

LoRa / LoRaWAN



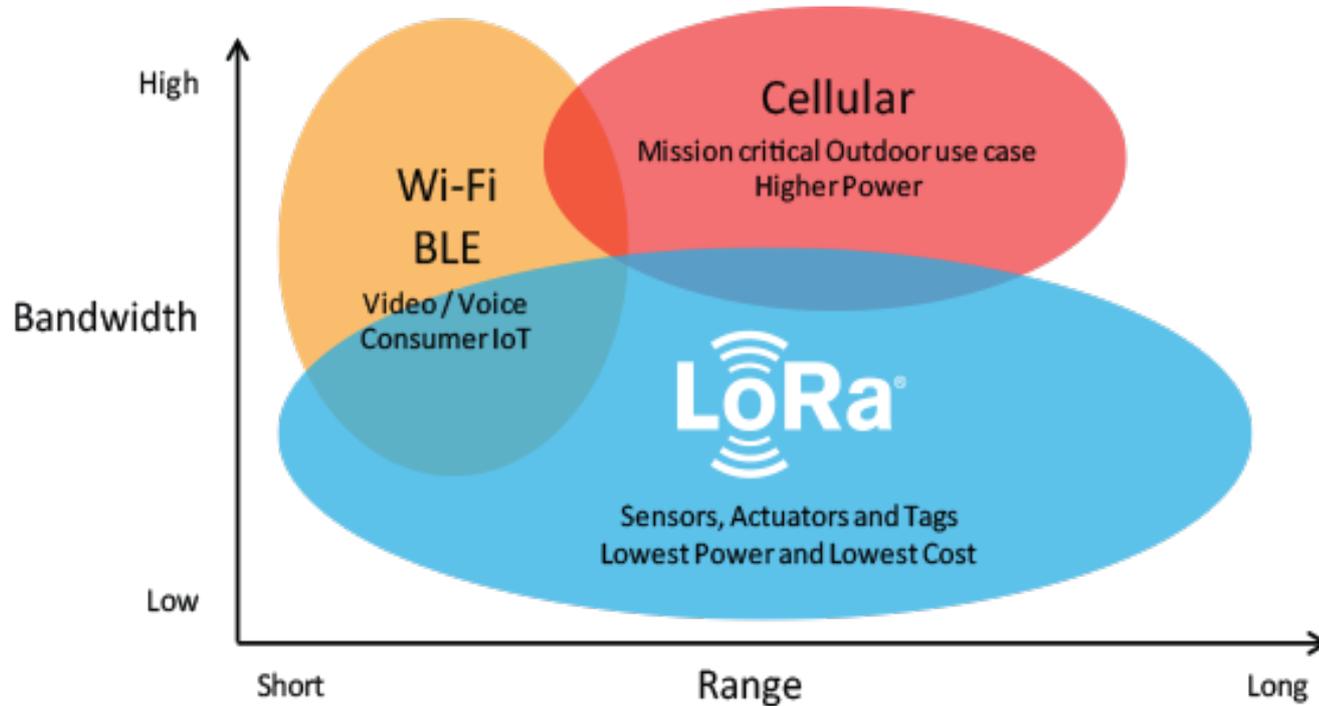
Quelle: www.semtech.com

Erstellt von Karl-Heinz Lohner

Was ist LoRa / LoRaWAN

- LoRa® definiert die Übertragungsschicht (Long Range)
- LoRaWAN® definiert das Netzwerk
(Long Range Wide Area Network)

Vergleich Funk-Übertragungstechniken



Quelle: www.semtech.com

Übersicht

- 1 - 9 LoRa Grundlagen
- 10 – 23 LoRaWAN Grundlagen
- 23 – 43 LoRaWAN – Aktivierung von End devices
- 44 – 50 TTN – Registrieren eines Gateway
- 51 – 52 TTN – Registrieren eines End Device
- 53 – 65 Chirpstack – Add device profile, Add application, Add Gateway, Add device
- 66 – 67 Links, Quellen

