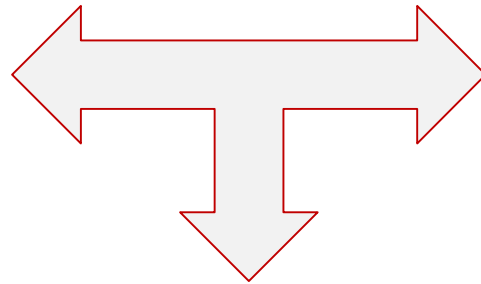


# Proof Of Concept - LORA-RFID-SAP for Waste & Recycling

---

Internet Of Things  
enabler



Environmental  
services

**Proof Of Concept:  
LoRa-RFID-SAP integration for  
Waste & Recycling**

Author: Marco Moschella  
Date: November 2018

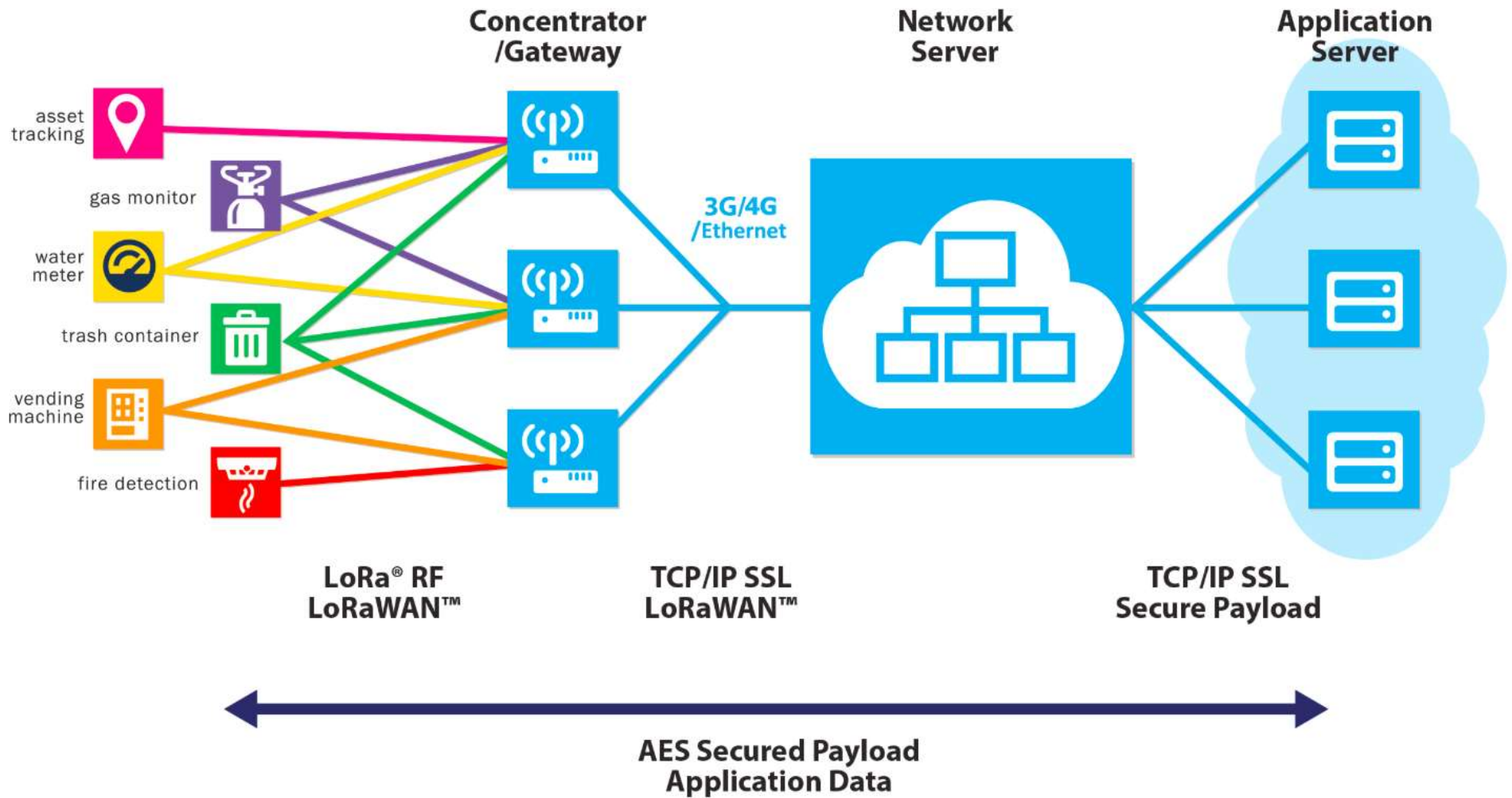
---

## Contents - Proof Of Concept

---

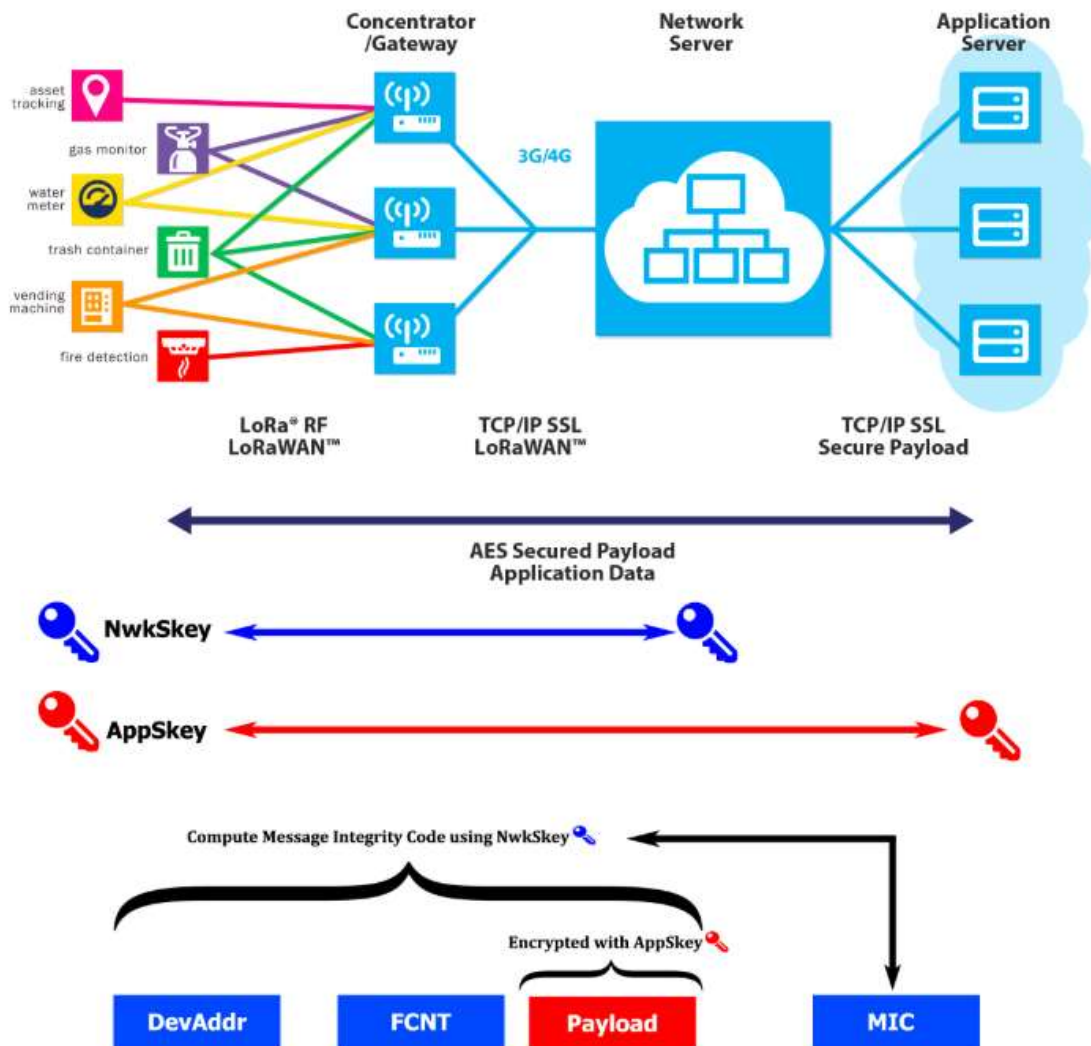
- **LoRaWAN™**
  - **POC Architecture**
  - **LoRa-RFID Devices**
  - **The Things Network**
  - **Web Service SAP oDATA**
  - **Node-Red sketch**
  - **SAP backend data reception**
  - **IBM IoT integration**
-

