Instalación y configuración de OpenHab en un Raspberry PI 3

Contents

[Introducción 3](#_Toc508709815)

[Instalación de OpenHabian 3](#_Toc508709816)

[Descargar la última versión de la imagen de OpenHabian 3](#_Toc508709817)

[Formatear la SD con SDFormatter 3](#_Toc508709818)

[Grabar la imagen en una SD de 8Gb 4](#_Toc508709819)

[Configurar Wifi 4](#_Toc508709820)

[Todo listo 5](#_Toc508709821)

[Configuración de sistema 5](#_Toc508709822)

[Cambio de password del sistema 5](#_Toc508709823)

[Cambio de password Samba y OpenHab 5](#_Toc508709824)

[Instalación de adicionales 7](#_Toc508709825)

[Servidor MQTT (Eclipse Mosquitto) 7](#_Toc508709826)

[Instalación 7](#_Toc508709827)

[Configuración 7](#_Toc508709828)

[Testing 7](#_Toc508709829)

[Referencias 7](#_Toc508709830)

[Instalación de Arping 7](#_Toc508709831)

[¿Qué es Arping? 7](#_Toc508709832)

[¿Para qué sirve? 7](#_Toc508709833)

[Instalación 8](#_Toc508709834)

[Referencias 8](#_Toc508709835)

[Configuración de OpenHab 9](#_Toc508709836)

[Primer acceso 9](#_Toc508709837)

[Habilitar los add-ons 1.x 9](#_Toc508709838)

[Instalar MAP Transform 9](#_Toc508709839)

[Instalar MQTT Bind 9](#_Toc508709840)

[Instalación 9](#_Toc508709841)

[Configuración 10](#_Toc508709842)

[Configurar un MAP 10](#_Toc508709843)

[Agregar ítems MQTT 10](#_Toc508709844)

[Referencias 11](#_Toc508709845)

[Instalar MQTT Action 11](#_Toc508709846)

[Instalar Notify my Android Action 11](#_Toc508709847)

[Instalar HABmin 11](#_Toc508709848)

[Instalar Network Binding 11](#_Toc508709849)

[Configuración 11](#_Toc508709850)

[Referencias 12](#_Toc508709851)

[Instalar MapDB Service 12](#_Toc508709852)

[Configuración 12](#_Toc508709853)

[Logueo 13](#_Toc508709854)

[Referencias 13](#_Toc508709855)

[Automación por reglas 14](#_Toc508709856)

[Introducción 14](#_Toc508709857)

[Crear una regla 14](#_Toc508709858)

[Ejemplo de regla 14](#_Toc508709859)

[Referencias 14](#_Toc508709860)

[Consola OpenHab (Karaf) 14](#_Toc508709861)

# Introducción

El motivo inicial de este proyecto fue poder construir la automatización de un hogar. Luego de investigar las opciones se decidió utilizar un sistema abierto (open source) que pudiera integrar la automatización a desarrollar, así como los artefactos inteligentes que puedan existir en la misma locación.

Con esto en mente se decidió utilizar **OpenHab** corriendo en un **Raspberry Pi** con un servidor **MQTT** y dispositivos **ESP** necesarios, donde el **Raspberry** brinda la infraestructura necesaria, **OpenHab** funciona como unidad (sistema) de integración y los módulos **ESP** controlan los artefactos que se deseen automatizar, monitorear o controlar remotamente.

Como **SO** del **Raspberry** **Pi** se va a utilizar **OpenHabian** que tiene las siguientes bondades frente a otras distros:

* Ya cuenta con **OpenHab** preinstalado
* Tiene una buena tool de personalización con la que se pueden instalar varios adicionales tales como:
  + Mosquitto (broker MQTT)
  + Grafana
  + Log Viewer
* Viene con Samba pre instalado
* Tiene una opción para hacer backups en USB

La única contra es que no tiene interfaz gráfica preinstalada si es que queremos acceder por esa vía al sistema.

