6. (Opcional) Comunicación de los datos utilizando el broker de MQTT de HiveMQ

¡¡NIVEL MEDIO!!

Para ejecutar este tutorial se debe utilizar la versión 1 del IDE de Arduino.

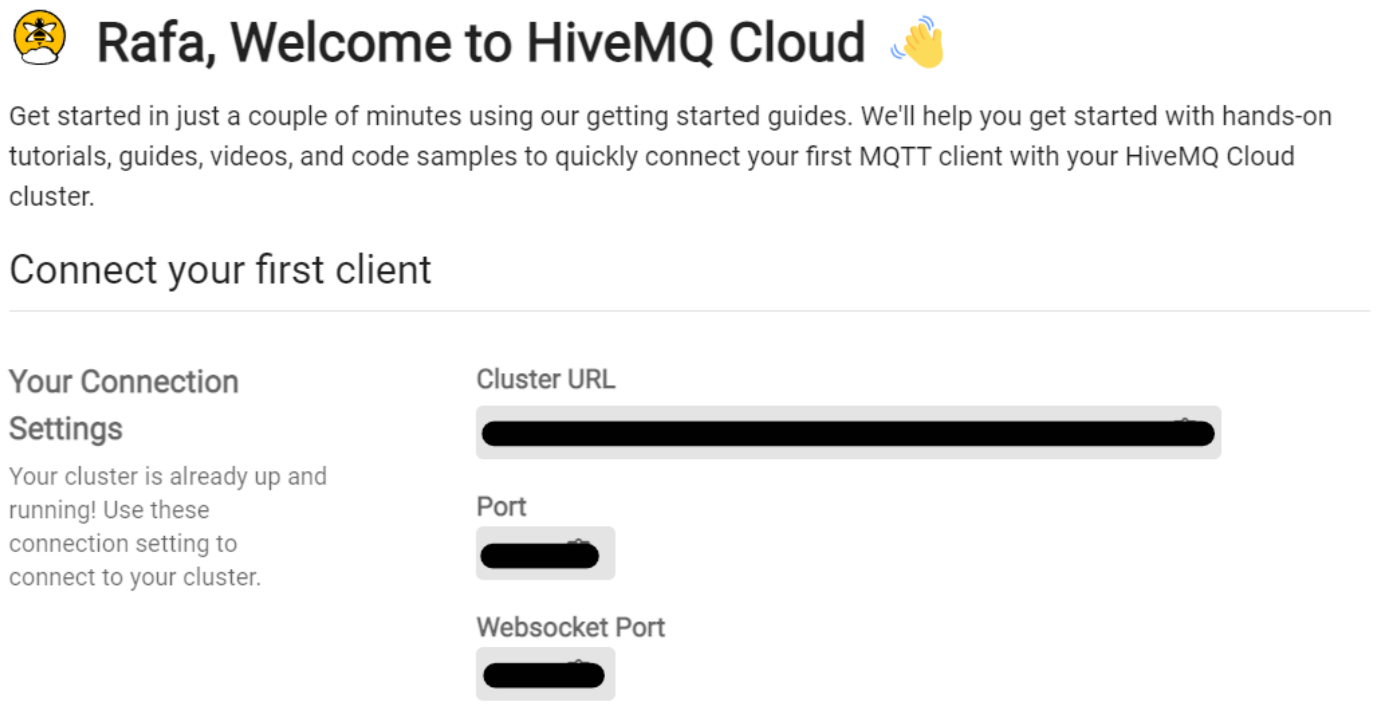
¿Porque HiveMQ Cloud?

Un agente gratuito de Cloud MQTT que le permite conectar hasta 100 dispositivos.

A free Cloud MQTT Broker that enables you to connect up to 100 devices.

https://www.hivemq.com/downloads/

Registrarse en el enlace anterior. Una vez registrado, nos aparece las siguientes opciones, los primeros datos hacen referencia a los datos de conexión del servidor.