# LoRaWAN™ Network Structure



Specifiche Tecniche

# LoRaWAN™ Network Security



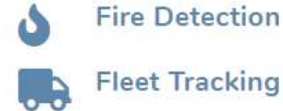
LoRaWAN™ works with two symmetrical key types for communication security, these are unique for each LoRa device. The **NwkSKey** is used in order to ensure message integrity from the device to the Network server.

**L'AppSKey** is used for AES-128 end-to-end encryption from the device to the Application Server.

# LoraWAN implementation for Waste Management

## Applicazioni LoRaWAN™

LoRaWAN™ è utilizzato in un'ampia gamma di settori



## ISM Frequency Bands

LoRaWAN™ utilizza le bande ISM regionali e gratuite per la trasmissione dei dati

✓ EU 863-870MHz

✓ AU 915-928MHz

✓ AS 923MHz

✓ EU 433MHz

✓ CN 779-787MHz

✓ KR 920-926MHz

✓ US 902-928MHz

✓ CN 470-510MHz

✓ IN 865-869MHz

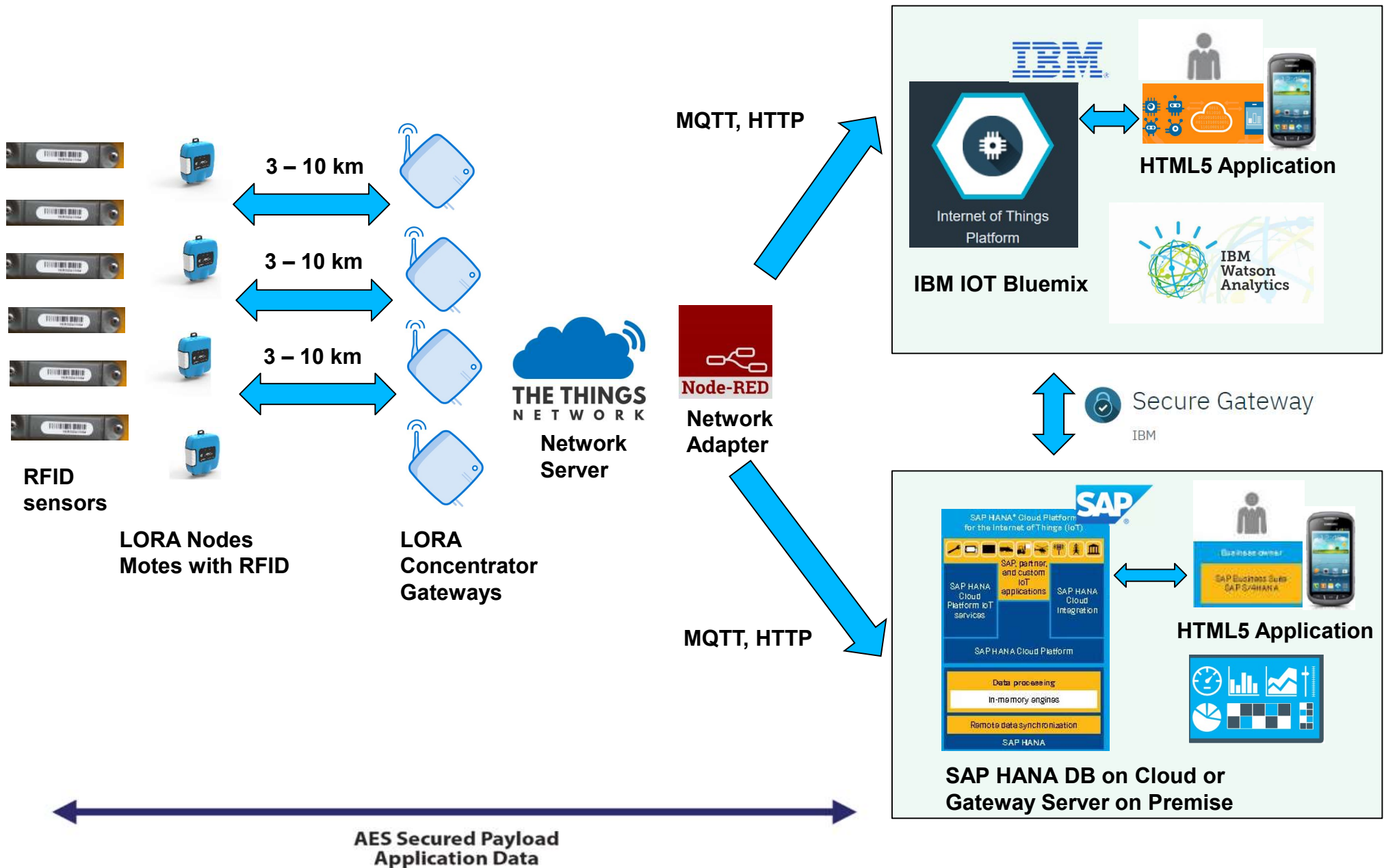
- Applicability:
  - Alerts communication
  - Status communication
- Potential applications:
  - GPS and tag reading
  - Massive transfer events sending
  - Configurations sending to sensors/bin caps
- Not applicable for:
  - Bin caps black list sending
  - Sensor firmware update (OTA)

# Proof Of Concept

---

- **LoRaWAN™**
  - **POC Architecture**
  - **LoRa-RFID Devices**
  - **The Things Network**
  - **Web Service SAP oDATA**
  - **Node-Red sketch**
  - **SAP backend data reception**
  - **IBM IoT integration**
-

# POC Architecture – Sending data either do IBM Bluemix or to SAP HANA Database



## Proof Of Concept

---

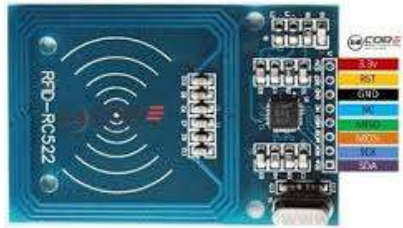
- **LoRaWAN™**
  - **POC Architecture**
  - **LoRa-RFID Devices**
  - **The Things Network**
  - **Web Service SAP oDATA**
  - **Node-Red sketch**
  - **SAP backend data reception**
  - **IBM IoT integration**
-