# Instalación de OpenHabian

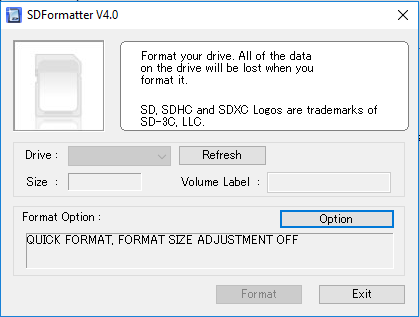
## Descargar la última versión de la imagen de OpenHabian

[Releases Openhabian](https://github.com/openhab/openhabian/releases)

Si el archivo tiene extensión XZ, descomprimirlo con **7Zip** (<http://www.7-zip.org/>)

## Formatear la SD con SDFormatter

[SD Formatter](https://www.sdcard.org/downloads/formatter_4/)

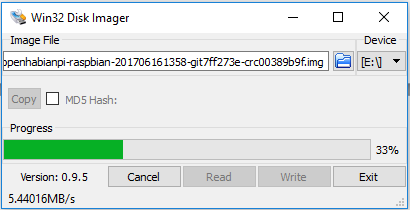


## Grabar la imagen en una SD de 8Gb

Es recomendable este tamaño porque con SD de mayor capacidad tuve problemas de compatibilidad.

Usar **Win32DiskImager**

[Win32 Disk Imager](https://sourceforge.net/projects/win32diskimager/)



## Configurar Wifi

Explorar la SD y editar el archivo **openhabian.conf** configurando el **wifi\_ssid** y el **wifi\_psk**

## Todo listo

Colocar la SD en el Raspberry y encenderlo.

El primer boot demora varios minutos (dependiendo de la velocidad de conexión entre 5 y 60).

## Configuración de sistema

La distro **OpenHabian** viene con una herramienta de configuración. Para abrirla ingresar:

sudo openhabian-config

Desde esta aplicación se pueden hacer varias cosas:

* Instalar adicionales
* Upgrade del sistema
* Backups
* Cambiar settings de sistema
* etc

## Cambio de password del sistema

Ingresar el comando

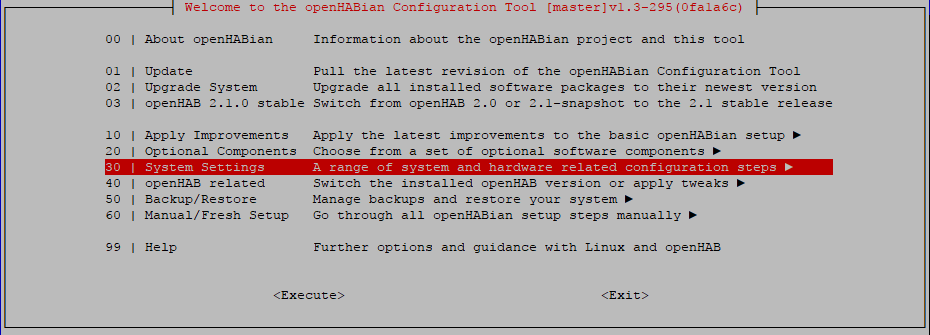
passwd

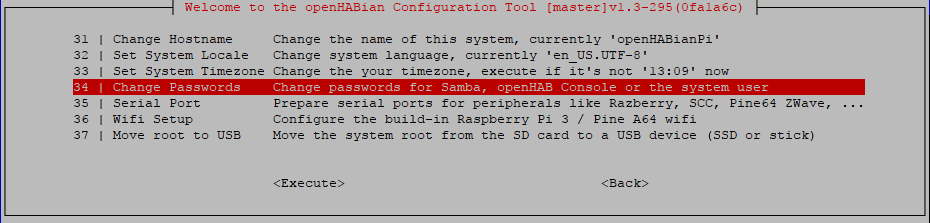
Seguir el prompt a continuación.

## Cambio de password Samba y OpenHab

Desde la herramienta de configuracion ingresar al menu de cambios de passwords.

sudo openhabian-config





# Instalación de adicionales

## Servidor MQTT (Eclipse Mosquitto)

### Instalación

1. Ingresar

sudo openhabian-config

1. Seleccionar la opción **Optional Components**.
2. Seleccionar **Mosquitto**
3. Configurar la password de **Mosquitto** (opcional). Si se cancela este paso, **Mosquitto** opera sin seguridad lo cual facilita la configuración del add-on que se instale en **OpenHab**. Se puede configurar una password más adelante.