Rafa, Welcome to HiveMQ Cloud 👋

Get started in just a couple of minutes using our getting started guides. We'll help you get started with hands-on tutorials, guides, videos, and code samples to quickly connect your first MQTT client with your HiveMQ Cloud cluster.

Connect your first client

Your Connection Settings

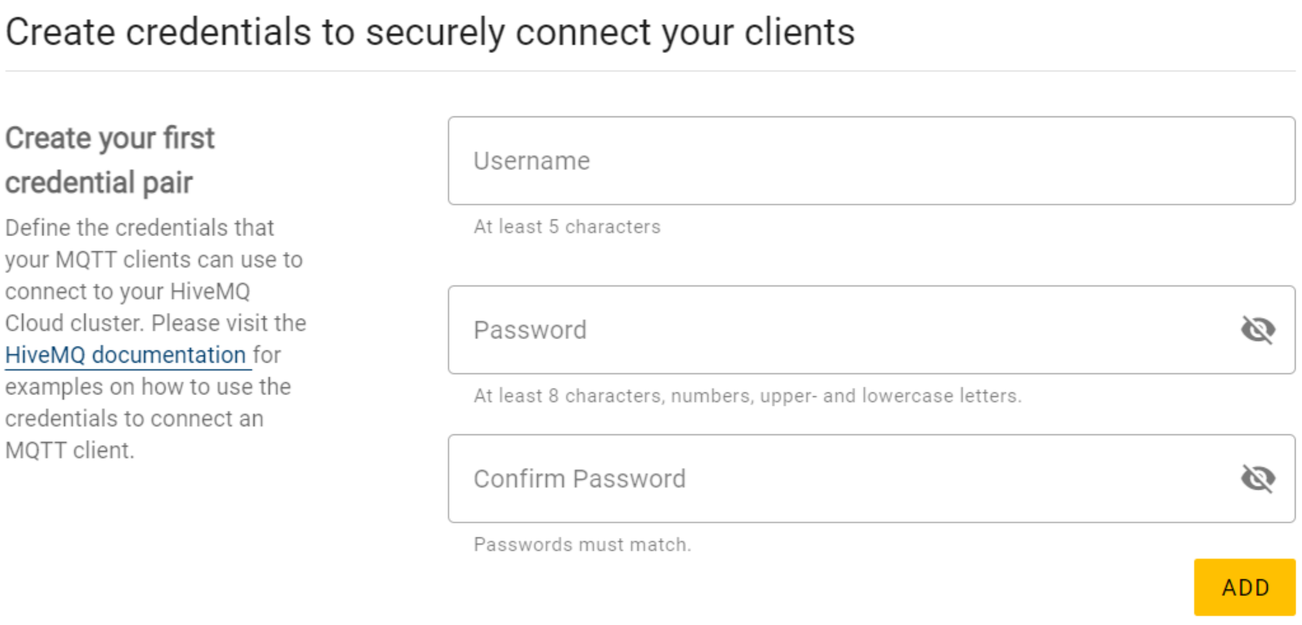
Your cluster is already up and running! Use these connection setting to connect to your cluster.

Cluster URL
[redacted]

Port
[redacted]

Websocket Port
[redacted]

El segundo apartado trata de rellenar los datos de conexión con un usuario y una contraseña robusta.



Create credentials to securely connect your clients

Create your first credential pair

Define the credentials that your MQTT clients can use to connect to your HiveMQ Cloud cluster. Please visit the HiveMQ documentation for examples on how to use the credentials to connect an MQTT client.