LoRaWAN Einführung

- Die Basis von LoRaWAN ist LoRa. Es zählt zu den LPWAN (Low Power Wide Area Networks).
- Mit dieser Übertragungstechnik können IoT-Geräte mit niedrigen Datenraten bei geringem Energieverbrauch und hoher Reichweite vernetzt werden.
- Batterielebensdauer bis zu 10 Jahren, realistisch 1-2 Jahre
- Für LoRa können in Europa folgende lizenzfreie Frequenzbänder benützt werden (ISM-Band):
 - 433 MHz (unidirektional)
 - 868 MHz (bidirektional) (863 - 870 MHz)
 - 2,4 GHz
 - 5 GHz

Übertragungstechnik LoRa

- Bidirektional, Uplink bevorzugt. Empfangsquittungen möglich.
- Modulation CSS (Chirp-Spread-Spectrum) und FSK (Frequenz Shift Keying) . FSK ist nur über einen Kanal möglich.
- In Europa werden die Frequenzbereiche 433 MHz und 868 MHz benutzt.
- Für CSS werden nebeneinander liegende Frequenzen (Kanäle) verwendet.
- Die Kanalbelegungsdauer ist reglementiert. Duty Cycle 1% D.h. 36 Sekunden Sendezeit pro Stunde.

Übertragungstechnik LoRa

- Die erzielbare Reichweite beträgt ca. 1 km bis 15 km. Abhängig von der Bebauung und der Topografie.
- Der Ruhestromverbrauch reicht von einigen hundert nA bzw. uA bis einige mA.
- Begrenzung der Sendeleistung auf +14 dBm bzw. maximal 25 mW.
- In der EU ist Payload auf 51 Byte begrenzt. (64Byte Paketlänge - 13 Byte Header).

Frequenzplan EU868

- Uplink:

- 868.1 - SF7BW125 to SF12BW125

- 868.3 - SF7BW125 to SF12BW125 and SF7BW250

- 868.5 - SF7BW125 to SF12BW125

- 867.1 - SF7BW125 to SF12BW125

- 867.3 - SF7BW125 to SF12BW125

- 867.5 - SF7BW125 to SF12BW125

- 867.7 - SF7BW125 to SF12BW125

- 867.9 - SF7BW125 to SF12BW125

- 868.8 – FSK

- Downlink:

- Uplink channels 1-9 (RX1)

- 869.525 - SF9BW125 (RX2)

