



Taller IOT: Introducción a LoRa[®], LoRaWAN[™] y The Things Network



















ALEJANDRO JUAN GARCÍA
 @ALEXCORVIS84

 @MAKERSASTURIAS
[HTTP://MAKERSASTURIAS.COM/](http://makersasturias.com/)

⚠️ DISCLAIMER ⚠️

- ESTO NO ES MI TRABAJO ✖️ 💰 ¡ES LO QUE ME APASIONA!
- OPINIONES PROPIAS BASADAS EN LA FILOSOFÍA 'JUAN PALOMO' 🧐 ⚡
- HE APRENDIDO GRACIAS A LA AYUDA DE OTROS 👥
- NO ES UN ANUNCIO O PROMOCIÓN DE NINGUNA COMPAÑÍA 🗣️ ✖️
- MATERIAL DISPONIBLE BAJO LICENCIA CC BY-NC-SA 📖
- MATERIAL USADO OPEN HARDWARE 🔧 ❤️ y OPEN SOURCE ⚙️ ❤️
- MATERIAL BASADO EN RECURSOS EXTERNOS Y PROPIOS 📚 🌐
- INTENTAR DIVULGAR LA CULTURA MAKER Y FILOSOFÍA DIY 🔧 🧐

¿Qué vamos a aprender?

- CONCEPTOS (30 minutos 
 - ¿Qué es LoRa®?   
 - ¿Qué son las LPWAN? 
 - ¿Qué es LoRaWAN™?   
 - ¿Qué es The Things Network?  
- TALLER  (1 Hora 
 - Montaje y configuración Gateway Monocanal   
 - Montaje y configuración de Nodos LoRa   

CONCEPTOS



SÉ BUENO Y AGUANTA LA TEORÍA. TODO TIENE SU POR QUÉ 😊
“NO ENTIENDES ALGO A MENOS QUE SEAS CAPAZ DE
EXPLICÁRSELO A TU ABUELA” A.EINSTEIN

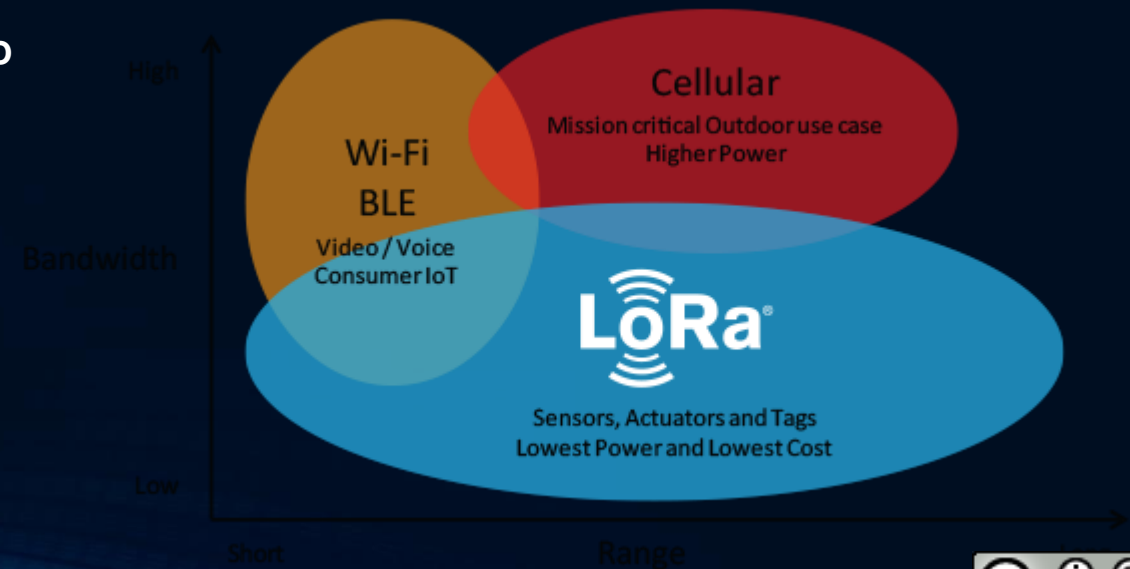
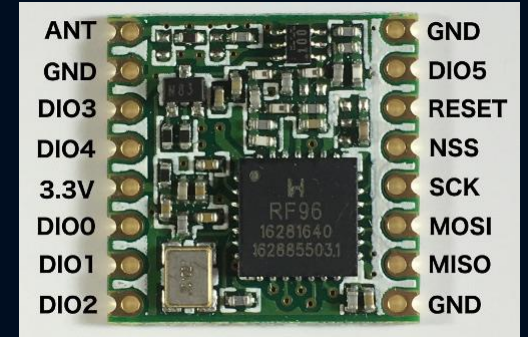


¿Qué es LoRa®? 🤖

LoRa®

- Tecnología de transmisión inalámbrica (capa física)
- Abreviatura de **Long** (largo) **Range** (alcance)
- Basada en la técnica modulación **CSS** (*Chirp Spread Spectrum*)
- Adquirida y **patentada** por la Empresa **Semtech** en 2012
- **Bajo consumo** de energía (vida de las baterías de 3 a 10 años)
- Primera solución de **bajo coste** de implementación para uso comercial
- Módulos de bajo coste (<5€ HopeRF 95)
- Gran **inmunidad** y **robustez** frente a **interferencias** y **ruido**
- Gran cobertura:
 - $\leq 2\text{km}$ en zonas urbanas (NLO)
 - $\geq 20\text{km}$ en zonas con visión directa (LOS)

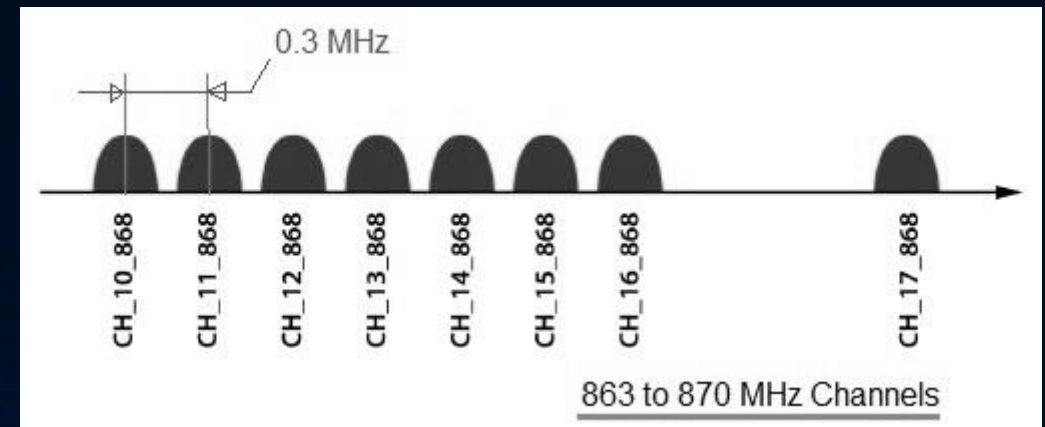
<https://www.semtech.com/lora>

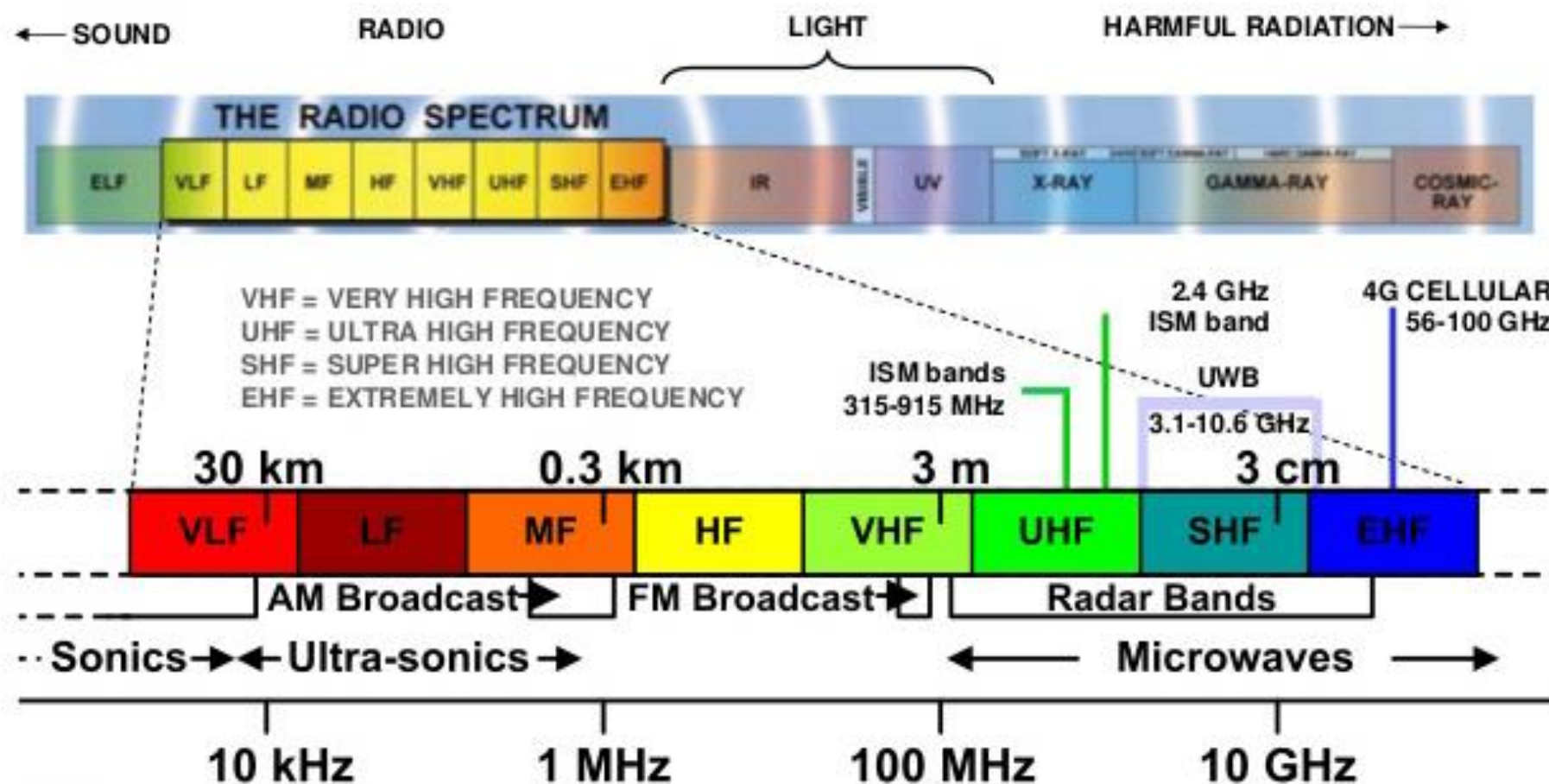


LoRa®

- Bandas **ISM** (Industrial, Scientific & Medical)
 - Uso libre sin licencia pero limitadas en potencia y tiempo de transmisión.
- Frecuencias definidas mundialmente:
 - **Europa 868Mhz**
 - América 915Mhz
 - Asia 920Mhz/433Mhz
- Dentro de las bandas, en cada región se definen unos canales:
 - En **Europa** existen **10 canales** y para América 64
- **Limitaciones** para Europa:
 - Potencia Máxima Transmisión 25mW (**14dBm**)
 - Tiempo de transmisión 1%

	Europe	North America
Frequency band	867-869MHz	902-928MHz
Channels	10	64 + 8 + 8
Channel BW Up	125/250kHz	125/500kHz
Channel BW Dn	125kHz	500kHz
TX Power Up	+14dBm	+20dBm typ (+30dBm allowed)
TX Power Dn	+14dBm	+27dBm
SF Up	7-12	7-10
Data rate	250bps- 50kbps	980bps-21.9kbps
Link Budget Up	155dB	154dB
Link Budget Dn	155dB	157dB