### Configuración

Para modificar la configuración, editar **mosquitto.conf**:

sudo nano /etc/mosquitto/mosquitto.conf

No es necesario modificar nada para que el servidor funcione.

Por default, mosquitto escucha en: HOST\_IP:1883 (host\_ip es la IP que se asignó al raspeberry)

### Testing

Para probarlo, en dos consolas diferentes escribir estos comandos

1. mosquitto\_sub -v -t 'test/topic'
2. mosquitto\_pub -t 'test/topic' -m 'helloWorld'

En la consola (A) debería aparecer el mensaje 'helloWord'

### Referencias

[Mosquitto](https://mosquitto.org/man/mosquitto-conf-5.html)

## Instalación de Arping

### ¿Qué es Arping?

Arping es una herramienta que permite hacer pings ARP (address resolution protocol). Es un protocolo no ruteado (sirve solo entre dispositivos conectados a una misma subred) utilizado para resolver la asociación entre direcciones físicas (MAC) y direcciones IP.

### ¿Para qué sirve?

Dentro de OpenHab, Arping sirve para poder detectar los teléfonos.

Los teléfonos suelen entrar en un modo *deep sleep* después de un periodo de inactividad y en ese estado no reaccionan a los pings convencionales (ICMP pings) pero si lo hacen a pings ARP. Las condiciones para que OpenHab pueda detectar un teléfono usando Arping son:

1. Que el teléfono tenga habilitado el wifi
2. Que se haya configurado para conectarlo a la red wifi de la casa
3. Que no tenga habilitado la opción “Disable wifi in stand by”

### Instalación

sudo apt-get install Arping

Para que OpenHab pueda hacer uso de Arping es necesario otorgar permisos.

sudo chmod u+s /usr/sbin/arping

### Referencias

[Ping ICMP vs ARP](https://www.linux.com/news/ping-icmp-vs-arp)

[Arping](https://en.wikipedia.org/wiki/Arping)

# Configuración de OpenHab

## Primer acceso

Una vez finalizada la instalación de **OpenHabian** y adicionales acceder a **OpenHab** a través de la url.

<http://openhabianpi:8080>

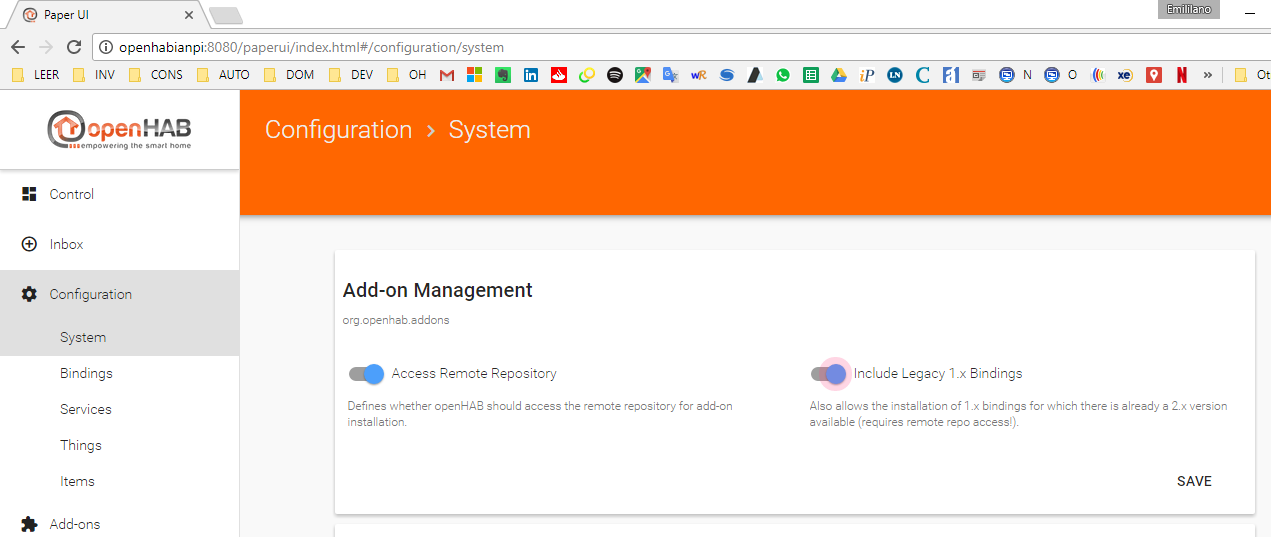
Una vez ahí seleccionar **Standard Setup**. Esto solo se realiza en el primer acceso al sistema y es donde se instalan algunas de las interfaces de acceso.