# RFID sensors used for the POC



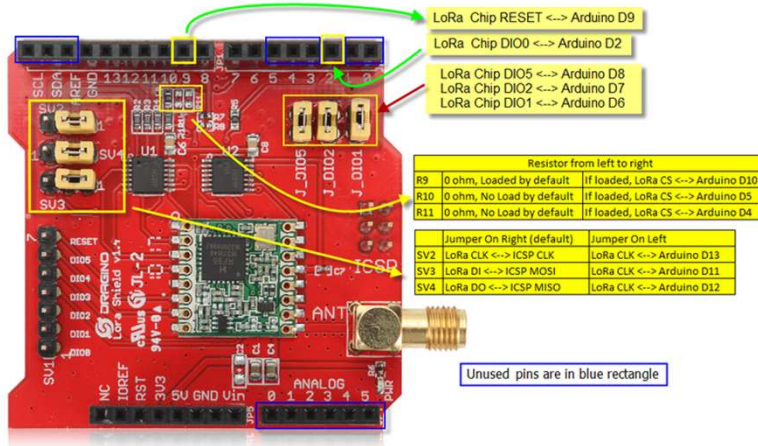
TAG RFID



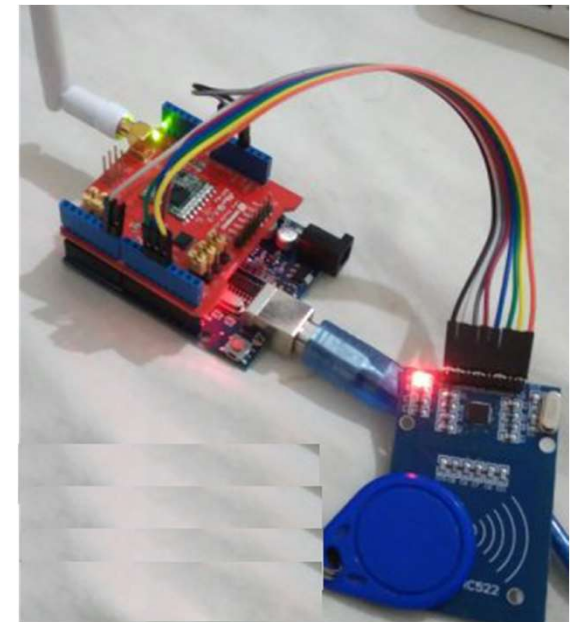
RFID MODULE	
3.3v	3.3v
RST (Reset)	5.3v
GND (Ground)	GND
NC	NC
MISO	MISO
MOSI	MOSI
SCK	SCK
SDA	SDA

RFID reader  
RC522

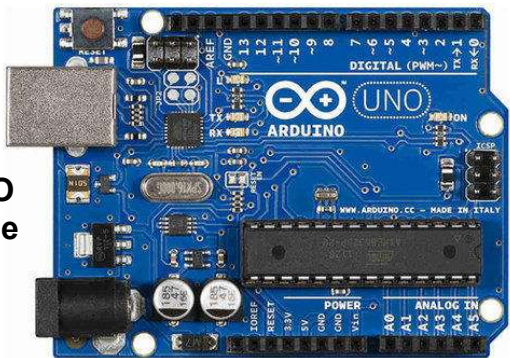
Pin Mapping For LoRa



Lora Shield with  
SX1278 868Mhz  
module



LORA-RFID final reader  
(Mote)



Arduino UNO  
Basic module

# LORA + RFID control firmware

```
1 /*
2  * Initial Author: ryand1011 (https://github.com/ryand1011)
3  * Added LORA component by Marco Moschella
4  *
5  * Reads data written by a program such as "rfid_write_personal_data.ino"
6  *
7  * See: https://github.com/miguelbalboa/rfid/tree/master/examples/rfid_write_personal_data
8  *
9  * Uses MIFARE RFID card using RFID-RC522 reader
10 * Uses MFRC522 - Library
11 *
12 *           MFRC522   Arduino   Arduino   Arduino   Arduino   Arduino
13 *           Reader/PCD Uno/101   Mega       Nano v3    Leonardo/Micro Pro Micro
14 * Signal    Pin      Pin       Pin       Pin       Pin       Pin
15 *-----
16 * RST/Reset RST      9         5         09        RESET/ICSP-5 RST
17 * SPI SS    SDA(SS) 10        53        010       10           10
18 * SPI MOSI  MOSI      11 / ICSP-4 51        011       ICSP-4       16
19 * SPI MISO  MISO      12 / ICSP-1 50        012       ICSP-1       14
20 * SPI SCK   SCK       13 / ICSP-3 52        013       ICSP-3       15
21 */
```

The control firmware works with these libraries:

- RFID reader library
- LMIC radiofrequency library
- Serial port SPI control library

The following parameters are manually set in the firmware (Activation by Personalization, i.e. ABP):

- NWKKEY[16] = server key
- APPSKEY[16] = application key
- DEVADDR[8] = device ID

These keys are used in order to send the message to the Network Server by LORA Gateway, which is at long distance from the device and is connected to the network (internet).

[https://github.com/landsat7/LORA\\_RFID/blob/master/RFID\\_reader\\_LORA\\_final.ino](https://github.com/landsat7/LORA_RFID/blob/master/RFID_reader_LORA_final.ino)

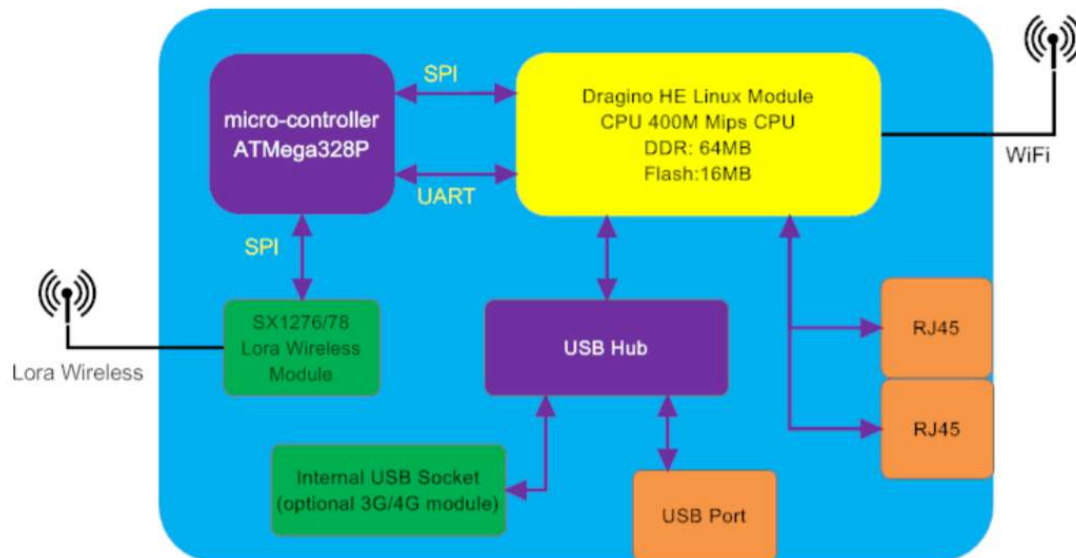
# Gateway type 1: LORA with DRAGINO LG01