Username

At least 5 characters

Password

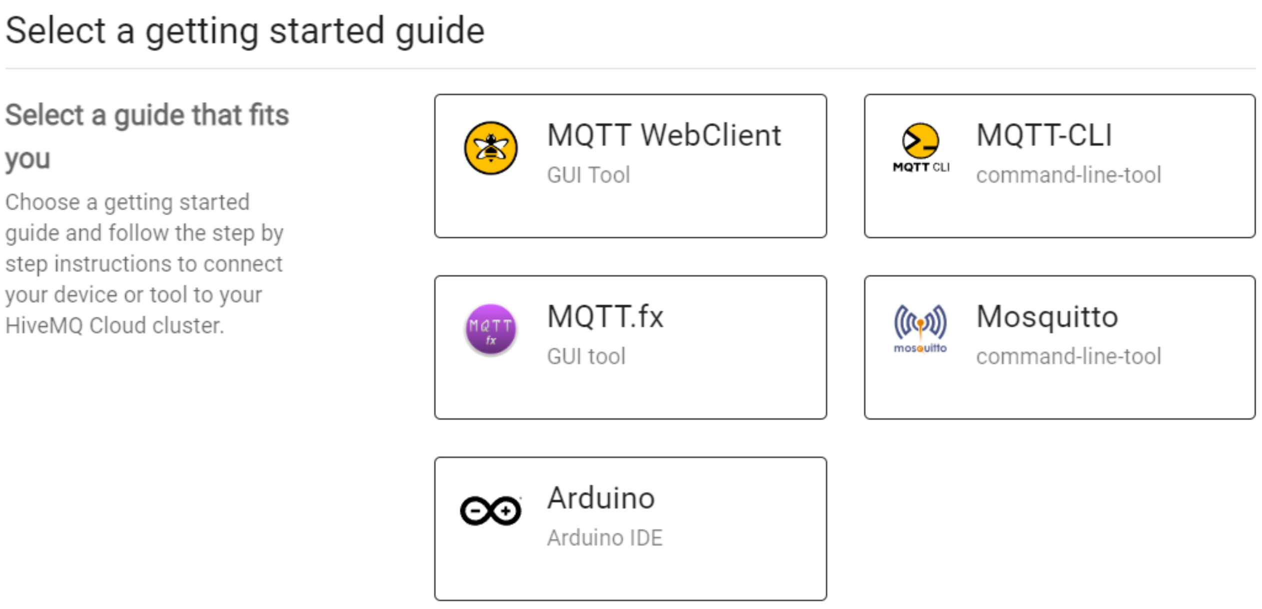
At least 8 characters, numbers, upper- and lowercase letters.

Confirm Password

Passwords must match.

ADD

Por último, accede al tutorial de arduino, justamente un tutorial para ESP8266.



Select a getting started guide

Select a guide that fits you

Choose a getting started guide and follow the step by step instructions to connect your device or tool to your HiveMQ Cloud cluster.