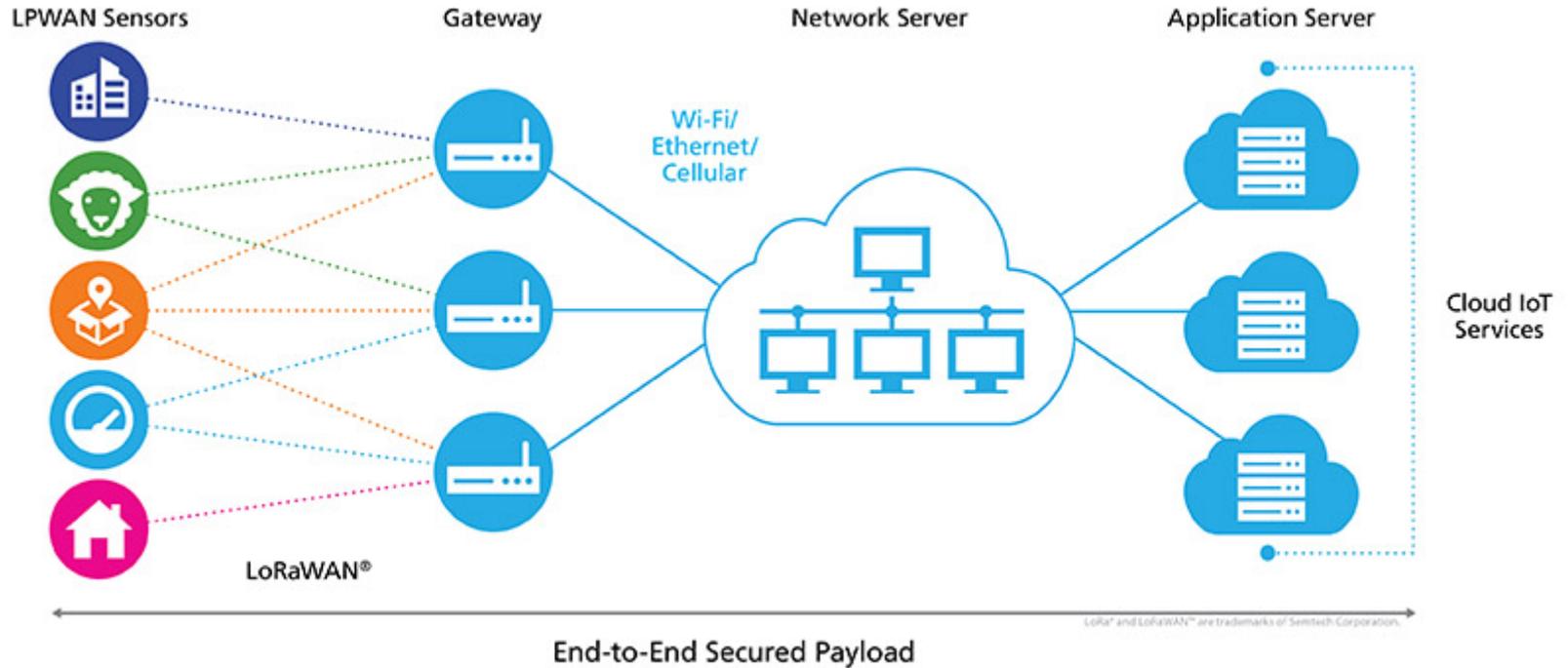
SF, Bandbreite, Bitrate

Spreizfaktor	Bandbreite	Bitrate	Dauer (64Bytes)	Reichweite
SF12	125 kHz	250 Bits/s	2470 ms	
SF11	125 kHz	440 Bits/s	1320 ms	> 2 km
SF10	125 kHz	980 Bits/s	700 ms	
SF9	125 kHz	1.760 Bits/s	390 ms	> 1 km
SF8	125 kHz	3.125 Bits/s	220 ms	
SF7	125 kHz	5.470 Bits/s	120 ms	100 – 500 m
	250 kHz	11.000 Bits/s		
FSK		50.000 Bits/s		

Anwendungsfälle von LoRaWAN

- Überwachung des Raumklimas (Temperatur, Luftfeuchte, co2-Gehalt, VoC)
- Fenstersensoren, Funksteckdosen usw.
- Landwirtschaft (Temperatur, Niederschlag, Bodenfeuchte usw.)
- Überwachung von Maschinen
- Smart City (z.B. Parkplätze, Verkehr)
- Überwachung der Kühlkette