Luego a medida que se vaya tomando conocimiento del sistema, se pueden instalar mediante otros medios, diferentes interfaces.

## Habilitar los add-ons 1.x

Algunos add-ons deseables (como el de MQTT) fueron desarrollados para la versión 1.x de **OpenHab**, por lo tanto, es necesario activar esta propiedad para poder instalarlos.

Desde Paper UI (<http://openhabianpi:8080/paperui/index.html#/configuration/system>) habilitar los add-ons 1.x



## Instalar MAP Transform

Instalar desde <http://openhabianpi:8080/paperui/index.html#/extensions>

## Instalar MQTT Bind

Para que este add-on pueda instalarse, es necesario el paso de habilitar los add-ons 1.x

### Instalación

Instalarlo desde <http://openhabianpi:8080/paperui/index.html#/extensions>

### Configuración

Es necesario configurar el broker MQTT al cual se va a apuntar. Pueden configurarse algunas cosas adicionales como:

* Calidad de servicio
* Seguridad
* Id de cliente, etc.

Para modificar la configuración:

1. Ingresar   
   nano /etc/openhab2/services/mqtt.conf
2. Descomentar y completar el <broker>.url: mosquitto.url=tcp://localhost:1883  
   Ingresamos localhost porque el server mosquitto está corriendo en el mismo host que **OpenHab** y **1883** como puerto porque es el default. Si se hubiera configurado otro puerto en la configuración de MQTT ese mismo debería ir acá también.
3. Descomentar y completar el <broker>.clientId: mosquitto.clientId=openhab.mqtt (opcional)

### Configurar un MAP

Este map va a servir para traducir los estados entre los **ESP** y **OpenHab** vía **MQTT**

1. Ingresar

touch /etc/openhab2/transform/onoff.map

1. Editar onoff.map agregando las siguientes dos líneas  
   0=OFF  
   1=ON

### Agregar ítems MQTT

1. Ingresar

touch /etc/openhab2/items/mqtt.items

1. Editar mqtt.items  
   Switch ESP01 {mqtt="<[mosquitto:esp/125603/state:state:MAP(onoff.map)],>[mosquitto:esp/125603/command:command:OFF:0],>[mosquitto:esp/125603/command:command:ON:1]", autoupdate="false"}

**Explicación:**

* **Switch** indica que el tipo de ítem configurado es justamente del tipo interruptor.
* **ESP01** indica el nombre del ítem
* **Mqtt** indica en inicio de la configuración de mensajes del protocolo. En el ejemplo hay tres mensajes configurados, dos salientes (inician con **>**) y un entrante (inicia con **<**).
  + **Saliente:** Son dos mensajes:
    - Uno indica que cuando desde alguna interfaz se haga clic en el ítem y el estado del mismo pase a **OFF**, se publique en el tópico **esp/125603/command** un **0** (cero).
    - El otro indica lo mismo para el caso de que el estado pase a **ON**, se deberá enviar un **1**.
  + **Entrante:** El mensaje entrante dice que cuando se reciba un mensaje en el tópico **esp/125603/state** se use el servicio de transformación con la transformación definida en **onoff.map**. En esa transformación se traduce 0 > OFF y 1 > ON, y el estado obtenido de dicha transformación se seteará en el ítem (switch)

### Referencias

[MQTT Binding](https://github.com/openhab/openhab1-addons/wiki/mqtt-binding)

## Instalar MQTT Action

Sirve para publicar mensajes a un broker mqtt desde OpenHab. Util por ekemplo, para publicar mensajes desde una regla. Ver Automación por reglas.

Instalar desde **Paper UI > Add-ons > Actions > MQTT Action**

## Instalar Notify my Android Action

Instalar desde **Paper UI > Add-ons > Actions > Notify my Android Action**

Para obtener la key loguearse en <https://www.notifymyandroid.com/account.jsp> y acceder a la solapa **Manage API Keys**.