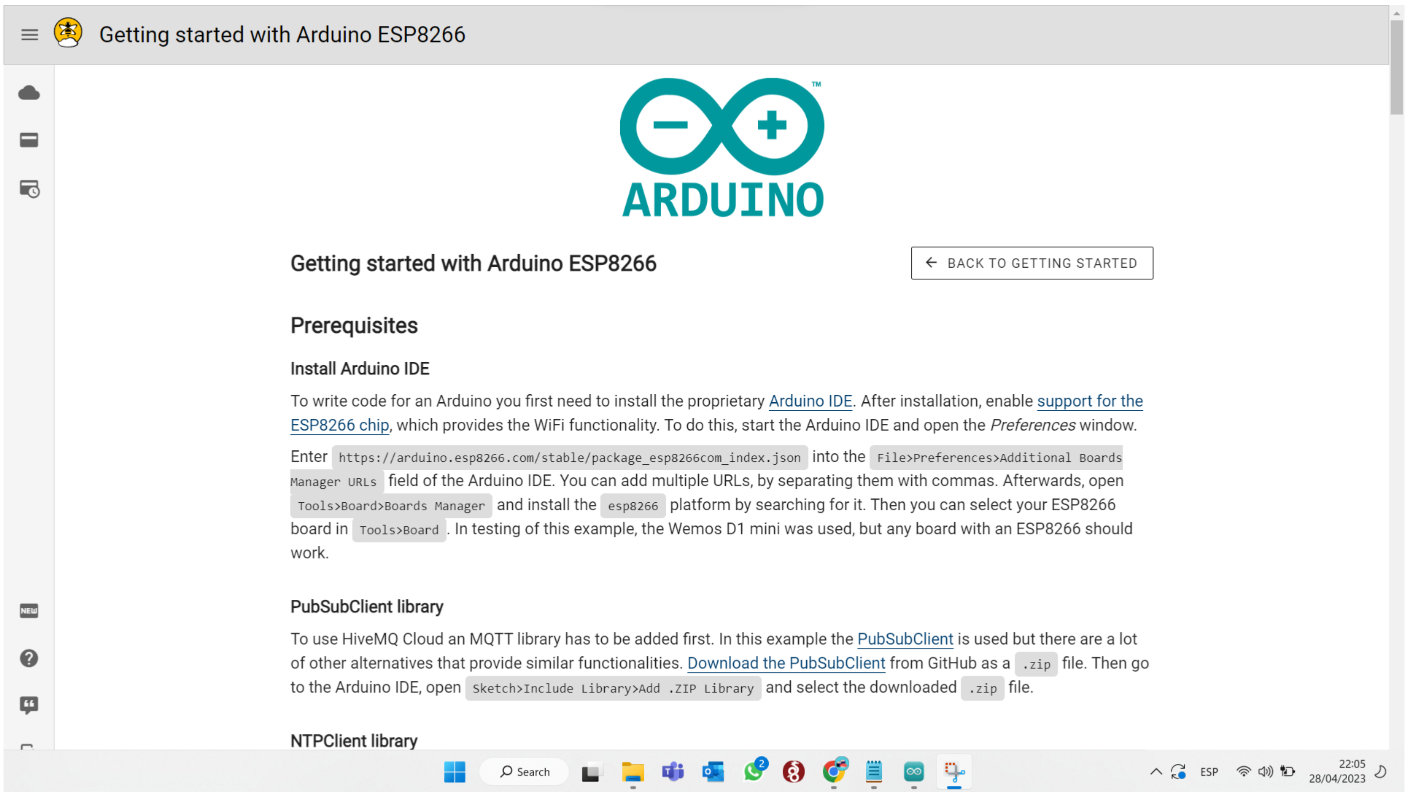
MQTT WebClient
GUI Tool

MQTT-CLI
command-line-tool

MQTT.fx
GUI tool

Mosquitto
command-line-tool

Arduino
Arduino IDE



Getting started with Arduino ESP8266

Prerequisites

Install Arduino IDE

To write code for an Arduino you first need to install the proprietary Arduino IDE. After installation, enable support for the ESP8266 chip, which provides the WiFi functionality. To do this, start the Arduino IDE and open the *Preferences* window.

Enter `https://arduino.esp8266.com/stable/package_esp8266com_index.json` into the `File>Preferences>Additional Boards Manager URLs` field of the Arduino IDE. You can add multiple URLs, by separating them with commas. Afterwards, open `Tools>Board>Boards Manager` and install the `esp8266` platform by searching for it. Then you can select your ESP8266 board in `Tools>Board`. In testing of this example, the Wemos D1 mini was used, but any board with an ESP8266 should work.

PubSubClient library

To use HiveMQ Cloud an MQTT library has to be added first. In this example the PubSubClient is used but there are a lot of other alternatives that provide similar functionalities. Download the PubSubClient from GitHub as a `.zip` file. Then go to the Arduino IDE, open `Sketch>Include Library>Add .ZIP Library` and select the downloaded `.zip` file.

NTPClient library

NTPClient library

The next library that is required is the NTPClient. Install this library by opening `Tools>Manage Libraries...` and searching for `NTPClient`. This library provides functionality for date and time. Here it is used to update the internal clock of the ESP8266 board, in order to use TLS certificates.