We used a DRAGINO LG01 Gateway that allows to:

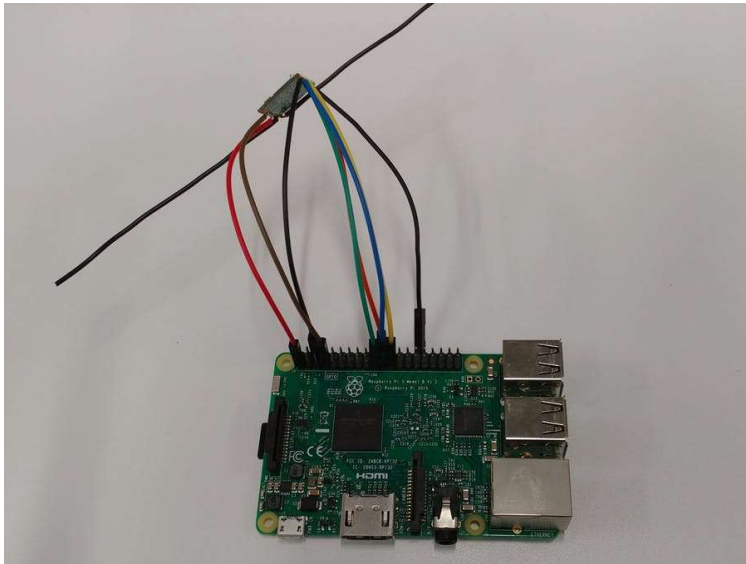
- receive radiofrequency signals from the sensors
- send them over the internet by a Wireless or RJ45 connection near the Gateway .

LG01 System Overview:



An ATmega328P processor is integrated in the Gateway, a SinglePacketForward firmware is installed on this processor by a connection with Arduino Software.

## Gateway type 2: LORA with Raspberry Pi model 3B and RFM95W transceiver



We used a Raspberry Pi Model 3B connected to a SX1278 SX1276 Lora Wireless Transceiver, 100mW RFM95W

Download the Single Packet Channel Forward software from the following website:

[https://github.com/tftelkamp/single\\_chan\\_pkt\\_fwd](https://github.com/tftelkamp/single_chan_pkt_fwd)

Then install the software using the «make» function and run it on the RaspBerry PI as follows.

```
pi@raspberrypi: ~/lorawan_gateway/single_chan_pkt_fwd
```

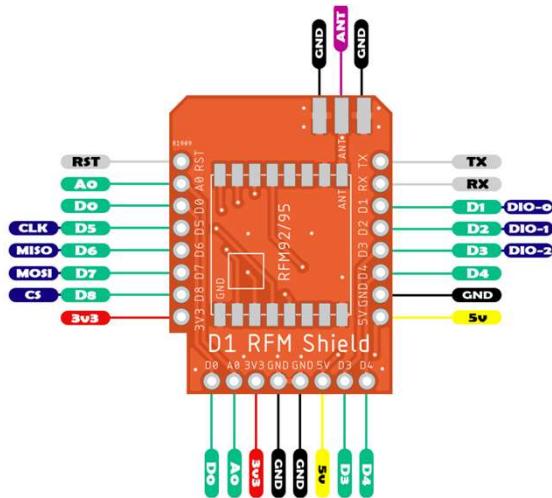
```
File Edit Tabs Help
```

```
pi@raspberrypi:~/lorawan_gateway/single_chan_pkt_fwd $ ls
base64.c base64.o main.cpp Makefile single_chan_pkt_fwd
base64.h LICENSE main.o README.md
pi@raspberrypi:~/lorawan_gateway/single_chan_pkt_fwd $ ./single_chan_pkt_fwd
SX1276 detected, starting.
Gateway ID: b8:27:eb:ff:ff:52:05:34
Listening at SF7 on 868.100000 Mhz.
-----
stat update: {"stat":{"time":"2019-12-28 13:58:15 GMT","lati":41.91512,"long":12.44185,"alti":100,"rxnb":0,"rxok":0,"rxfw":0,"ackr":0.0,"dwnb":0,"txnb":0,"pfrm":"Single Channel Gateway","mail":"marco.moschella@my-sap.it","desc":"RasPi3B with RMF95"}}
```

# Gateway type 3: LORA with Wemos D1 and RFM95W transceiver

We used a Wemos D1 connected to D1 RFM shield (that includes SX1276 Lora Wireless Transceiver, 100mW RFM95W).

Download the Single Packet Channel Forward software from the following website: <https://diycon.nl/archives/1911>



ESP-1ch-Gateway  
-v5.0-master



[Home](#) [Shop](#) [News](#) [My Account](#)

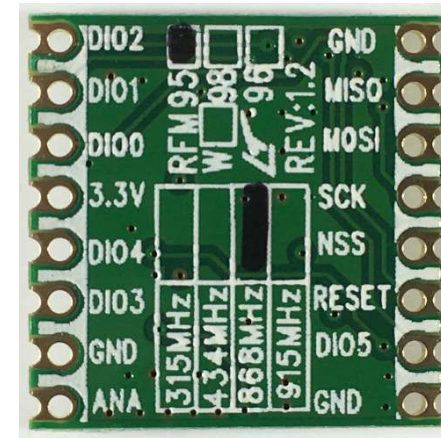
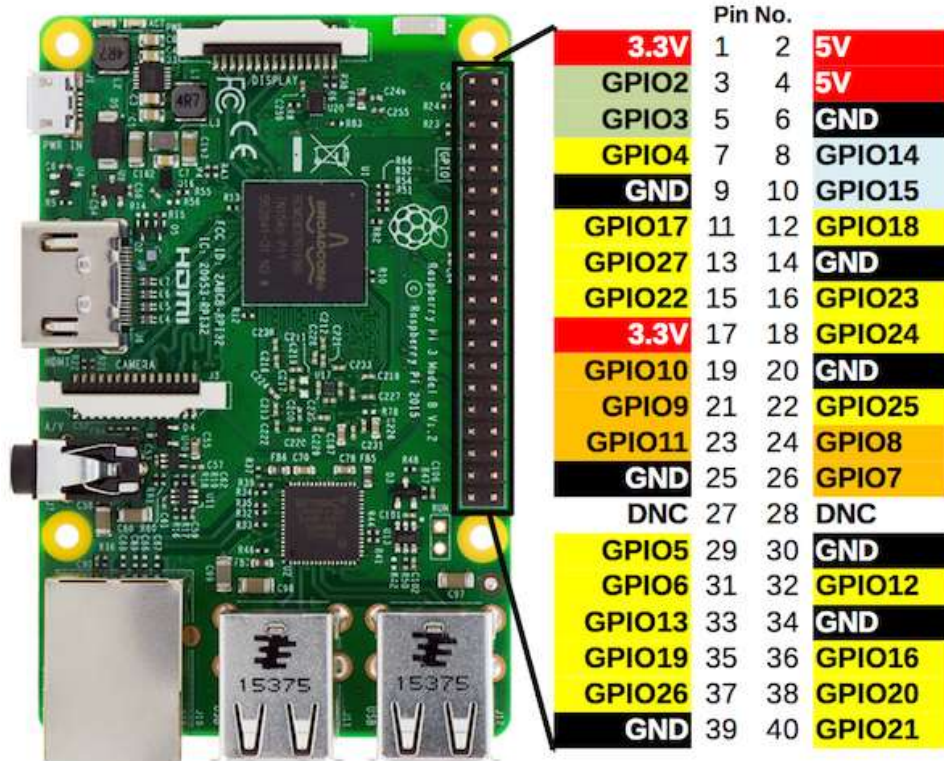
Thank you. Your order has been received.