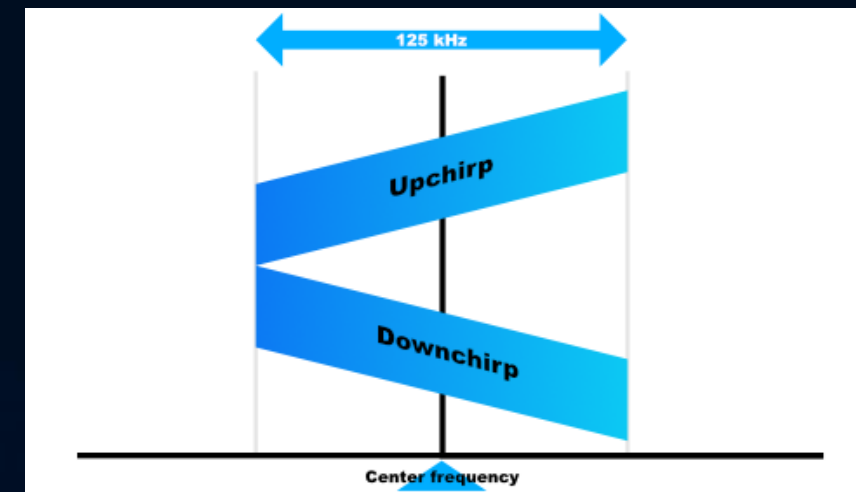
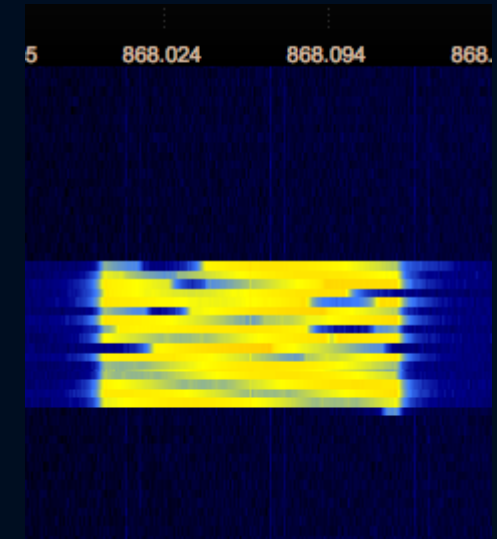


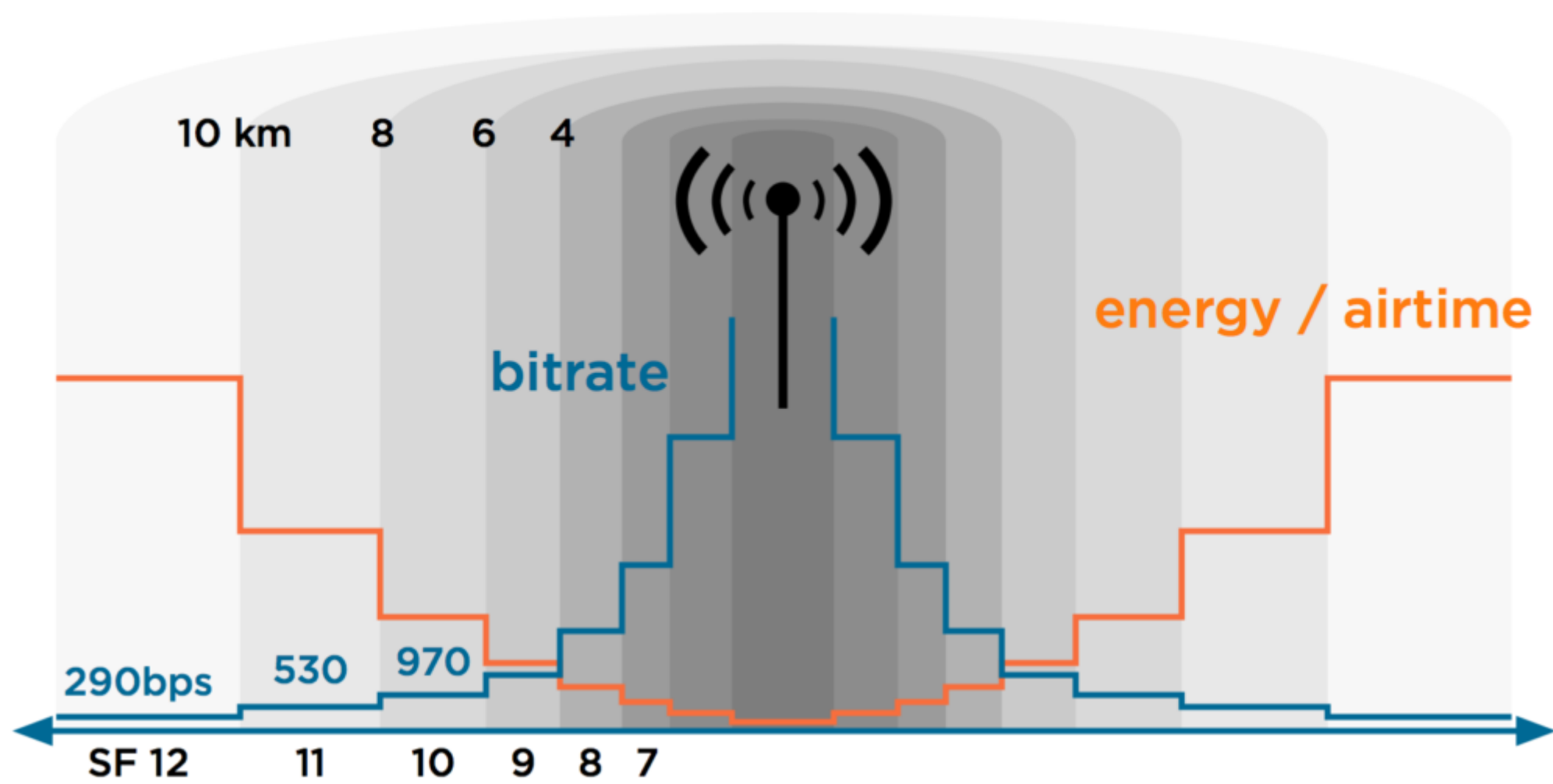
Source: JSC.MIL

LoRa®

- **SF (Spreading Factor – Factor de esparcimiento)**
 - Factor que define el tiempo (en el aire) de envío/energía para la transmisión de datos
 - LoRa trabaja con factores que van del 7 al 12
 - **SF7** es el más rápido
 - **SF12** es el más lento
- **BW (Bandwidth – Ancho de banda)**
 - LoRa utiliza 3 anchos de banda **125**, 250 y 500 kHz
- **DR (Data Rate – Velocidad de datos)**
 - Dependiendo de la región, se enumeran distintos DR definidos por el SF y BW que se utilicen
 - En Europa, DR0 – DR7
 - Velocidades de 300 bps a 50 kbps








https://docs.exploratory.engineering/lora/dr_sf/





¿Qué son las LPWAN? 🤖

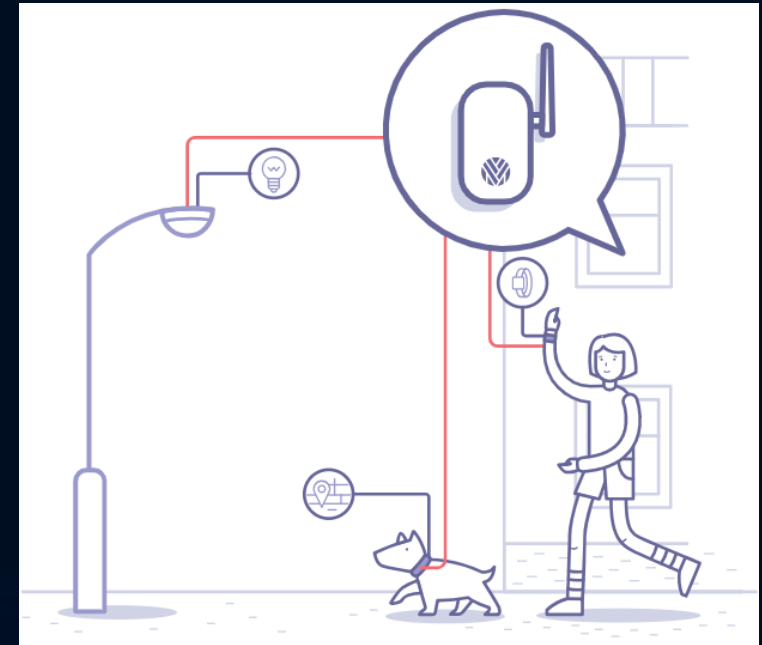


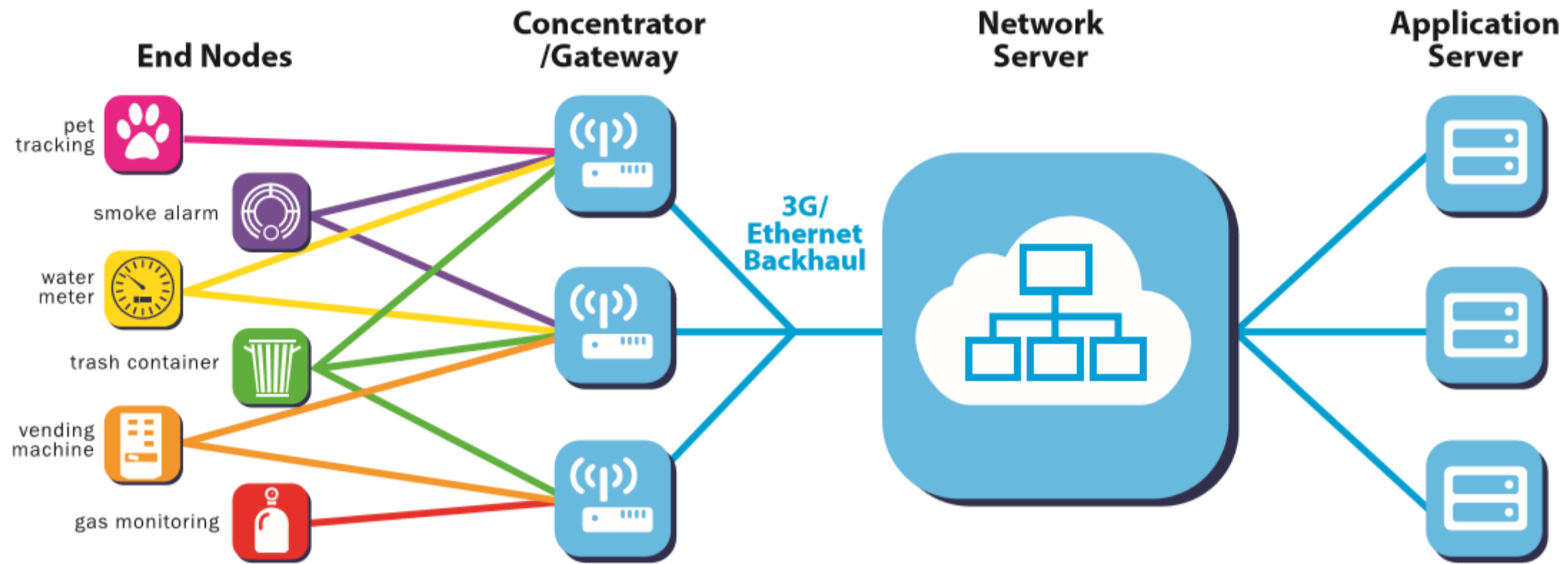
	Local Area Network Short Range Communication	Low Power Wide Area (LPWAN) Internet of Things	Cellular Network Traditional M2M
	40%	45%	15%
	Well established standards In building	Low power consumption Low cost Positioning	Existing coverage High data rate
	Battery Live Provisioning Network cost & dependencies	High data rate Emerging standards	Autonomy Total cost of ownership
	Bluetooth 4.0  		 3G+ / H+ 4G

<https://lora-alliance.org/resource-hub/what-lorawantm>

LPWAN (*Low Power Wide Area Network*)

- Redes inalámbricas de área **amplia y baja potencia**
- Estructura básica para la implementación del **Internet de las Cosas (IOT)**
- **Aumento** constante de la cantidad de 'dispositivos conectados'
- Factores críticos:
 - **Arquitectura** de red
 - **Rango** de enlace
 - Vida de las **baterías** o **baja potencia**
 - **Robustez** ante Interferencias/ruido
 - **Capacidad** de la red (nº máximo de nodos en la red)
 - **Seguridad**
 - Variedad de las **aplicaciones y servicios**







	SIGFOX	LoRa	clean slate cloT	NB LTE-M Rel. 13	LTE-M Rel. 12/13	EC-GSM Rel. 13	5G (targets)
Range (outdoor) MCL	<13km 160 dB	<11km 157 dB	<15km 164 dB	<15km 164 dB	<11km 156 dB	<15km 164 dB	<15km 164 dB
Spectrum Bandwidth	Unlicensed 900MHz 100Hz	Unlicensed 900MHz <50kHz	Licensed 7-900MHz 200kHz or dedicated	Licensed 7-900MHz 200kHz or shared	Licensed 7-900MHz 1.4 MHz or shared	Licensed 8-900MHz 2.4 MHz or shared	Licensed 7-900MHz shared
Data rate	<100bps	<10 kbps	<50kbps	<150kbps	<1 Mbps	10kbps	<1 Mbps
Battery life	>10 years	>10 years	>10 years	>10 years	>10 years	>10 years	>10 years
Availability	Today	Today	2016	2016	2016	2016	beyond 2020