LittleFS Filesystem Uploader

The LittleFS Filesystem Uploader is needed to upload certificates to your board. This enables you to connect securely with HiveMQ Cloud. Download the `ESP8266LittleFS-X.zip`. Go to the Arduino IDE directory and open the `tools` folder. Unzip the downloaded `.zip` folder to the `tools` folder. The result will be a folder called `ESP8266LittleFS-2.6.0` or it can have a different version. This folder contains the `ESP8266LittleFS` folder. Cut the `ESP8266LittleFS` folder and paste it into the `tools` folder. Afterwards, you can delete the `ESP8266LittleFS-2.6.0` folder. The resulting folder structure should look like this:

```
<home_dir>/Arduino-<version>/tools/ESP8266LittleFS/tool/esp8266littlefs.jar
```

On OS X create the `tools` directory in `~/Documents/Arduino/` and unpack the files there. Now restart Arduino IDE and `ESP8266 LittleFS Data Upload` will appear when opening the `tools` menu.

EXAMPLE FILES - CERTS.AR AND SKETCH

Upload certificates to the board

The next step is to get the certificates that are needed for an encrypted connection to HiveMQ Cloud. For this purpose use this script by downloading the repository as a `.zip` folder. Only the file `certs-from-mozilla.py` is needed, you can delete the rest. Open this script in a Python IDE of your choice, for example PyCharm or Visual Studio Code. Execute the script in your IDE. A file will be generated, that is called `certs.ar`. Move this file into the directory of your Arduino IDE sketch. To find it you can go to `Sketch>Show Sketch Folder` in Arduino IDE. The folder where your sketch is saved should open. Inside this folder, create a new folder called `data`. Put your generated `certs.ar` into this `data` folder. In the Arduino IDE, in the `Tools` menu, you may want to select a bigger flash size, this depends on the size of your files. Then open `Tools>ESP8266 LittleFS Data Upload`. This will upload the files located in `data` to your board's flash memory. After a few seconds, you should get the message `LittleFS Image Uploaded`.

Connect MQTT Clients

Use the code below to connect to your HiveMQ Cloud instance. First add your WiFi information, so that the ESP8266 can connect to the internet. The `host name` of your cluster is already inserted as `mqtt_server`. Your `username` is also inserted automatically. To fully verify your credentials, replace the variable `"your_password"` with the value you entered when creating your credentials. As HiveMQ Cloud does not support insecure connections, TLS is required. A secure connection will be established by using the certificates we downloaded earlier. The default port used for secure MQTT connections is `8883`.

https://create.arduino.cc/editor/racarla96/49008de2-433a-46ce-bd77-ec90e24d438e/preview?embed

Run this example by connecting your board to your PC via a USB-cable and uploading the sketch with the button in the Arduino IDE.

Publish and Subscribe with your MQTT client

The code shown above creates an MQTT client that is publishing and subscribing to a topic on your HiveMQ Cloud cluster. First it sets up the WiFi by using the SSID and password you set. Then the internal clock of the board is updated by getting the present time from an NTP server. This is necessary to validate the certificates that were uploaded to the board with the LittleFS Filesystem Uploader. For `callbacks`, `Message arrived` is registered. The LED of the board will also flash when a non-empty message arrives. After `setup_wifi()` and `setDateTime()`, the certificate store (`certStore`) is being initialized and made to store the certificates of `certs.ar`. Then the WiFi client will be integrated with the `certStore`. It sets up the MQTT client by using the earlier set host name and port. Afterwards it connects to the cluster using your username and password in the `reconnect()` function, that gets executed in a loop. It subscribes to the topic `"testTopic"` to which it also publishes. Then it publishes a message with the payload `"hello world"` and incrementing numbers to this topic.

Next Steps

Get familiar with the Arduino IDE API and build your first application. Further information on the PubSubClient for Arduino can be found in our MQTT Client Library Encyclopedia. Learn more about MQTT by visiting the MQTT Essentials guide, that explains the core of MQTT concepts, its features and other essential information.

Revision #8
Created 28 April 2023 19:51:02 by Rafa
Updated 29 April 2023 18:42:37 by Rafa