LoRaWAN Architektur

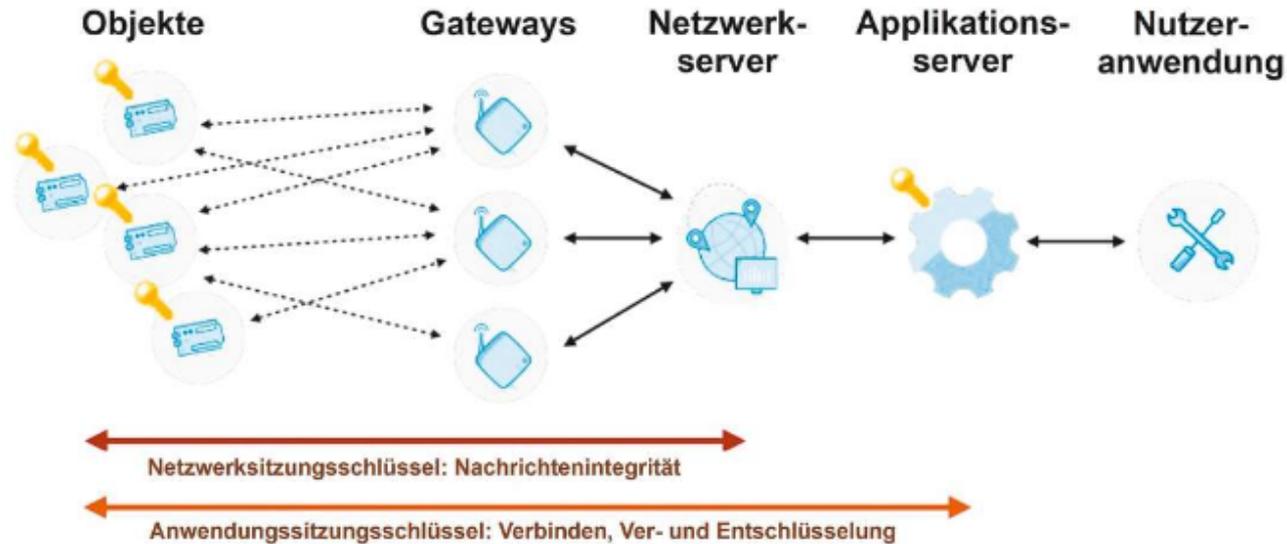


Quelle: <https://blog.semtech.com/>

LoRaWAN Architektur



Typische LoRaWAN-Topologie



Quelle: [3]

LoRaWAN Komponenten

- Ein LoRaWAN Netzwerk besteht aus drei Komponenten:
 - Sensoren (Nodes)
 - Gateway
 - Network- / Application-Server
- Die Nodes senden ihre Daten an alle Gateways in ihrer Umgebung. Die Gateways leiten diese an den/die Network-Server weiter. Diese schicken dann die Daten an Join- und Application-Server zur weiteren Verarbeitung.
- Die Datenübertragung erfolgt verschlüsselt. (nicht beim Join-Request)

LoRaWAN Komponenten

- Gateways
 - Ein oder mehrere Gateways empfangen die LoRa Nachrichten von den End-Devices
 - Es demoduliert die LoRa Nachricht
 - Leitet die Nachricht an den Network-Server weiter
- Network Server
 - Ist für die Authentifizierung und Identifizierung der End-Devices zuständig
 - Sendet die Antworten/Kommandos an die End-Devices
 - Kommuniziert mit Join-Server und Application-Server