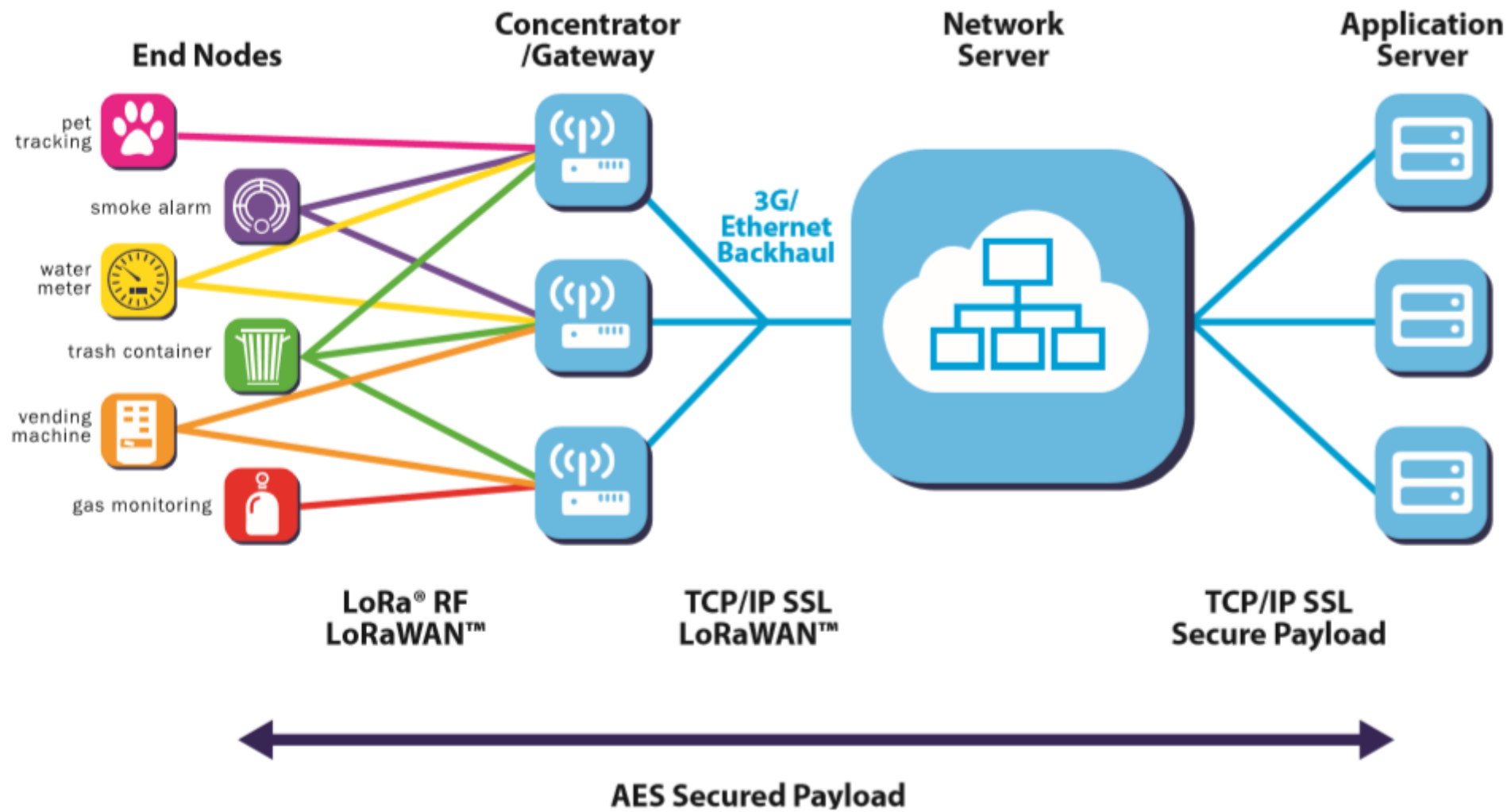
¿Qué es LoRaWAN™? 🗨️

LoRa® Wide Area Network (LoRaWAN)

- Especificación de código abierto que define un **protocolo** de comunicación y **arquitectura** para redes **LPWAN**
- Construida en base a la tecnología **LoRa®**
- **Desarrollada** por la [LoRa Alliance™](#)
 - Permite crear a empresas redes IOT basadas en sus **especificaciones** tecnológicas
 - > 500 empresas miembro
 - Creada para convertirse en un **estándar**
- Topología en **estrella** ☆
- Arquitectura **definida** por:
 - Nodos
 - Puertas de enlace (*Gateways*)
 - Servidor de red
 - Servidor de aplicación
- **Seguridad:** Red y Aplicación (*AES₁₂₈*) 🔒



Application				
LoRa® MAC				
MAC options				
Class A (Baseline)	Class B (Baseline)	Class C (Continuous)		
LoRa® Modulation				
Regional ISM band				
EU 868	EU 433	US 915	AS 430	—



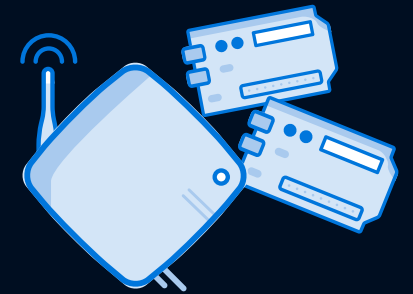
LoRa® Wide Area Network (LoRaWAN)

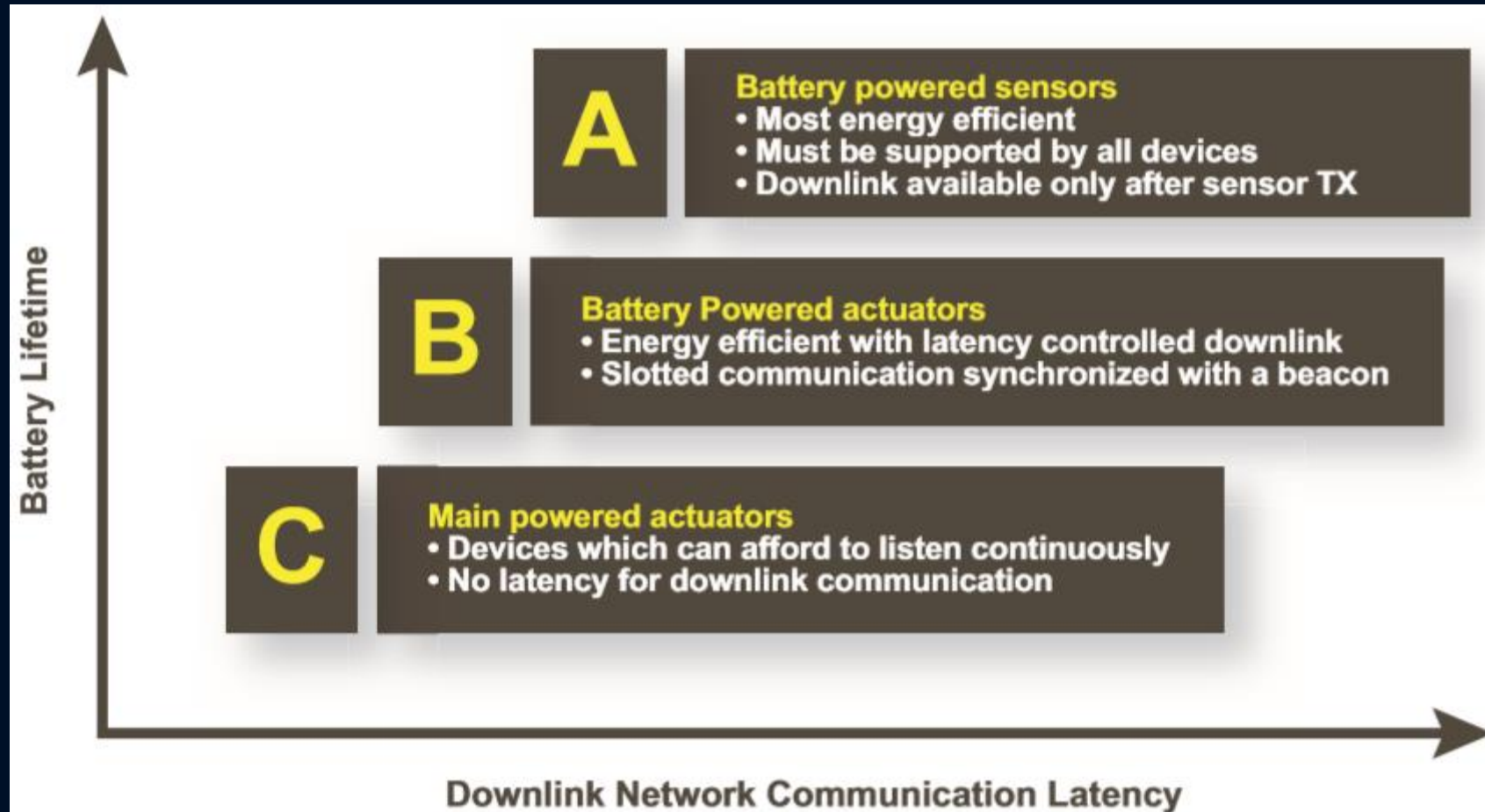
- Aspectos importantes:
 - **Arquitectura de red:**
 - Los nodos **NO** están **asociados** a un **gateway** específico
 - Los **datos** emitidos por un nodo son recibidos por **varios GWs**
 - Cada GW **reenviará** los datos recibidos al **Servidor de Red Cloud** vía Internet.
 - La **inteligencia** y **complejidad** de tratar los datos es del **Servidor de Red**, quien:
 - Gestionará la red
 - Aceptará mensajes del GW óptimo
 - Filtrará mensajes redundantes
 - Realizará comprobaciones de seguridad
 - Llevará a cabo la adaptación de la velocidad de los datos






LoRa® Wide Area Network (LoRaWAN)

- Aspectos importantes:
 - **Vida de las baterías:**
 - Nodos asíncronos. Envío de datos basado en eventos o planificados.
 - **Capacidad de la red:**
 - Gateways capaces de recibir **gran volumen** de datos
 - Gateways **Multicanal** → Permiten recibir mensajes simultáneos en múltiples canales
 - Escalabilidad
 - Enlaces simétricos de Uplink/Downlink
 - **Clases de dispositivos:**
 - Clase A → Baterías. Mayor ahorro de energía. Ventana RX tras envío
 - Clase B → Baterías o fuentes externas. Ventana RX predeterminadas con GW
 - Clase C → Ventana RX permanente. Fuentes externas