## Instalar HABmin

Instalar desde **Paper UI > Add-ons > User Interfaces > HABmin**

## Instalar Network Binding

Permite detectar los dispositivos “pingueables” dentro de la red.

Sirve para detectar presencia (cuando un dispositivo se presenta en la red) o verificar la salud de la red (si un dispositivo esta activo).

### Configuración

Para que funcione el DHCP Listen, es necesario hacer un *port forwarding* del puerto 67 (puerto DHCP) al puerto 6767 (puerto donde OpenHab escucha por default)

sysctl -w net.ipv4.ip\_forward**=**1

iptables -A INPUT -p udp --dport 6767 -j ACCEPT

iptables -t nat -A PREROUTING -p udp --dport 67 -j REDIRECT --to-ports 6767

### Referencias

[Network Binding](https://docs.openhab.org/addons/bindings/network/readme.html)

## Instalar MapDB Service

MapDB es un servicio de persistencia que guarda el ultimo valor de un ítem. Es particularmente útil para poder recuperar el ultimo estado de un ítem luego de una baja de alimentación.

Instalar desde **Paper UI > Add-ons > Persistence > MapDB**

### Configuración

MapDB es uno de los varios servicios de persistencia que se pueden usar en **OpenHab** (se puede usar más de uno en simultaneo).

**Para configurarlo se debe:**

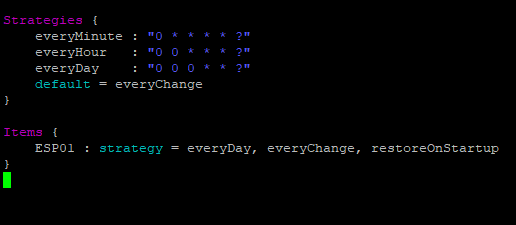
1. Modificar el archivo **/etc/openhab2/services/mapdb.cfg**.

sudo nano /etc/openhab2/services/mapdb.cfg

Esta configuración no es necesaria porque los tópicos de configuración son dos y los valores default son los que mejor aplican en la mayoría de los escenarios.

1. Crear el archivo **/etc/openhab2/persistence/mapdb.persist**

sudo nano /etc/openhab2/persistence/mapdb.persist

En este archivo se configuran las estrategias de persistencia para los ítems del sistema. Un ejemplo sería:

### Logueo

Para configurar el logueo, desde la consola (Karaf) ingresar el siguiente comando:

log:set LEVEL org.openhab.persistence.mapdb

Donde LEVEL puede ser cualquier nivel de logueo que se desee.

### Referencias

[Estrategias de persistencia](https://docs.openhab.org/configuration/persistence.html)

[MapDB](https://docs.openhab.org/addons/persistence/mapdb/readme.html)

## Automación por reglas

### Introducción

Se puede desarrollar automatización usando **OpenHab** a través de la creación de reglas.

Se pueden crear reglas desde la interfaz HABmin o escribiéndolas manualmente.

Si bien desde HABmin parece más sencillo, la funcionalidad no es muy robusta (al menos hasta la versión 2.2 de OpenHab) por tal motivo es recomendable hacerlo manualmente aunque requiera manejo de sintaxis.

### Crear una regla

Para crear una regla manualmente:

1. Crear un archivo <nombre\_regla>.rules en /etc/openhab2/rules  
    nano /etc/openhab2/rules/myRule.rules
2. Editar el archivo y escribir en el la regla deseada

### Ejemplo de regla

rule "riego"

when

Time cron "0 0 21 \* \* ?"

then

notifyMyAndroid("5bc56db507eabfe3fb26b4e97ebddbcc75eaa46d370127e2", "Riego", $

publish("mosquitto","irrigation/valve01/cmd","1");

end

### Referencias

**Documentación**

<https://github.com/openhab/openhab1-addons/wiki/rules>

**Ejemplos**

<https://github.com/openhab/openhab1-addons/wiki/Samples-Rules>

**Logging en reglas**

<https://docs.openhab.org/administration/logging.html#create-log-entries-in-rules>

**XTend**

<https://www.eclipse.org/xtend/documentation/index.html>

# Consola OpenHab (Karaf)

1. Ingresar ssh -p 8101 openhab@localhost (puede dar timeout, volver a intentar)
2. Ingresar “yes” para aceptar el certificado
3. Password: habopen