## Order details

PRODUCT	TOTAL
<a href="#">LoRa GateWay PCB 301 Shield Only for RFM92/RFM95 Wemos D1</a> × 4	€15,80
SUBTOTAL:	€15,80
SHIPPING:	€3,00 via Without Tracking
PAYMENT METHOD:	PayPal
TOTAL:	€18,80

**Costs**

# Gateway LORA with Raspberry Pi model 3B and RFM95W transceiver – connection details



Pino Módulo LoRa	Pino Arduino
SCK	D13
MISO	D12
MOSI	D11
NSS	D10
(DIO)	D4
NRESET	D0

Rpi pin	Rpi descr.	RFM95W
1	3.3 V	3.3 V
6	GND	GND
7	GPIO4	DIO0
11	GPIO17	RESET
19	GPIO10	MOSI
21	GPIO09	MISO
22	GPIO25	NSS
23	GPIO11	SCK
		ANA (antenna 8,6cm)
		GND (antenna 8,6cm)

## Proof Of Concept

---

- **LoRaWAN™**
  - **POC Architecture**
  - **LoRa-RFID Devices**
  - **The Things Network**
  - **Web Service SAP oDATA**
  - **Node-Red sketch**
  - **SAP backend data reception**
  - **IBM IoT integration**
-

# The Things Network Manifest

---

Everything that carries power will be connected to Internet eventually.

Controlling the network that makes this possible means controlling the world. ***We believe that this power should not be restricted to a few people, companies or nations. Instead this should be distributed over as many people as possible without the possibility to be taken away by anyone.*** We therefore founded "The Things Network".

The Things Network is an open source, free initiative with the following properties:

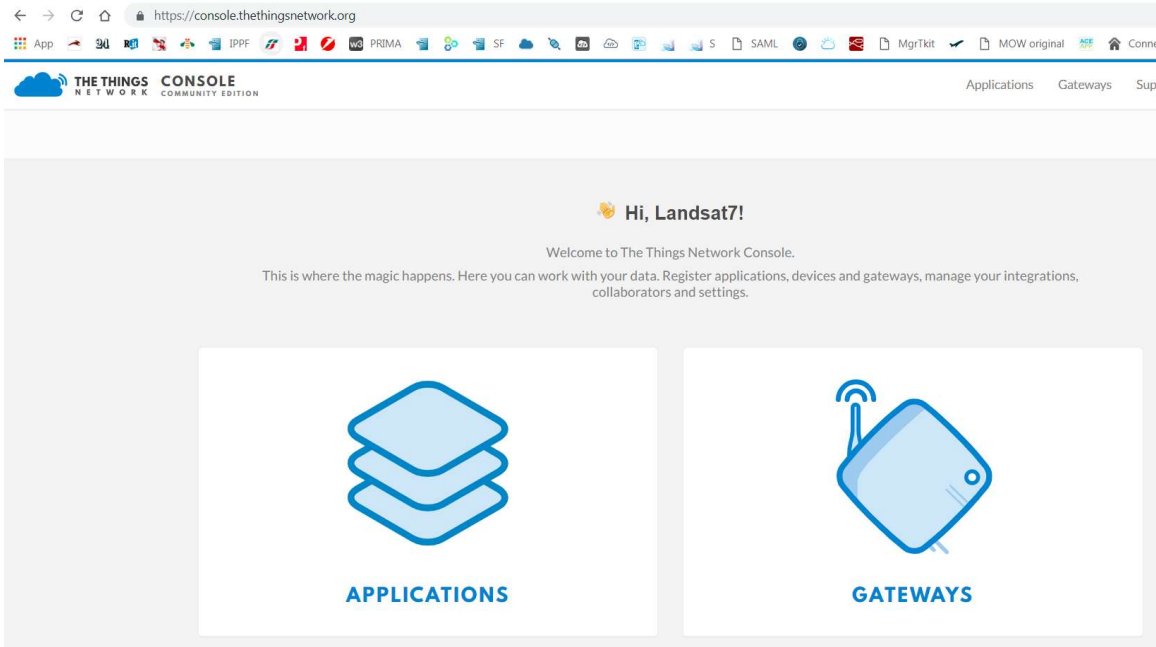
- It connects sensors and actuators, called "Things", with transceivers called "Things Gateways" to servers called "Things Access".
- The first connection is "Over The Air", the second is "Over The Net". The distributed implementation of these concepts is called "The Things Network".
- Anyone shall be free to set up "Things" and connect to "Things Gateways" that may or may not be their own.
- Anyone shall be free to set up "Things Gateways" and connect to "Things Access" that may or may not be their own. Their "Things Gateways" will give access to all "Things" in a net neutral manner, limited by the maximum available capacity alone.
- Anyone shall be free to set up "Things Access" and allow anonymous connections from the Internet. Their "Things Access" will give access to all "Things Gateways" in a net neutral manner, limited by the maximum available capacity alone. Furthermore their "Things Access" will allow connection of other "Things Access" servers for the distribution of data.
- The "Over The Air" and "Over The Net" networks shall be protocol agnostic, as long as these protocols are not proprietary, open source and free of rights.
- Anyone who perpetrates a "Things Access" or a "Things Gateway" will do so free of charge for all connecting devices and servers.
- Anyone making use of the network is allowed to do so for any reason or cause, possibly limited by local law, fully at own risk and realizing that services are provided "as is" and may be terminated for any reason at any moment. The use may be open for anybody, limited to customers, commercial, not-for-profit, or in any other fashion. "The Things Network" providers will not pose restrictions upon its users.