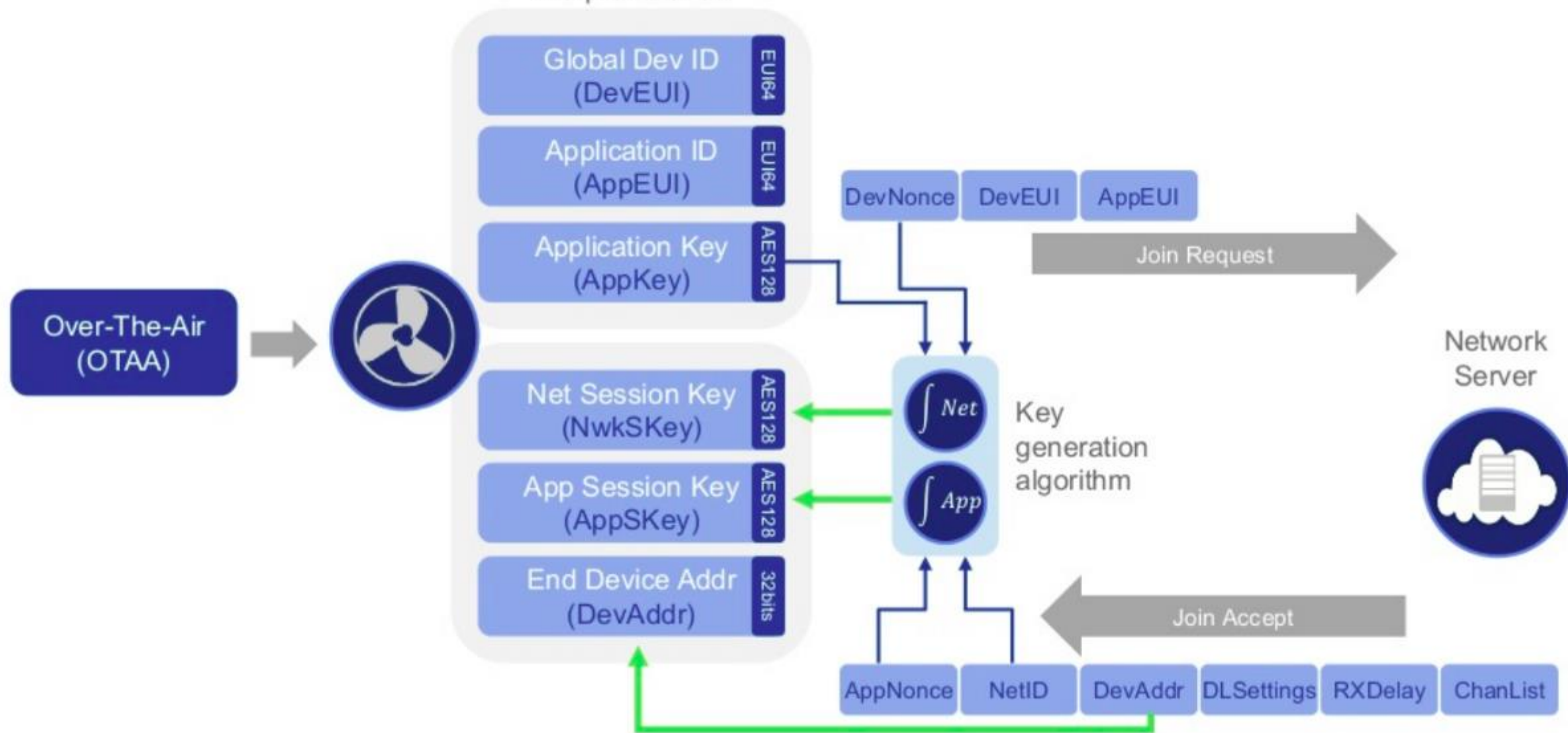


LoRa® Wide Area Network

- Aspectos importantes:
 - **Seguridad (encriptación AES128):**
 - Red → Asegura la autenticidad del nodo en la red
 - Aplicación → Asegura al operador de red que no tiene acceso a los datos finales del usuario
- **Conexión OTAA y ABP ( Información)**
 - **OTAA** → *Over-The-Air-Activation*. Un nodo prueba a conectarse a empleando las claves **DevEUI**, **AppEUI** y **AppKey**. Si las claves son correctas, el GW contestará al nodo con un mensaje de aceptación (join) y desde ese momento el nodo puede recibir y enviar datos.
 - Método más seguro 
 - **ABP** → *Activation-By-Personalization*. Las claves de cifrado se configuran manualmente en el dispositivo permitiendo la comunicación directa entre GW sin pedir permiso.
 - Método más sencillo pero inseguro 

LoRaWAN Over-The-Air Activation (OTAA)

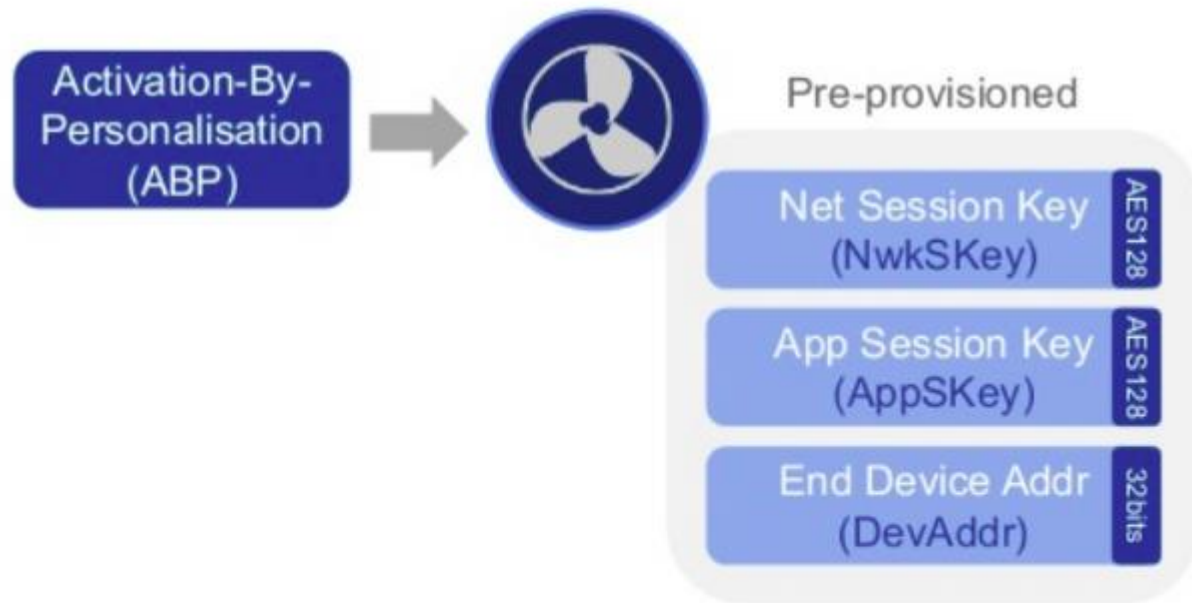
Pre-provisioned

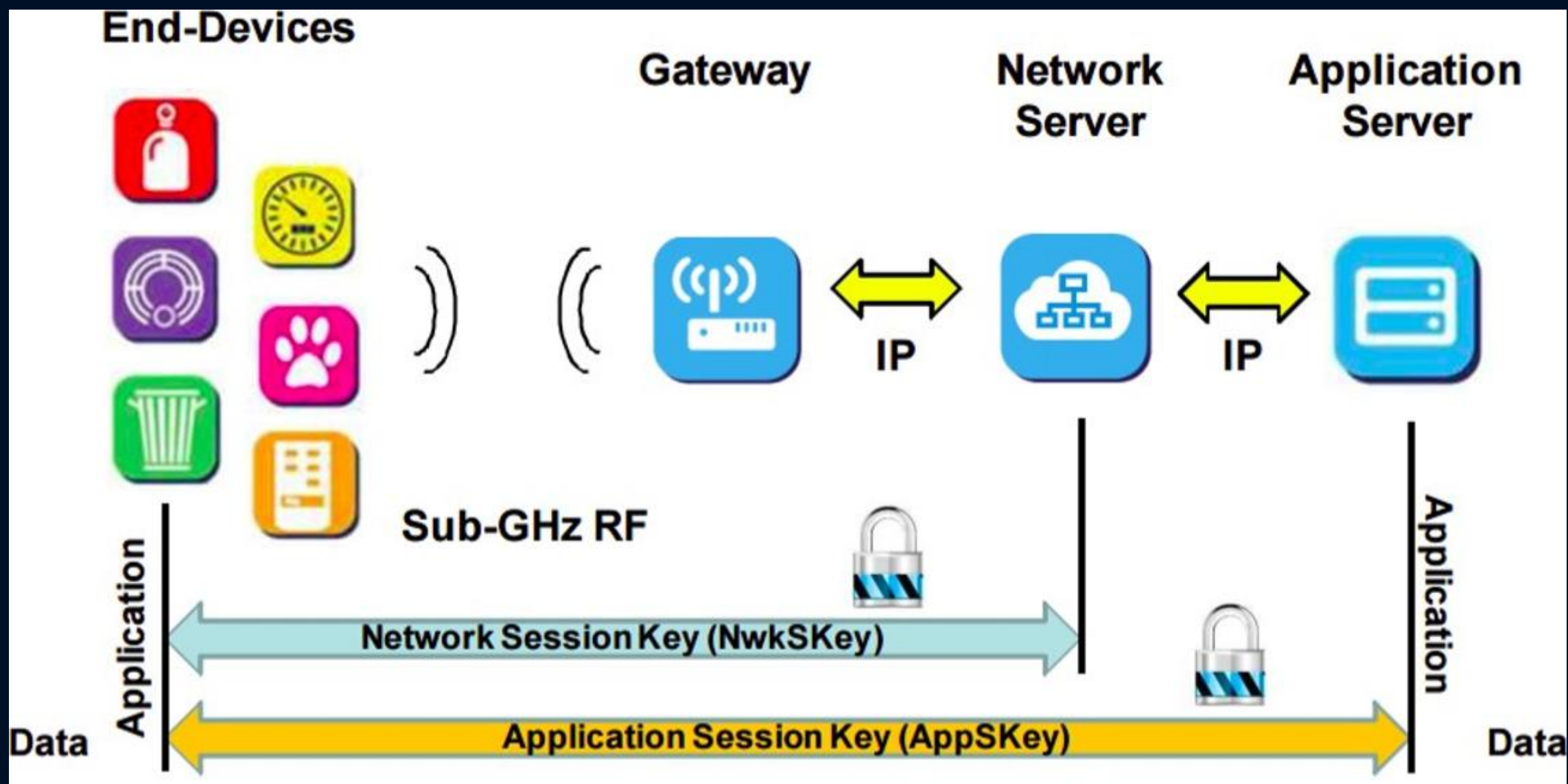


LoRaWAN Activation-By-Personalisation (ABP)

ABP pre-provisions keys and device address

Join procedure is bypassed







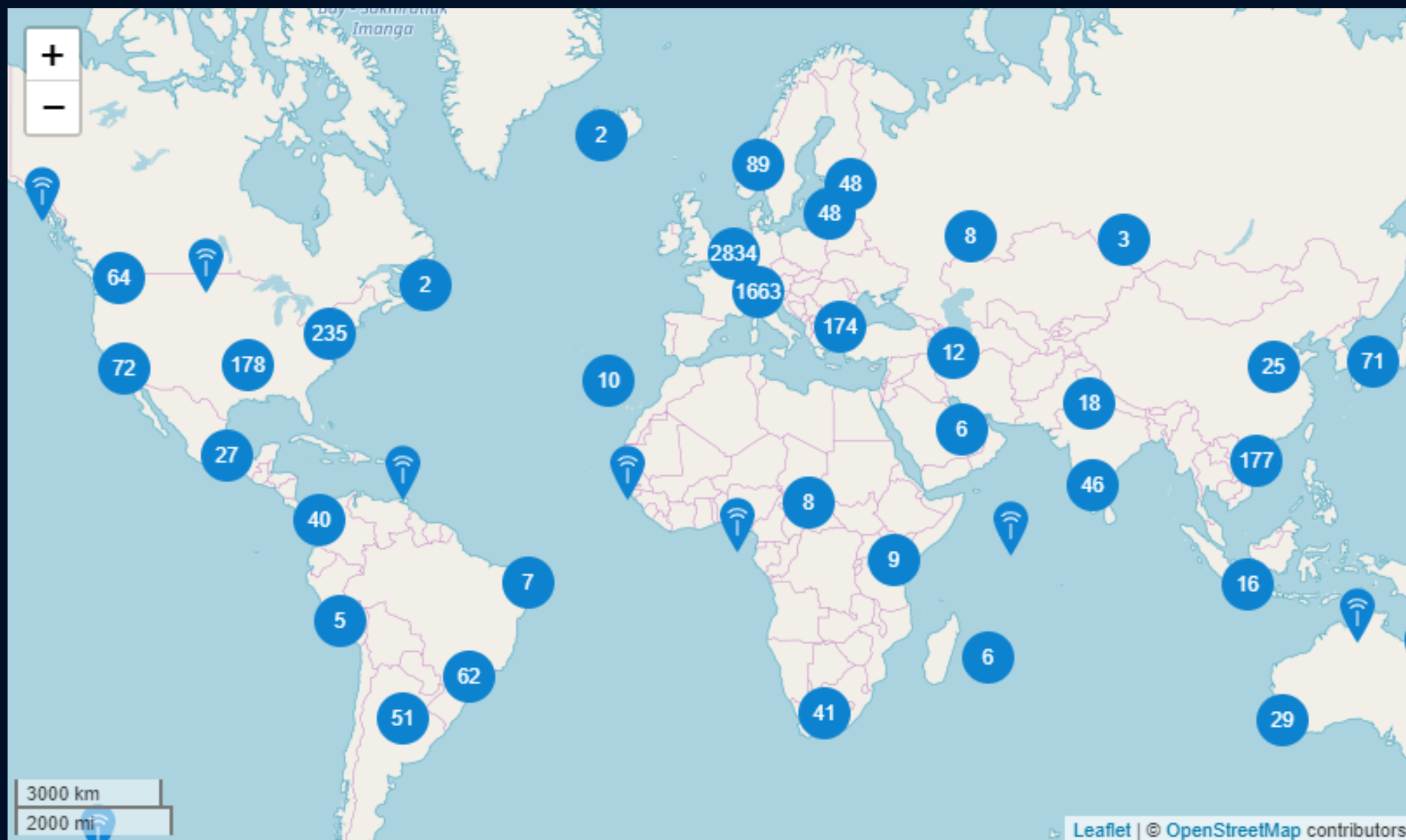
THE THINGS NETWORK

¿Qué es The Things Network? 🗨️

The Things Network

- Iniciativa (2015, Holanda) que ha creado una red **descentralizada y distribuida** para el IOT basada en la especificación **LoRaWAN**
- Permite que 'las cosas' hablen a Internet sin WiFi o 3G
- Filosofía **Open Source Hardware y Software** (Plataforma 'Maker')
- La red es **creada y mantenida** por la **comunidad** aportando nuevas puertas de enlace
- Actualmente:
 - > 6700 GWs distribuidos por el mundo 🌐
 - > 90 países 🌍
 - > 66.000 usuarios 👤
- Manifiesto:
 - Tus datos son tus datos (cifrado punto a punto) 🔒
 - Neutralidad de la red (todos los datos se tratan igual) ☮️
 - Open Source (tecnología desarrollada se hace abierta) ♻️







¿Por qué utilizar The Things Network?