- Application Server

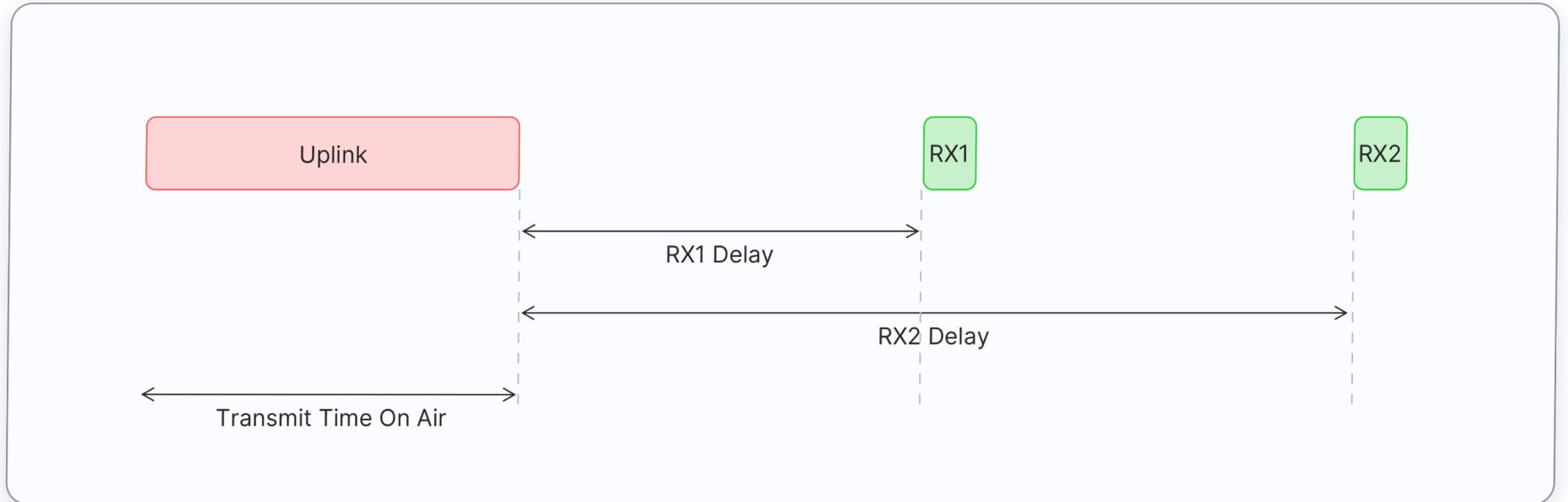
- Ist für die Anwendungsschicht des Protokolls zuständig
- Empfängt die Nachrichten vom Network Server und dekodiert diese
- Verschlüsselt die Nachrichten und sendet sie zum Network Server für den Downlink
- Bearbeitet AWS-, MQTT-, HTTP-Anfragen und Dashboard usw.

Geräte Klassen

- Es gibt drei Arten von Geräten: Klasse A, Klasse B und Klasse C. Alle LoRaWAN End-Devices müssen die Klasse-A-Implementierung unterstützen.
- End-Device der Klasse A, die oft batteriebetrieben sind und die meiste Zeit im Ruhezustand verbringen, können jederzeit eine Uplink-Nachricht senden. Sobald die Uplink-Übertragung abgeschlossen ist, öffnet das Gerät zwei kurze Empfangsfenster (RX1, RX2). Zwischen dem Ende der Uplink-Übertragung und dem Beginn der Empfangsfenster (RX1 oder RX2) besteht eine Verzögerung. Wenn der Netzwerkserver während dieser beiden Empfangsfenster nicht antwortet, erfolgt der nächste Downlink nach der nächsten Uplink-Übertragung.

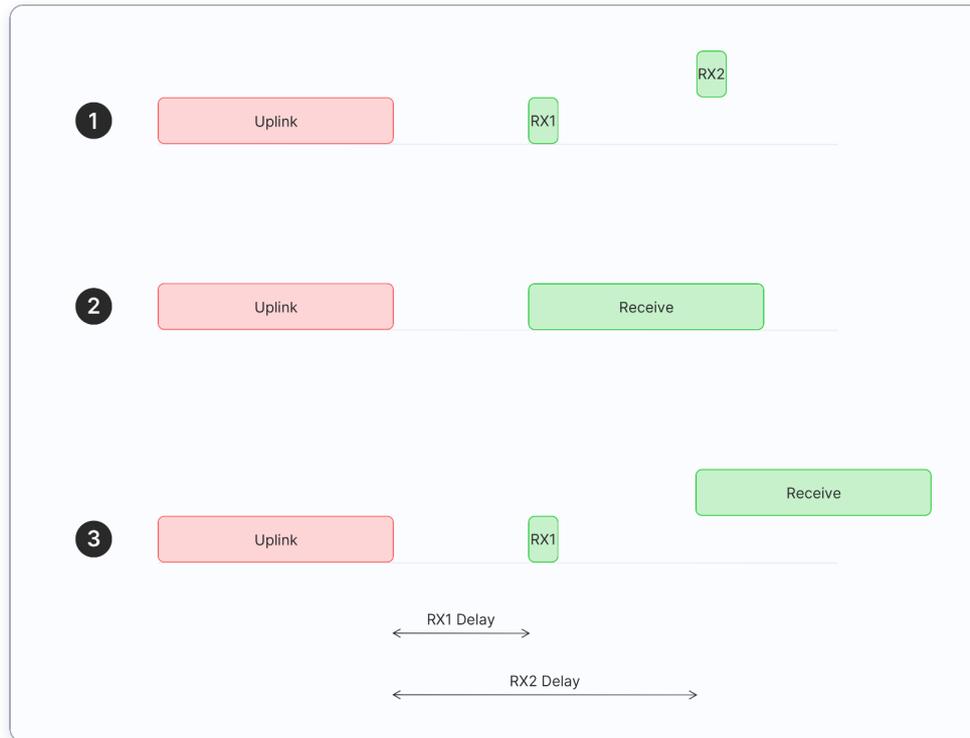
Quelle: [4]