**We invite you to sign this Manifesto, and uphold its principles to the best of your abilities.**

---



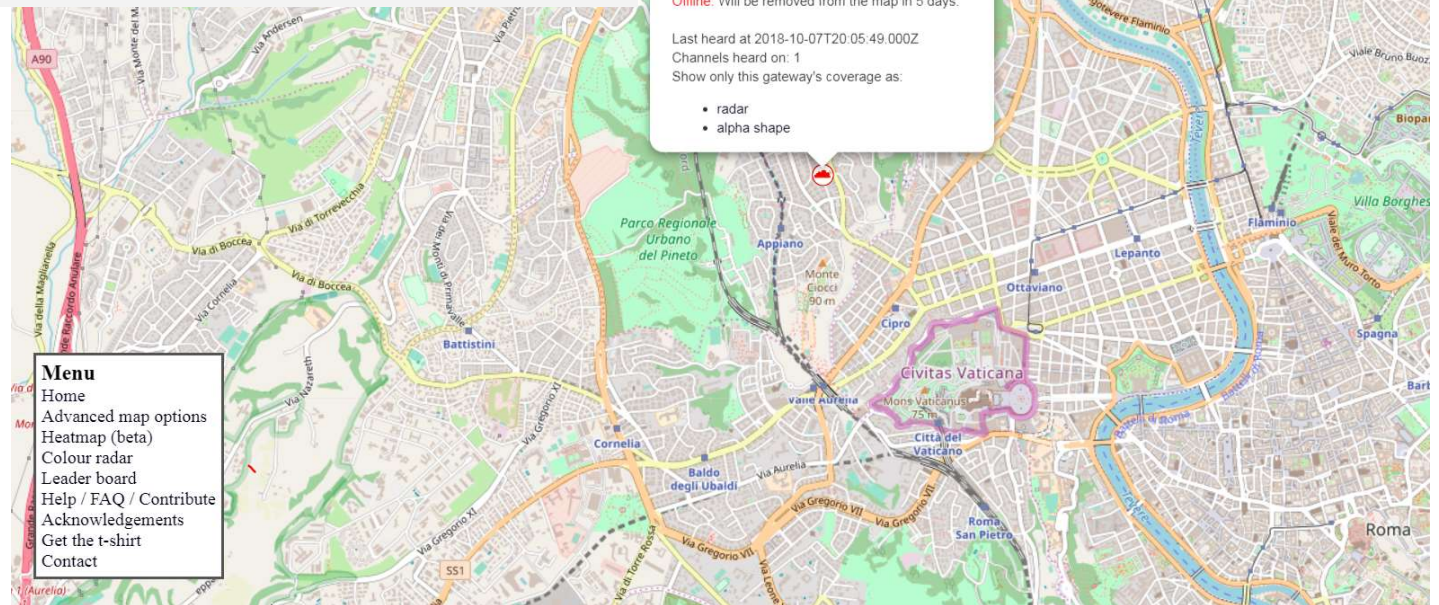
# Signal reception on TheThingsNetwork



The LORA Gateway could be logged on TTN and it is shown on the map.

Data traffic addressed by the Gateway to every visible sensor is shown on TTN.

Each Device Address could be logged on TTN so that we can receive data traffic when data are sent to the nearby Gateway.



# TTN payload decoding function for GPS node and RFID node

---

```
function Decoder(bytes, port) {
  // Decode an uplink message from a buffer
  // (array) of bytes to an object of fields.
  var decoded = {};
  // if (port === 1) decoded.led = bytes[0];
  decoded.latitude = ((bytes[0]<<16)>>>0) + ((bytes[1]<<8)>>>0) + bytes[2];
  decoded.latitude = (decoded.latitude / 16777215.0 * 180) - 90;
  decoded.longitude = ((bytes[3]<<16)>>>0) + ((bytes[4]<<8)>>>0) + bytes[5];
  decoded.longitude = (decoded.longitude / 16777215.0 * 360) - 180;
  var altValue = ((bytes[6]<<8)>>>0) + bytes[7];
  var sign = bytes[6] & (1 << 7);
  if(sign)
  {
    decoded.altitude = 0xFFFF0000 | altValue;
  }
  else
  {
    decoded.altitude = altValue;
  }
  decoded.hdop = bytes[8] / 10.0;
  return decoded;
}
```

```
function Decoder(bytes, port) {
  // Decode plain text; for testing only
  return {
    RFIDreading: String.fromCharCode.apply(null, bytes)
  };
}
```

---

# oDATA SAP service Zwa\_rfid\_srv service definition

[http://sapdemo/sap/opu/odata/sap/ZWA\\_RFID\\_SRV/\\$metadata](http://sapdemo/sap/opu/odata/sap/ZWA_RFID_SRV/$metadata)

The screenshot shows the SAP Gateway Service Builder interface. On the left, a tree view shows the project structure for 'ZWA\_RFID', including 'Data Model', 'Entity Types', 'Properties', and 'Navigation Properties'. The 'Properties' table is displayed in the center, listing various properties with their data types and ABAP field names. The 'Messages' pane at the bottom indicates that runtime objects for the project were generated successfully.

Name	Is Key	Edm Core Type	Prec.	Scale	Max	Unit Prop.	Creat.	Updat.	Sorta.	Nullab.	Flt.	Label	La	Comp. Type	ABAP Field Name	AB	Semantics
TimestampRece	<input checked="" type="checkbox"/>	Edm.DateTime	0	0	0		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				ZRECEPTION_T		
TransponderID	<input checked="" type="checkbox"/>	Edm.String	0	0	25		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				ZTRANSPONDE		
VehicleID	<input checked="" type="checkbox"/>	Edm.String	0	0	18		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				ZVEHICLE_ID		
RFIDcard	<input type="checkbox"/>	Edm.String	0	0	25		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				ZCARD_ID		
RFIDlatitude	<input type="checkbox"/>	Edm.Decimal	15	12	0		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>				ZGEOPOS_LATI		
RFIDlongitude	<input type="checkbox"/>	Edm.Decimal	15	12	0		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>				ZGEOPOS_LON		
RFIDtimestamp	<input type="checkbox"/>	Edm.DateTime	0	0	0		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>				ZGEOPOS_TIME		

```
<?xml version="1.0" encoding="utf-8" ?>
<edmx:Edmx xmlns:edmx="http://schemas.microsoft.com/ado/2008/09/edmx" ?>
  <edmx:DataServices m:DataServiceVersion="2.0">
    <Schema xmlns="http://schemas.microsoft.com/ado/2008/09/edmx" ?>
      <EntityType Name="RFIDreadings" sap:content-version="1">
        <Key>
          <PropertyRef Name="TimestampReceivedData"/>
          <PropertyRef Name="TransponderID"/>
          <PropertyRef Name="VehicleID"/>
          <PropertyRef Name="RFIDcard"/>
        </Key>
        <Property Name="TimestampReceivedData" Type="Edm.DateTime"
          sap:filterable="false"/>
        <Property Name="TransponderID" Type="Edm.String" Nullable="true"
          sap:filterable="false"/>
        <Property Name="VehicleID" Type="Edm.String" Nullable="true"
          sap:filterable="false"/>
        <Property Name="RFIDcard" Type="Edm.String" Nullable="true"
          sap:filterable="false"/>
        <Property Name="RFIDlatitude" Type="Edm.Decimal" Nullable="true"
          sap:filterable="false"/>
        <Property Name="RFIDlongitude" Type="Edm.Decimal" Nullable="true"
          sap:filterable="false"/>
        <Property Name="RFIDtimestamp" Type="Edm.DateTime" Nullable="true"
          sap:filterable="false"/>
      </EntityType>
      <EntityContainer Name="ZWA_RFID_SRV_Entities" m:IsDefault="true">
        <EntitySet Name="RFIDreadings" EntityType="ZWA_RFID_SRV_Entities.RFIDreadings"
          filter="true" sap:content-version="1"/>
      </EntityContainer>
    </Schema>
  </DataServices>
</edmx:Edmx>
```

## Proof Of Concept

---

- **LoRaWAN™**
  - **POC Architecture**
  - **LoRa-RFID Devices**
  - **The Things Network**
  - **Web Service SAP oDATA**
  - **Node-Red sketch**
  - **SAP backend data reception**
  - **IBM IoT integration**
-