- Si existe **cobertura** en tu zona...
 - ¡Sólo necesitarás un **NODO** para empezar a cacharrear! 😊🔧
 - **Ahorro** en costes (no necesitas instalar un Gateway) ↓💰
- Posibilidad de acceder a tus **datos** desde Internet ↑🌐
- **Seguridad** y **cifrado** de las comunicaciones 🔒🔑
- **Escalabilidad** sencilla 📈
- Basado en el estándar **LoRaWAN™** 📡📊
- Gran comunidad **Maker** 👤🔧



TALLER MAKER



LO VAMOS A PASAR MUY BIEN Y LO SABES 😊 🤖



Montaje y configuración Gateway Monocanal




Gateway Monocanal

- Configuración IDE Arduino:
 - [Descarga IDE](#)
 - Añadir soporte ESP8266 (ESP32 si se usa ESP32 TTGO) . **Importante usar V 2.4.2**
 - Archivo -> Preferencias -> Gestor de URLs Adicionales de Tarjetas
http://arduino.esp8266.com/stable/package_esp8266com_index.json
https://dl.espressif.com/dl/package_esp32_index.json

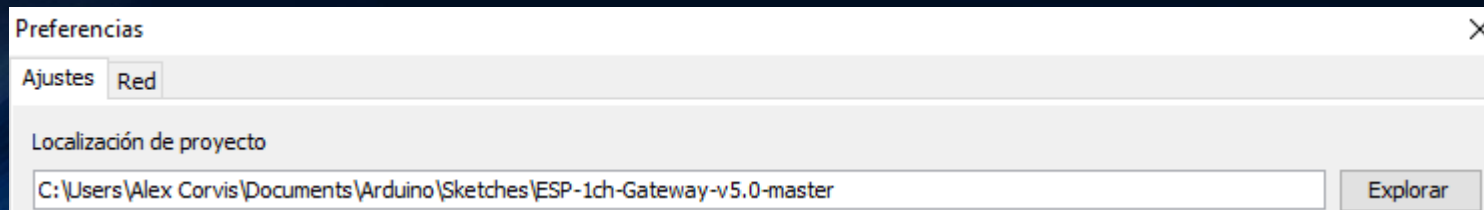
esp8266 by ESP8266 Community versión 2.4.2 **INSTALLED**

Tarjetas incluidas en éste paquete

Generic ESP8266 Module, Generic ESP8285 Module, Phoenix (ESP-13 Module), Adafruit Feather HUZZAH ESP8266, Invent One, XinaBox CW01, ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 2.0, NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Olimex MOD-WIFI-ESP8266(-DEV), SparkFun ESP8266 Thing, SparkFun ESP8266 Thing Dev, SweetPea ESP-210, LOLIN(WEMOS) D1 R2 & mini, LOLIN(WEMOS) D1 mini Pro, LOLIN(WEMOS) D1 mini Lite, WeMos D1 R1, ESPino (ESP-12 Module), ThaiEasyElec's ESPino, WifInfo, Arduino, 4D Systems gen4 IoD Range, Digistump Oak, WiFiduino, Amperka WiFi Slot, Sseed Wio Link, ESPectro Core.



- Descargar código [Repositorio GW Monocanal](#)
- Cambiar la localización del proyecto a la carpeta donde se encuentre el código
 - Archivo -> Preferencias -> Localización del Proyecto



Gateway Monocanal

- Abrir el proyecto (**ESP-sc-gway.ino**) y realizar las siguientes modificaciones de los archivos
 - **ESP-sc-gway.h**

```
// This value of DEBUG determines whether some parts of code get compiled.
// Also this is the initial value of debug parameter.
// The value can be changed using the admin webserver
// For operational use, set initial DEBUG value 0
#define DEBUG 1

// Debug message will be put on Serial if this one is set.
// If set to 0, not USB Serial prints are done
// Set to 1 it will print all user level messages (with correct debug set)
// If set to 2 it will also print interrupt messages (not recommended)
#define DUSB 1

// Define whether we should do a formatting of SPIFFS when starting the gateway
// This is usually a good idea if the webserver is interrupted halfway a writing
// operation.
// Normally, value 0 is a good default.
#define _SPIFF_FORMAT 0

// Define the LoRa Frequency band that is used. TTN Supported values are 925MHz, 868MHz and 433MHz.
// So supported values are: 433 868 915
#define _LFREQ 868
```


Gateway Monocanal

- Abrir el proyecto (ESP-sc-gway.ino) y realizar las siguientes modificaciones de los archivos
 - ESP-sc-gway.h

```
// The spreading factor is the most important parameter to set for a single channel
// gateway. It specifies the speed/datarate in which the gateway and node communicate.
// As the name says, in principle the single channel gateway listens to one channel/frequency
// and to one spreading factor only.
// This parameters contains the default value of SF, the actual version can be set with
// the webserver and it will be stored in SPIFF
// NOTE: The frequency is set in the loraModem.h file and is default 868.100000 MHz.
#define _SPREADING SF7

// Channel Activity Detection
// This function will scan for valid LoRa headers and determine the Spreading
// factor accordingly. If set to 1 we will use this function which means the
// 1-channel gateway will become even more versatile. If set to 0 we will use the
// continuous listen mode.
// Using this function means that we HAVE to use more dio pins on the RFM95/sx1276
// device and also connect enable diol to detect this state.
#define _CAD 0
```


Gateway Monocanal

- Abrir el proyecto (**ESP-sc-gway.ino**) y realizar las siguientes modificaciones de los archivos
 - **ESP-sc-gway.h**

```
// Definitions for over the air updates. At the moment we support OTA with IDE
// Make sure that you have installed Python version 2.7 and have Bonjour in your network.
// Bonjour is included in iTunes (which is free) and OTA is recommended to install
// the firmware on your router without having to be really close to the gateway and
// connect with USB.
#define A_OTA 0 // NOT SUPPORTED YET FOR ESP32

// We support a few pin-out configurations out-of-the-box: HALLARD, COMPRESULT and TTGO ESP32.
// If you use one of these two, just set the parameter to the right value.
// If your pin definitions are different, update the loraModem.h file to reflect these settings.
// 1: Bricolabs WemosD1 ESP8266 Shield
// 2: COMPRESULT pin out
// 3: ESP32 Wemos pin out
// 4: ESP32 TTGO pinning (should work for 433 and OLED too).
// 5: ESP32 TTGO EU433 MHz with OLED
// 6: Other, define your own in loraModem.h
#define _PIN_OUT 1

// Gather statistics on sensor and Wifi status
// 0= No statistics
// 1= Keep track of messages statistics, number determined by MAX_STAT
// 2= Option 1 + Keep track of messages received PER each SF (default)
// 3= See Option 2, but with extra channel info (Do not use when no Hopping is done)
#define STATISTICS 3
```

Gateway Monocanal

- Abrir el proyecto (**ESP-sc-gway.ino**) y realizar las siguientes modificaciones de los archivos
 - **ESP-sc-gway.h**

```
// Single channel gateways if they behave strict should only use one frequency
// channel and one spreading factor. However, the TTN backend replies on RX2
// timeslot for spreading factors SF9-SF12.
// Also, the server will respond with SF12 in the RX2 timeslot.
// If the lch gateway is working in and for nodes that ONLY transmit and receive on the set
// and agreed frequency and spreading factor. make sure to set STRICT to 1.
// In this case, the frequency and spreading factor for downlink messages is adapted by this
// gateway
// NOTE: If your node has only one frequency enabled and one SF, you must set this to 1
//      in order to receive downlink messages
// NOTE: In all other cases, value 0 works for most gateways with CAD enabled
#define _STRICT_LCH 1

// Allows configuration through WifiManager AP setup. Must be 0 or 1
#define WIFIMANAGER 0 // NOT SUPPORTED YET FOR ESP32
```

Gateway Monocanal

- Abrir el proyecto (ESP-sc-gway.ino) y realizar las siguientes modificaciones de los archivos
 - ESP-sc-gway.h

```
// Define if OLED Display is connected to I2C bus. Note that defining an OLED display does not
// impact performance very much, certainly if no OLED is connected. Wrong OLED will not show
// sensible results on display
// OLED==0; No OLED display connected
// OLED==1; 0.9 Oled Screen based on SSD1306
// OLED==2; 1"3 Oled screens for Wemos, 128x64 SH1106
#define OLED 0

// MQTT definitions, these settings should be standard for TTN
// and need not changing
#define _TTNPORT 1700           // Standard port for TTN
#define _TTNSERVER "router.eu.thethings.network"

// Gateway Ident definitions
#define _DESCRIPTION "WemosD1 ESP8266 GW 868.1 SF7" // Name of the gateway
#define _EMAIL "your_email_adress@domain.com" // Owner
#define _PLATFORM "ESP8266"
#define _LAT 43.537962
#define _LON -5.637166
#define _ALT 0 // Altitude

// ntp
// Please add daylight saving time to NTP_TIMEZONES when desired
#define NTP_TIMESERVER "es.pool.ntp.org" // Country and region specific
#define NTP_TIMEZONES 1 // How far is our Timezone from UTC (excl daylight saving/summer time)
#define SECS_IN_HOUR 3600
#define NTP_INTR 0 // Do NTP processing with interrupts or in loop();
```

Gateway Monocanal

- Abrir el proyecto (**ESP-sc-gway.ino**) y realizar las siguientes modificaciones de los archivos
 - **ESP-sc-gway.h**
 - Añadir los SSID y password de las Wi-Fi a las que el Gateway se conectará.
 - Importante dejar la **primera línea sin modificar**

```
// Define the correct radio type that you are using
#define CFG_sx1276_radio
//#define CFG_sx1272_radio

// Please fill in at least ONE SSID and password from your own WiFi network
// below. This is needed to get the gateway working
// Note: DO NOT use the first and the last line of the structure, these should be empty strings and
// the first line in the structure is reserved for WifiManager.
//
wpas wpa[] = {
  { "", "" }, // Reserved for WiFi Manager
  { "SpaceX", "XXXXXXXXXX" },
  { "MOVISTAR_B6A4", "XXXXXXXXXX" },
};
```


Gateway Monocanal

- Abrir el proyecto (**ESP-sc-gway.ino**) y realizar las siguientes modificaciones de los archivos
 - **loraModem.h**

 [Layout PCB LoRa Bricolabs](#)

 [Layout TTGOs](#)

- **ESP8266**

```
#if _PIN_OUT==1
// -----
// Definition of the GPIO pins used by the Gateway for Bricolabs ESP8266 LoRa Shield
//
struct pins {
    uint8_t dio0=0;    //D3 GPIO0  Disconnected from the shield :( seems to work ok without it :)
    uint8_t diol=2;    //D4 GPIO2  Disconnected from the shield Used for CAD, may or not be shared with DI00
    uint8_t dio2=16;   //D0 GPIO16 Disconnected from the shield (jumper cut) Used for frequency hopping, don't care
    uint8_t ss=15;     //D8 GPIO15 Select pin
    uint8_t rst=16;    //D0 GPIO16 Reset pin not used
    // MISO 12 / D6
    // MOSI 13 / D7
    // CLK  14 / D5
} pins;
```

Gateway Monocanal

- Abrir el proyecto (ESP-sc-gway.ino) y realizar las siguientes modificaciones de los archivos
 - loraModem.h

