 5 working days to be answered.

Support with commercial use.
support@openqbit.com

Sales for commercial use.
sales@openqbit.com

Legal information and licensing questions or concerns
legal@openqbit.com