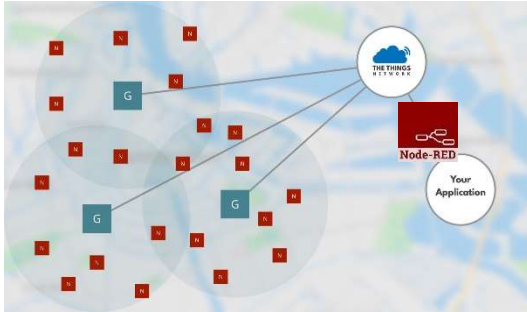


## Proof Of Concept

---

- **LoRaWAN™**
  - **POC Architecture**
  - **LoRa-RFID Devices**
  - **The Things Network**
  - **Web Service SAP oDATA**
  - **Node-Red sketch**
  - **SAP backend data reception**
  - **IBM IoT integration**
-

# Node-Red data mining from TTN Application Server and Get & Post call.



Node-Red adapter is easily configurable and enables:

- instant check of sent and received data
- data conversion, when required.
- In order to post to SAP a CSRF and token exchange is needed upfront

The screenshot shows the Node-RED interface with a flow titled 'TTN>SAP oDATA'. The flow starts with a 'TTN input from moschella-app2' node (highlighted with a red box), which is connected to a 'json' node. The 'json' node is connected to a 'Set global variables' node, which is then connected to a 'request CSRF token to SAP' node. This node is connected to a 'Call GET to SAP Gateway' node, which is connected to a 'Set token and cookie in msg headers' node. This node is connected to a 'to be posted to SAP' node, which is connected to a 'Call POST to SAP Gateway' node. This node is connected to an 'xml' node, which is connected to a 'Result final (XML)' node. The debug console shows the resulting message payload, which is highlighted with a red box:

```
airtime: 92416000
coding_rate: "4/5"
gateways: array[1]
12/10/2018, 21:31:54 node: to be posted to SAP
moschella-app2/devices/moschella-device5/up : msg.payload : Object
  object
  d: object
    TimestampReceivedData: "2018-10-12T19:31:46"
    TransponderID: "moschella-device5"
    VehicleID: "SF7BW125"
    RFIDcard: "95b36867"
    RFIDtimestamp: "2018-10-12T19:31:46"
12/10/2018, 21:31:55 node: Result final (XML)
moschella-app2/devices/moschella-device5/up : msg : Object
  object
  topic: "moschella-app2/devices/moschella-device5/up"
  payload: object
  qos: 0
  retain: false
  _msgid: "74eb049c.6e2e6c"
  headers: object
  statusCode: 201
  responseUrl:
    "http://9.137.251.96:8010/sap/opu/odata/sap/ZWA_RFID_SR"
```

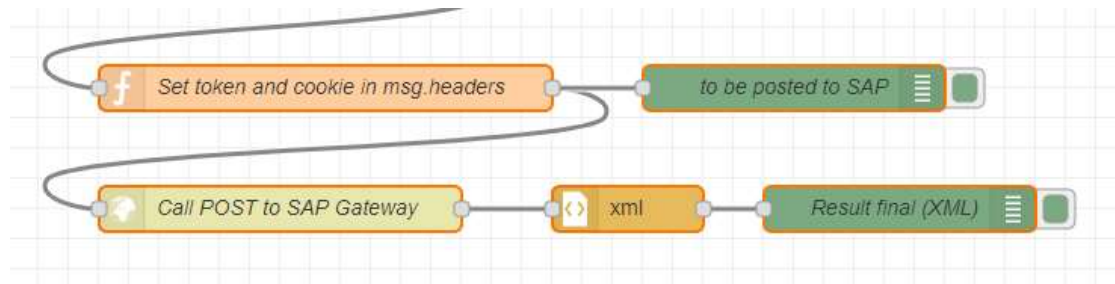
## Proof Of Concept

---

- **LoRaWAN™**
  - **POC Architecture**
  - **LoRa-RFID Devices**
  - **The Things Network**
  - **Web Service SAP oDATA**
  - **Node-Red sketch**
  - **SAP backend data reception**
  - **IBM IoT integration**
-



# oDATA SAP service execution results



12/10/2018, 21:31:54 node: to be posted to SAP

moschella-app2/devices/moschella-device5/up : msg.payload : Object

▼ object

▼ d: object

TimestampReceivedData: "2018-10-12T19:31:46"

TransponderID: "moschella-device5"

VehicleID: "SF7BW125"

RFIDcard: "95b36867"

RFIDtimestamp: "2018-10-12T19:31:46"

Insertione tabella Elaborare Passaggio a Parametrizzazione Utilities Ambiente Sistema Help

Data Browser: tabella ZWATB\_RFID 23 hit

Tabella: ZWATB\_RFID  
Campi visualizz.: 8 Da 8 Colonne iniziali fisse: 5 Lar. lista 0250

	MANDT	ZRECEPTION_TIMESTAMP	ZTRANSPONDER_ID	ZVEHICLE_ID	ZCARD_ID	ZGEOPOS_LATITUDE	ZGEOPOS_LONGITUDE	ZGEOPOS_TIMESTAMP
<input type="checkbox"/>	110	20.181.012.182.029	TRANSPONDER1000	CSL400	38d52b5#	0,000000000000	0,000000000000	20.181.012.071.533
<input type="checkbox"/>	110	20.181.012.182.029	moschella-device5	SF7BW125	838d52b5	0,000000000000	0,000000000000	20.181.012.182.029
<input type="checkbox"/>	110	20.181.012.182.230	moschella-device5	SF7BW125	838d52b5	0,000000000000	0,000000000000	20.181.012.182.230
<input type="checkbox"/>	110	20.181.012.183.050	moschella-device5	SF7BW125	efdb9e24	0,000000000000	0,000000000000	20.181.012.183.050
<input type="checkbox"/>	110	20.181.012.183.052	moschella-device5	SF7BW125	3098d0e4	0,000000000000	0,000000000000	20.181.012.183.052
<input type="checkbox"/>	110	20.181.012.183.054	moschella-device5	SF7BW125	3098d0e4	0,000000000000	0,000000000000	20.181.012.183.054
<input type="checkbox"/>	110	20.181.012.183.312	moschella-device5	SF7BW125	257125d9	0,000000000000	0,000000000000	20.181.012.183.312
<input type="checkbox"/>	110	20.181.012.183.334	moschella-device5	SF7BW125	257125d9	0,000000000000	0,000000000000	20.181.012.183.334
<input type="checkbox"/>	110	20.181.012.183.902	moschella-device5	SF7BW125	3098d0e4	0,000000000000	0,000000000000	20.181.012.183.902
<input type="checkbox"/>	110	20.181.012.183.952	moschella-device5	SF7BW125	257125d9	0,000000000000	0,000000000000	20.181.012.183.952
<input type="checkbox"/>	110	20.181.012.191.619	moschella-device5	SF7BW125	95b36867	0,000000000000	0,000000000000	20.181.012.191.619
<input type="checkbox"/>	110	20.181.012.191.625	moschella-device5	SF7BW125	95b36867	0,000000000000	0,000000000000	20.181.012.191.625
<input type="checkbox"/>	110	20.181.012.191.711	moschella-device5	SF7BW125	3098d0e4	0,000000000000	0,000000000000	20.181.012.191.711
<input type="checkbox"/>	110	20.181.012.191.814	moschella-device5	SF7BW125	95b36867	0,000000000000	0,000000000000	20.181.012.191.814
<input type="checkbox"/>	110	20.181.012.192.244	moschella-device5	SF7BW125	95b36867	0,000000000000	0,000000000000	20.181.012.192.244
<input type="checkbox"/>	110	20.181.012.192.302	moschella-device5	SF7BW125	3098d0e4	0,000000000000	0,000000000000	20.181.012.192.302
<input type="checkbox"/>	110	20.181.012.192.441	moschella-device5	SF7BW125	3098d0e4	0,000000000000	0,000000000000	20.181.012.192.441
<input type="checkbox"/>	110	20.181.012.192.446	moschella-device5	SF7BW125	95b36867	0,000000000000	0,000000000000	20.181.012.192.446
<input type="checkbox"/>	110	20.181.012.192.456	moschella-device5	SF7BW125	3098d0e4	0,000000000000	0,000000000000	20.181.012.192.456
<input type="checkbox"/>	110	20.181.012.192.504	moschella-device5	SF7BW125	95b36867	0,000000000000	0,000000000000	20.181.012.192.504
<input type="checkbox"/>	110	20.181.012.192.533	moschella-device5	SF7BW125	3098d0e4	0,000000000000	0,000000000000	20.181.012.192.533
<input type="checkbox"/>	110	20.181.012.192.547	moschella-device5	SF7BW125	95b36867	0,000000000000	0,000000000000	20.181.012.192.547
<input type="checkbox"/>	110	20.181.012.193.146	moschella-device5	SF7BW125	95b36867	0,000000000000	0,000000000000	20.181.012.193.146