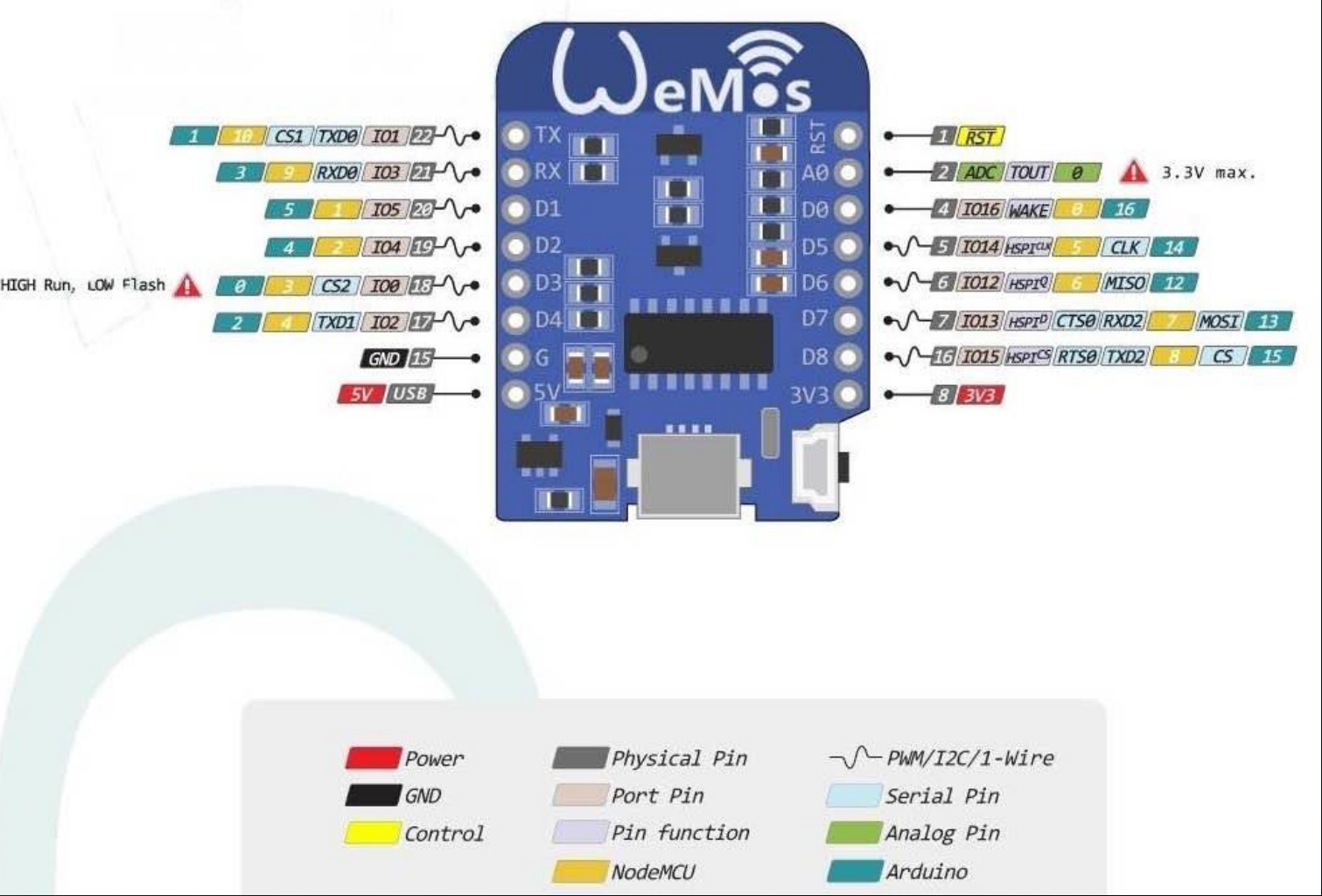
 [Layout PCB LoRa Bricolabs](#)

 [Layout TTGOs](#)

- ESP32 TTGo V1

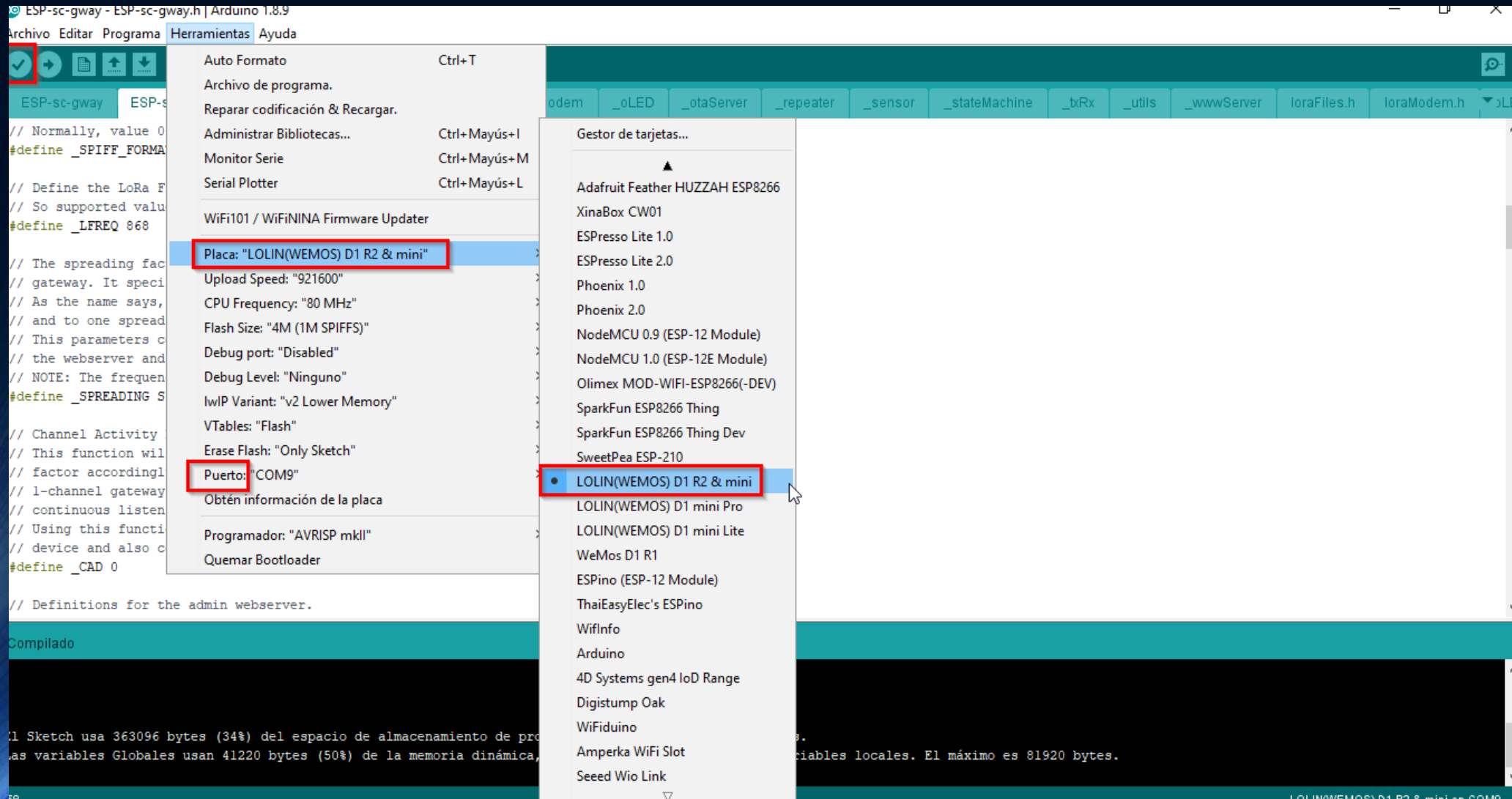
```
#elif _PIN_OUT==4
// -----
// For ESP32/TTGO based board.
// SCK == GPIO5/ PIN5
// SS == GPIO18/PIN18 CS
// MISO == GPIO19/ PIN19
// MOSI == GPIO27/ PIN27
// RST == GPIO14/ PIN14
struct pins {
    uint8_t dio0=26; // GPIO26 / Dio0 used for one frequency and one SF
    uint8_t dio1=33; // GPIO26 / Used for CAD, may or not be shared with DIO0
    uint8_t dio2=32; // GPIO26 / Used for frequency hopping, don't care
    uint8_t ss=18; // GPIO18 / Dx. Select pin connected to GPIO18
    uint8_t rst=14; // GPIO0 / D3. Reset pin not used
} pins;
#define SCK 5
#define MISO 19
#define MOSI 27
#define RST 14
#define SS 18
#define GPS_RX 15
#define GPS_TX 12
```

-



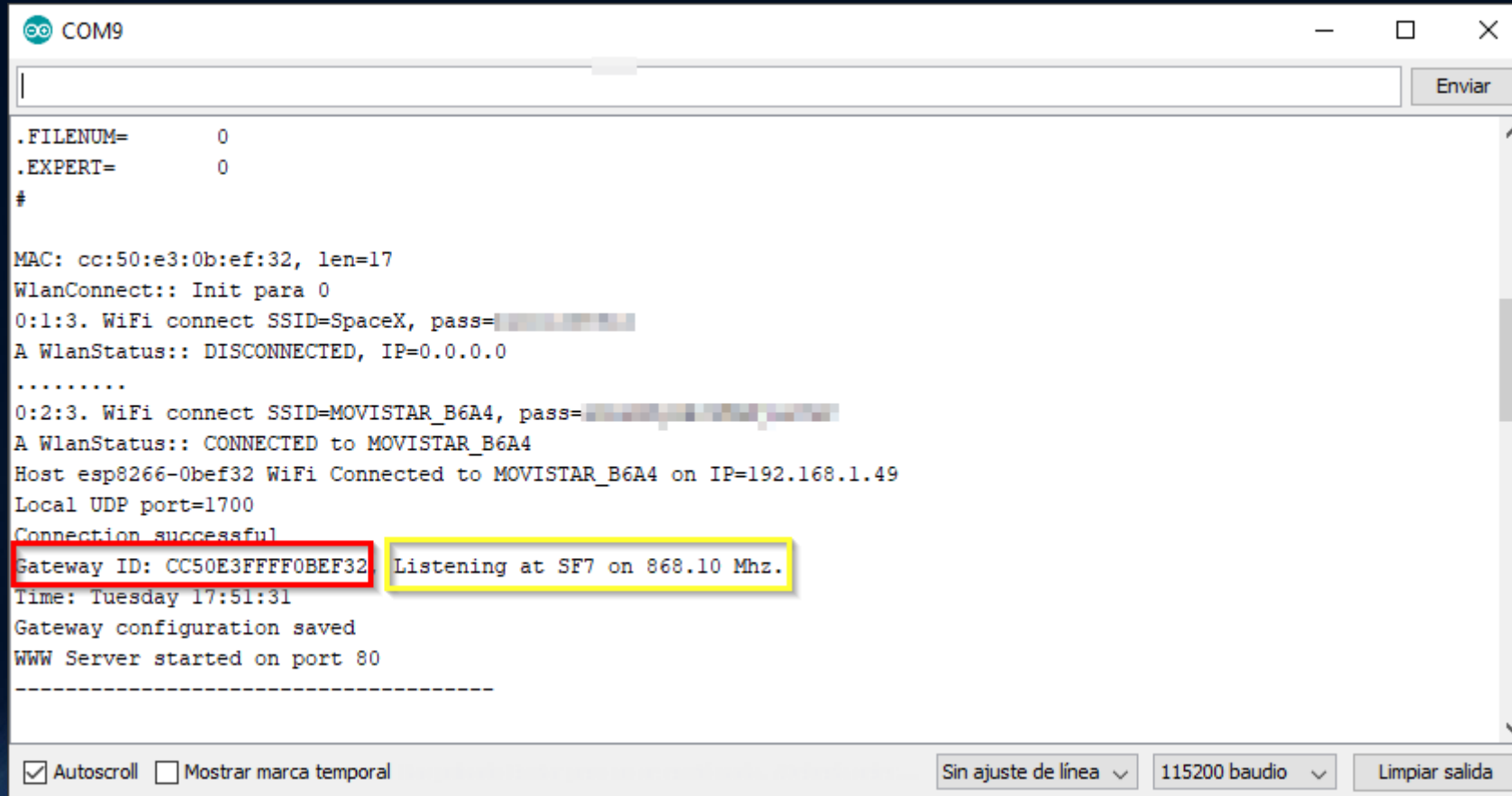
Gateway Monocanal

- ¿Compilación sin errores 🤔 del código del proyecto?