## Proof Of Concept

---

- **LoRaWAN™**
  - **POC Architecture**
  - **LoRa-RFID Devices**
  - **The Things Network**
  - **Web Service SAP oDATA**
  - **Node-Red sketch**
  - **SAP backend data reception**
  - **IBM IoT integration**
-

# API key generation for IBM IoT secured communication

## Credenziali dispositivo

Il dispositivo è stato registrato nell'organizzazione. Aggiungere queste credenziali al dispositivo per collegarlo alla piattaforma. Una volta connesso il dispositivo, è possibile navigare per visualizzare i dettagli di connessione e degli eventi.

ID organizzazione	rhi7wb
Tipo di dispositivo	loranode
ID dispositivo	lora_device1
Metodo di autenticazione	use-token-auth
Token di autenticazione	wtN((*SL_23-maoP6g



I token di autenticazione non sono ripristinabili. Se si smarrisce il token, sarà necessario registrare nuovamente il dispositivo per generare un nuovo token di autenticazione.

Each sensor logged on IBM IoT has an authentication token so that we can monitor every login to the permitted applications on IBM IoT.

An API key will be generated to login on IBM IoT.

The screenshot shows the IBM Watson IoT Platform interface. The header includes the logo and the text 'App IBM Cloud'. A navigation sidebar on the left contains icons for various functions. The main content area displays a success message: 'La chiave API è stata aggiunta.' Below this, a warning icon and text state: 'I token di autenticazione non sono ripristinabili. Se si smarrisce il token, sarà necessario registrare nuovamente la chiave API per generare un nuovo token di autenticazione.' The interface is divided into two columns: 'Dettagli generati' and 'Informazioni chiave API'. The 'Dettagli generati' column lists the generated API key and authentication token. The 'Informazioni chiave API' column provides details about the key, including its description, role, and expiration date.

Dettagli generati		Informazioni chiave API	
Chiave API	a-rhi7wb-xljf2rbt56	Descrizione	Chiave API per applicazioni TTN
Token di autenticazione	Ge4iaKcntSl6y-HHSj	Ruolo	Applicazione dispositivo
		Scadenza	Mai

Prendere nota del token di autenticazione generato. I token di autenticazione perduti non possono essere recuperati. Se si perde il token, è necessario registrare nuovamente l'API per generare un nuovo token.

# Node-red sketch for sensor data sending to IBM IOT Dashboard

The screenshot displays the Node-RED web interface. On the left, the 'output' tab is selected, showing various nodes including 'debug', 'link', 'mqtt', 'http response', 'websocket', 'tcp', 'udp', and 'ibmiot'. The main workspace contains a flow with the following nodes: 'TTN real input' (purple), 'Simulation' (blue), 'JSON formatter' (orange), 'msg.payload' (green), and 'IBM IoT' (blue). The 'TTN real input' and 'IBM IoT' nodes are both marked as 'connected'. The flow is as follows: 'TTN real input' connects to 'IBM IoT'. 'Simulation' connects to 'JSON formatter', which connects to 'msg.payload'. The right sidebar shows a debug console with the following JSON object:

```
15/10/2018, 09:52:43 node: 869eb504.fd7b38
msg.payload: Object
  object
    app_id: "moschella-app1"
    dev_id: "moschella-device3"
    hardware_serial: "00C6FDC1FCEF2A99"
    port: 1
    counter: 22
    payload_raw: "u5y/iNjxAHUU"
  payload_fields: object
    myTestValue: "3ab4m251"
  metadata: object
    time: "2018-10-07T18:02:51.819193102Z"
    frequency: 868100000
    modulation: "LORA"
    data_rate: "SF7BW125"
    airtime: 56576000
    coding_rate: "4/5"
```

IBM IoT nodes can be added to the node-red sketch.

It is easily possible to retrieve or send data from/to IBM IoT devices.

# IBM IOT Device Analytics

## < Analisi incentrata sul dispositivo



+ Aggiungi nuova scheda



Impostazioni

The screenshot displays the IBM IoT Device Analytics dashboard with three main panels:

- Dispositivi a cui si presta attenzione** (Devices of interest): A table listing devices with columns for ID, type, and a filter icon.
- Proprietà del dispositivo** (Device properties): A list of specific data points for a selected device.
- Tutte le proprietà del disp...** (All device properties): A list of all available data points for the selected device.

ID dispositivo	Tipo di dispositivo	
112233445566	Android	
belt1	iot-conveyor-belt	
lora_device1	loranode	

Proprietà del dispositivo	
Nome dispositivo	lora_device1
dev_id	moschella-device3 (1 minuti fa)
payload_fields.myTestValue	3ab4m25l (1 minuti fa)

Tutte le proprietà del disp...	
Nome dispositivo	lora_device1
metadata.modulation	LORA (1 minuti fa)
metadata.time	2018-10-07T18:02:51.819193102Z (1 minuti fa)
payload_fields.myTestValue	3ab4m25l (1 minuti fa)
payload_raw	u5y/iNjxAHUU (1 minuti fa)
port	1 (1 minuti fa)

Incoming data to IBM IoT are displayed immediately in the IoT Dashboard.

# References

---

- <https://www.resiot.io/it/cosa-e-lorawan/>
  - [https://lora-alliance.org/sites/default/files/2018-04/lorawantm\\_specification\\_v1.1.pdf](https://lora-alliance.org/sites/default/files/2018-04/lorawantm_specification_v1.1.pdf)
  - <https://www.thethingsnetwork.org/>
  - <https://nodered.org/>
  - <https://nayarweb.com/blog/2017/rfid-and-lora-on-arduino/>
  - <http://blog.acorel.nl/2016/11/great-iot-opportunities-with-sap-and.html>
  - <https://www.hackster.io/ChrisSamuelson/lora-raspberry-pi-single-channel-gateway-cheap-d57d36>
  - [https://www.mobilefish.com/developer/lorawan/lorawan\\_quickguide\\_build\\_lora\\_node\\_rfm95\\_arduino\\_uno.html](https://www.mobilefish.com/developer/lorawan/lorawan_quickguide_build_lora_node_rfm95_arduino_uno.html)
-