Gateway Monocanal

- Subir programa y obtener **Gateway ID** por monitor serie



```
COM9
. FILENUM=      0
. EXPERT=       0
#
MAC: cc:50:e3:0b:ef:32, len=17
WlanConnect:: Init para 0
0:1:3. WiFi connect SSID=SpaceX, pass=
A WlanStatus:: DISCONNECTED, IP=0.0.0.0
.....
0:2:3. WiFi connect SSID=MOVISTAR_B6A4, pass=
A WlanStatus:: CONNECTED to MOVISTAR_B6A4
Host esp8266-0bef32 WiFi Connected to MOVISTAR_B6A4 on IP=192.168.1.49
Local UDP port=1700
Connection successful
Gateway ID: CC50E3FFFF0BEF32 Listening at SF7 on 868.10 Mhz.
Time: Tuesday 17:51:31
Gateway configuration saved
WWW Server started on port 80
-----
☒ Autoscroll ☐ Mostrar marca temporal Sin ajuste de línea 115200 baudio Limpiar salida
```

Gateway Monocanal

- Abrir dirección IP asignada en el navegador  para acceder a la configuración  del GW

ESP Gateway Config

Version: V5.3.3-B 180825a
ESP alive since: Tuesday 26-3-2019 17:59:47, Uptime: 0:00:06.03
Current time: Tuesday 26-3-2019 18:05:02

[Documentation](#) [Expert Mode](#) [Log Files](#)

Package Statistics

Counter	C 0	C 1	C 2	Pkgs	Pkgs/hr
Packages Downlink				0	
Packages Uplink Total				0	0
Packages Uplink OK				0	
SF7 revd	0	0	0	0	0 %
SF8 revd	0	0	0	0	0 %
SF9 revd	0	0	0	0	0 %
SF10 revd	0	0	0	0	0 %
SF11 revd	0	0	0	0	0 %
SF12 revd	0	0	0	0	0 %

Message History

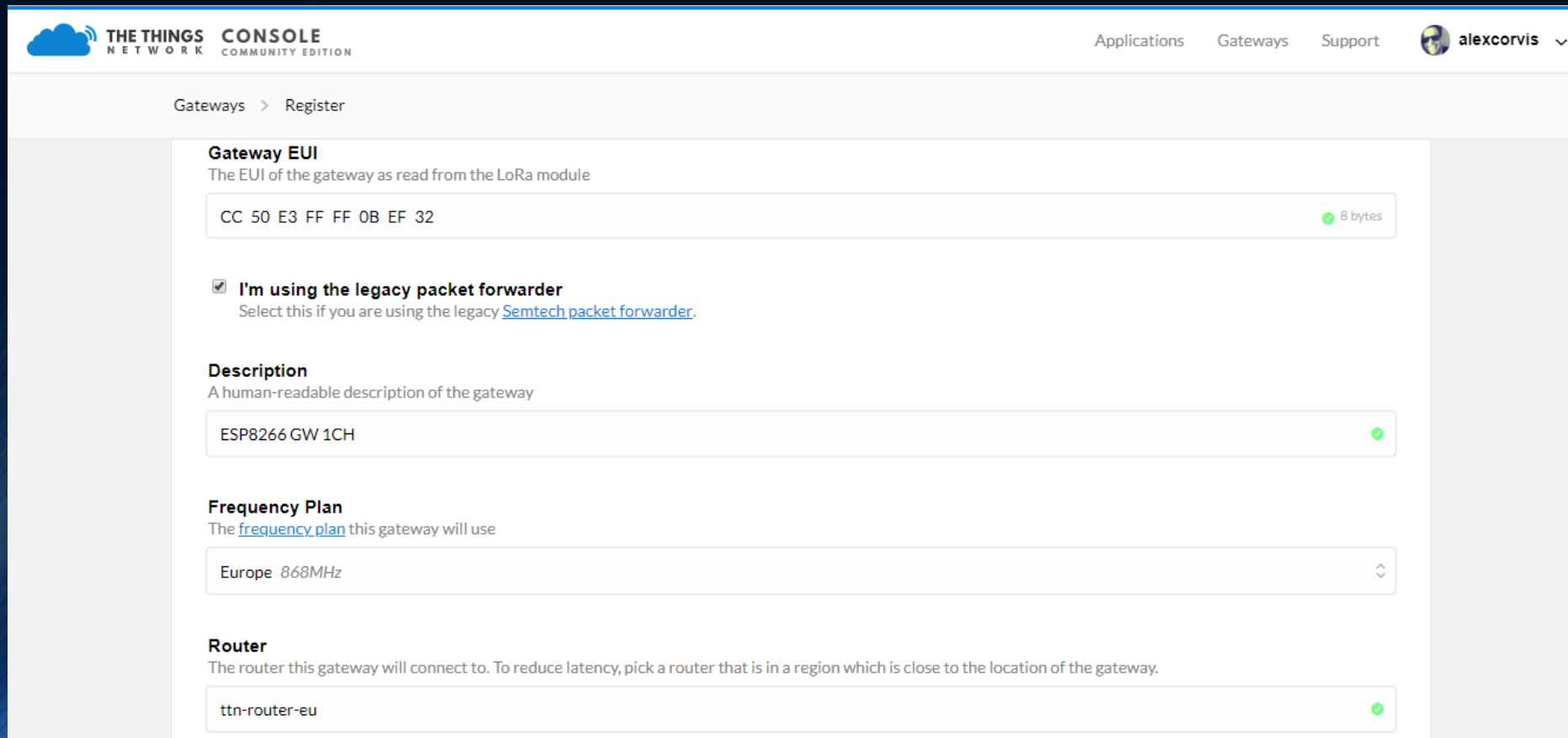
Time	Node	C	Freq	SF	pRSSI
------	------	---	------	----	-------

Gateway Settings

Setting	Value	Set
CAD	OFF	ON OFF
HOP	OFF	ON OFF
SF Setting	7	- +
Channel	0	- +
Debug level	1	- +
Debug pattern	SCN CAD RX TX	PRE MAI GUI RDIO
Usb Debug	1	
WWW Refresh	OFF	ON OFF
Update Firmware		UPDATE
Format SPIFFS		FORMAT
Statistics	0	RESET
Boots and Resets	3	RESET

Gateway Monocanal

- Registrar Gateway en The Things Network
 - [Crear cuenta](#)
 - Registrar GW -> Gateway EUI que hemos obtenido anteriormente

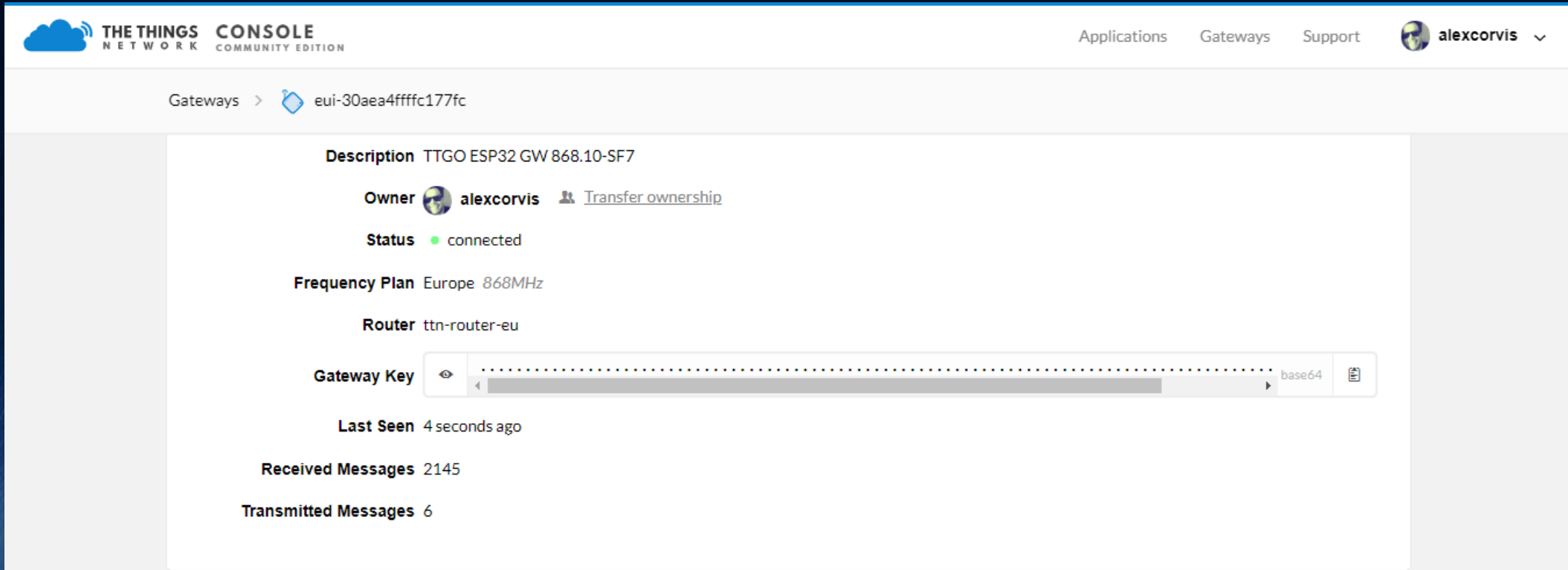


The screenshot shows the 'Register' page for a gateway in the The Things Network Console. The page is titled 'Gateways > Register'. It contains several form fields for gateway registration:

- Gateway EUI**: The EUI of the gateway as read from the LoRa module. The input field contains 'CC 50 E3 FF FF 0B EF 32' and is marked as valid (8 bytes).
- I'm using the legacy packet forwarder**: A checkbox that is checked. Below it, a note says 'Select this if you are using the legacy [Semtech packet forwarder](#).'
- Description**: A human-readable description of the gateway. The input field contains 'ESP8266 GW 1CH' and is marked as valid.
- Frequency Plan**: The [frequency plan](#) this gateway will use. The dropdown menu is set to 'Europe 868MHz'.
- Router**: The router this gateway will connect to. To reduce latency, pick a router that is in a region which is close to the location of the gateway. The dropdown menu is set to 'ttn-router-eu' and is marked as valid.

Gateway Monocanal

- Registrar Gateway en The Things Network



The screenshot displays the 'The Things Network Console' interface. The top navigation bar includes the logo, 'CONSOLE COMMUNITY EDITION', and links for 'Applications', 'Gateways', and 'Support'. A user profile for 'alexcorvis' is visible in the top right. The main content area shows the details for a specific gateway with EUI 'eui-30aea4fffc177fc'. The details include:

- Description:** TTGO ESP32 GW 868.10-SF7
- Owner:** alexcorvis, with a link to 'Transfer ownership'.
- Status:** connected (indicated by a green dot).
- Frequency Plan:** Europe 868MHz
- Router:** ttn-router-eu
- Gateway Key:** A field with a toggle for visibility (currently hidden), a base64 encoding option, and a copy icon.
- Last Seen:** 4 seconds ago
- Received Messages:** 2145
- Transmitted Messages:** 6

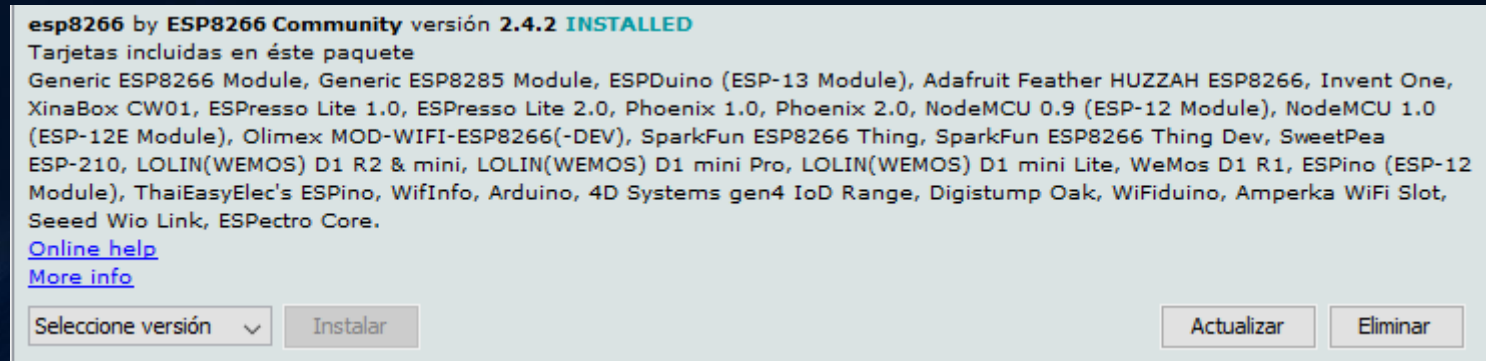


Montaje y configuración de Nodos



Nodos LoRa: Wemos D1+Bricolabs Shield

- Configuración IDE Arduino:
 - [Descarga IDE](#)
 - Añadir soporte ESP8266 (ESP32 si se usa ESP32 TTGO)
 - Archivo -> Preferencias -> Gestor de URLs Adicionales de Tarjetas
http://arduino.esp8266.com/stable/package_esp8266com_index.json (para ESP8266)
https://dl.espressif.com/dl/package_esp32_index.json (para ESP32)
 - Reiniciar IDE e Instalar soporte
 - Herramientas -> Placa -> Gestor de Tarjetas -> **esp8266 by ESP8266 Community**
⚠ Importante usar V 2.4.2 ⚠



Nodos LoRa: Wemos D1+Bricolabs Shield

- Añadir librería MCCI LoRaWAN LMIC library V2.3.2
<https://github.com/mcci-catena/arduino-lmic>
 - Programa -> Incluir Librería -> Administrar Bibliotecas

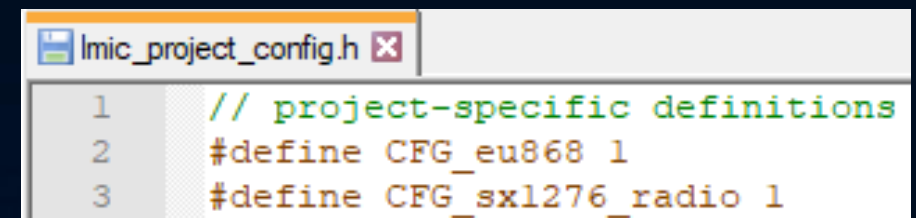
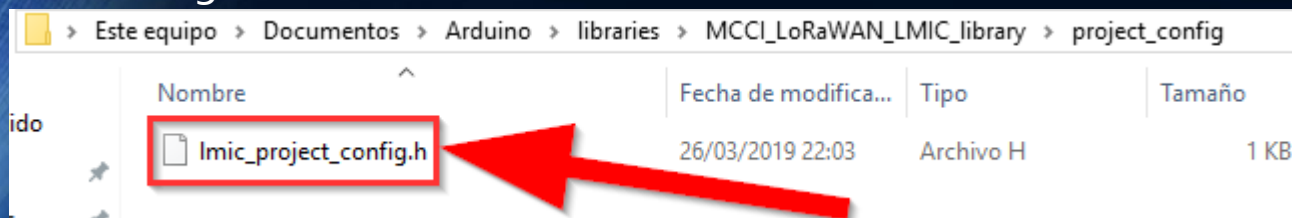
MCCI LoRaWAN LMIC library by IBM, Matthijs Kooijman, Terry Moore, ChaeHee Won, Frank Rose Versión 2.3.2 **INSTALLED**
Arduino port of the LMIC (LoRaWAN-MAC-in-C) framework provided by IBM. Supports SX1272/SX1276 and HopeRF RFM92/RFM95 transceivers. Refactored to support multiple bandplans beyond the original two supported by the IBM LMIC code. Various enhancements and bug fixes from MCCI and The Things Network New York. Original IBM URL <http://www.research.ibm.com/labs/zurich/ics/lrsc/lmic.html>.

[More info](#)

Seleccione versión ▾

Instalar

- Configuración librería MCCI-Catena/arduino-lmic





Nodos LoRa: Wemos D1+Bricolabs Shield

- Descargar repositorio

https://github.com/alexcorvis84/LoRa_MakersAsturias/tree/master/Nodo-TTN-ABP-ESP8266-BricolabsGen

- Abrir código con el IDE de Arduino

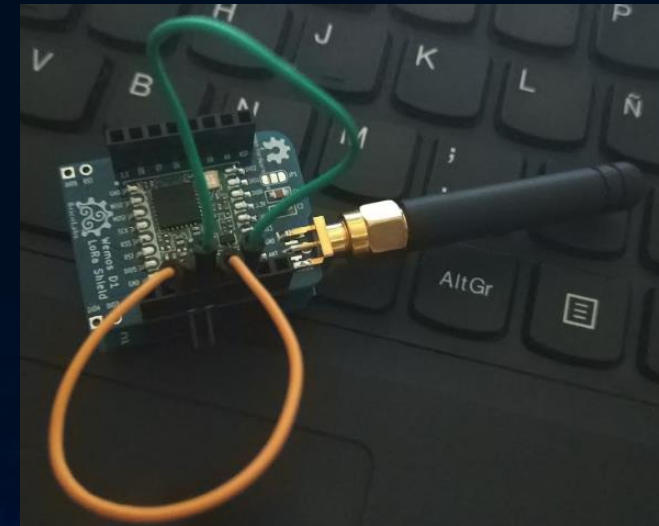
 `Nodo-TTN-ABP-ESP8266-BricolabsGen.ino`

- Rellenar los campos necesarios (**NWSKEY**, **APPSKEY**, **DEVADDR**) para **método ABP** obtenidos a través **The Things Network** al añadir un Dispositivo (**Device**) a una Aplicación (**Application**) previamente creados ( seguid los pasos del tutorial de @akirasan Nodo LoRaWAN con ESP32)

```
static const PROGMEM u1_t NWKEY[16] = { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 };
static const u1_t PROGMEM APPKEY[16] = { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 };
static const u4_t DEVADDR = 0x00000000 ; // Node address
```

- ¡Hacer la 'Maker Ñapa' con 2 cables 😊 para que funcione el nodo!

```
* -----
* -----
* IMPORTANT NOTE: Need to connect a pair of Wires from D3 to D1 & from D4 to D2
* This is needed due the pins used by the PCB (D3 & D4) for DIO0 and DIO1
* respectively, are used by the ESP8266 on the initialization and will make it
* reboot (PCB design error ^^)
* Changing them to use D1 & D2 instead, makes it to work OK! :-)
```




Applications > test_lab_lora_nodes > Devices > esp8266_lora_rfm95

Application ID test_lab_lora_nodes

Device ID esp8266_lora_rfm95



Description ESP8266 LoRa Node RF95



Activation Method ABP

Device EUI <> ⇅ 00 74 E2 88 47 0D 24 65 

Application EUI <> ⇅ 70 B3 D5 7E D0 01 94 0A 

Device Address <> ⇅ 26 01 18 DD 

Network Session Key <> ⇅  

App Session Key <> ⇅  

Applications > test_lab_lora_nodes > Devices > esp8266_lora_rfm95 > Data

Overview

Data

Settings

APPLICATION DATA

|| pause  clear

Filters

uplink

downlink

activation

ack

error

	time	counter	port	
▲	21:24:53	4	1	payload: 4D 61 6B 65 72 73 41 73 74 75 72 69 61 73 21 message: "MakersAsturias!"
▲	21:24:31	3	1	payload: 4D 61 6B 65 72 73 41 73 74 75 72 69 61 73 21 message: "MakersAsturias!"
▲	21:24:10	2	1	payload: 4D 61 6B 65 72 73 41 73 74 75 72 69 61 73 21 message: "MakersAsturias!"



Ejemplo Nodo ESP32 TTGO V1+DHT22



Nodo ESP32 TTGO V1+DHT22

- ⚠ Necesaria la librería **Cayenne LPP** (incluida en la librería *The Things Network*)
- Conectar un **sensor DHT22** (*Vcc*, *Gnd* y salida datos al **Pin Digital 23** de la placa *TTGO V1*)
- Descargar código del [Repositorio Nodo ESP32 TTGO V1+DHT22](#)
- Modificad las claves para **método ABP**

```
// LoRaWAN NwksKey, network session key
static const PROGMEM ul_t NWKSKY[16] = { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 };

// LoRaWAN AppSKey, application session key
static const ul_t PROGMEM APPSKY[16] = { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 };

// LoRaWAN end-device address (DevAddr)
// See http://thethingsnetwork.org/wiki/AddressSpace
// The library converts the address to network byte order as needed.
// #ifndef COMPILER_REGRESSION_TEST
static const u4_t DEVADDR = 0x00000000;
```

- Realizar [integración](#) con Cayenne My devices

[i CayenneLPP API Reference](#)

[i My Devices Cayenne](#)



ttn-abp-ttgoV1-eu868-dht22_lpp_gen

/*****

* The Things Network

* Example of using

* Single-channel TT

* <https://github.com>

* This uses ABP (Ac

* communication wou

* Learn Guide: <http>

* Copyright (c) 201

* Copyright (c) 201

* Copyright (c) 201

* Copyright (c) 201

* Permission is hereby granted, free of charge, to anyone

* obtaining a copy of this document and accompanying files,

* to do whatever they want with them without any restriction,

* including, but not limited to, copying, modification and redistri

* NO WARRANTY OF ANY KIND IS PROVIDED.

* Heltec Wifi LoRa 32, TTGO LoRa and TTGO LoRa32 V1:

ESP32 LoRa (SPI) Display (I2C) LED

GPIO5 SCK SCK

GPIO27 MOSI MOSI

GPIO19 MISO MISO

- Auto Formato Ctrl+T
- Archivo de programa.
- Reparar codificación & Recargar.
- Administrar Bibliotecas... Ctrl+Mayús+I
- Monitor Serie Ctrl+Mayús+M
- Serial Plotter Ctrl+Mayús+L
- WiFi101 / WiFiNINA Firmware Updater
- Placa: "TTGO LoRa32-OLED V1"
- Upload Speed: "921600"
- Flash Frequency: "80MHz"
- Core Debug Level: "Ninguno"
- Puerto
- Obtén información de la placa
- Programador: "AVRISP mkII"
- Quemar Bootloader

Gestor de tarjetas...

Linino One

Arduino Uno WiFi

Digistump AVR Boards

Digispark (Default - 16.5mhz)

Digispark Pro (Default 16 Mhz)

Digispark Pro (16 Mhz) (32 byte buffer)

Digispark Pro (16 Mhz) (64 byte buffer)

Digispark (16mhz - No USB)

Digispark (8mhz - No USB)

Digispark (1mhz - No USB)

ESP32 Arduino

ESP32 Dev Module

ESP32 Wrover Module

ESP32 Pico Kit

Turta IoT Node

• TTGO LoRa32-OLED V1

XinaBox CW02

SparkFun ESP32 Thing

u-blox NINA-W10 series (ESP32)

Widora AIR

Electronic SweetPeas - ESP320

Nano32

LOLIN D32

ttn-abp-ttgoV1-eu868-dht22_lpp

```
#include <lmic.h>
#include <hal/hal.h>
#include <SPI.h>
#include <U8x8lib.h>
#include <CayenneLPP.h>

// Include the DHT22 Sensor Library
#include "DHT.h"

// DHT digital pin and sensor type
#define DHTPIN 23
#define DHTTYPE DHT22
#define LEDPIN 2 // Programmable Blue LED BUILTIN

//OLED Pin I2C Definitions
#define SCL 15
#define SDA 4
#define RST 16

U8X8_SSD1306_128X64_NONAME_SW_I2C u8x8 (SCL, SDA, RST);

int channel = 0;

CayenneLPP lpp(51);
```

```
// Pin mapping for for TTGO LoRa32 V1
const lmic_pinmap lmic_pins = {
    .nss = 18,
    .rxtx = LMIC_UNUSED_PIN,
    .rst = 14,
    .dio = {26, 33, 32},
    // .rxtx_rx_active = 0,
    // .rssi_cal = 8, // LBT cal for the Adafruit Feather M0 LoRa, in dB
    // .spi_freq = 8000000,
};
```

ttn-abp-ttgoV1-eu868-dht22_lpp

```
void do_send(osjob_t* j){
    // Check if there is not a current TX/RX job running
    if (LMIC.opmode & OP_TXRXPEND) {
        Serial.println(F("OP_TXRXPEND, not sending"));
        u8x8.drawString(0, 7, "OP_TXRXPEND, not sent");

    } else {
        // read the temperature from the DHT22
        float temperature = dht.readTemperature();
        Serial.print("Temperature: "); Serial.print(temperature);
        Serial.println(" *C");
        // read the humidity from the DHT22
        float rHumidity = dht.readHumidity(); Serial.print("Humidity: ");
        Serial.print(rHumidity);
        Serial.println(" %RH ");

        lpp.reset();
        lpp.addTemperature(1, temperature);
        lpp.addRelativeHumidity(2, rHumidity);

        // prepare upstream data transmission at the next possible time.
        // transmit on port 1 (the first parameter); you can use any value from 1 to 223 (others are reserved).
        // don't request an ack (the last parameter, if not zero, requests an ack from the network).
        // Remember, acks consume a lot of network resources; don't ask for an ack unless you really need it.
        digitalWrite(LEDPIN, HIGH);
        delay(1000);
        LMIC_setTxData2(1, lpp.getBuffer(), lpp.getSize(), 0);
        digitalWrite(LEDPIN, LOW);
    }
}
```

Applications > alex_corvis84_ttn_app > Integrations

Overview

Devices

Payload Formats

Integrations

Data

Settings

INTEGRATIONS

[+ add integration](#)



MyDevices

test_lab


[+ Create new proj...](#)

 [Create App](#)


 [Submit Project](#)

 [Community](#)

 [Docs](#)

 [User Menu](#)

Add new... ▾

 Cayenne LPP ▾

 Overview

 Data

Cayenne LPP

Network: The Things Network



RSSI



-30.00

dBm

SNR



9.00

Decibels

Temperature (1)



24.00

Celsius

Humidity (2)



62.50

Percent (%)

GRACIAS A...

- Comunidades The Things Network ([Cataluña](#), Madrid, Sevilla, Málaga...)
- Asociación [BricoLabs](#)
- AlexTC ([@TCRobotics](#)) por las [LoRa PCBs](#)
- Juan Félix Mateos ([@juanfelixmateos](#)) de [TTN Madrid](#)
- La Hora Maker (https://youtu.be/EYcq3XEq_2c)
- Jorge ([@akirasan](#)) por sus fantásticos tutoriales
 - [Preparando Arduino IDE para ESP32+LoRa](#)
 - [MiniGateway LoRa monocanal con ESP32](#)
 - [Nodo LoRaWAN con ESP32](#)
 - [Nodo de mapeo TTNMapper: Arduino + GPS + LoRaWAN](#)
- [Grupo comunicaciones LoRa de larga Distancia](#) ([Germán Martín](#), [Galileo](#), [Xosé Pérez](#), [Gustavo](#),...)
- [CITECH](#)
- [Makers Asturias](#)