Geräteklasse A



Quelle: [4]

Geräteklasse A



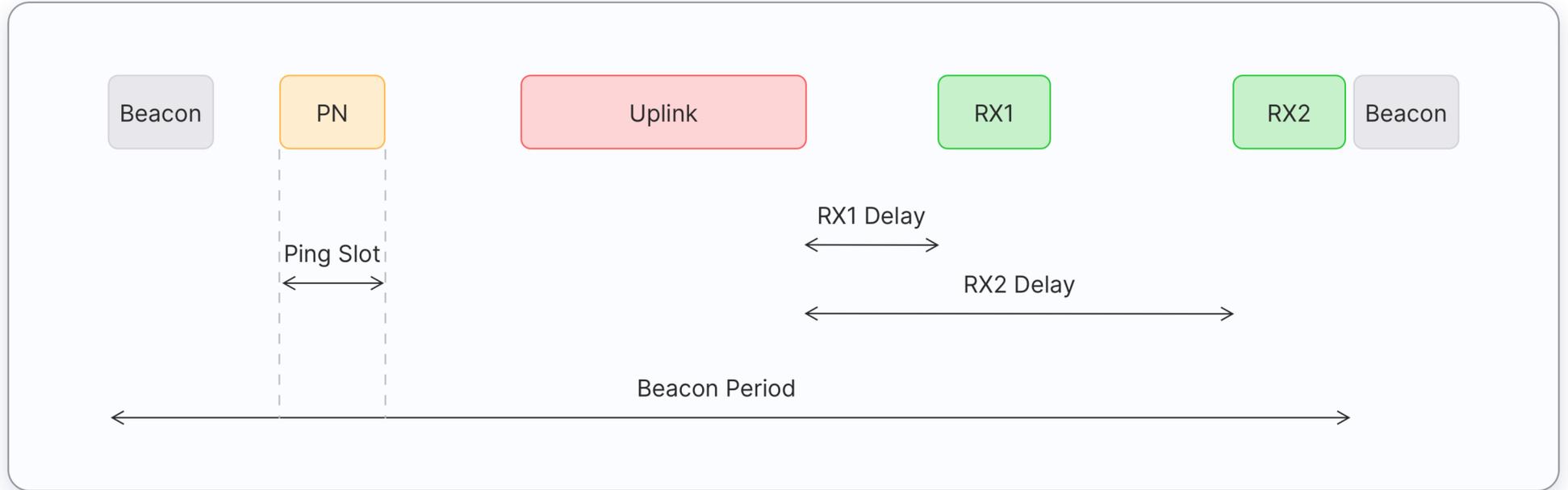
Quelle: [4]

Geräteklasse B

- Zusätzlich zu den Empfangsfenstern, die von Klasse A Geräten geöffnet werden, öffnen Geräte der Klasse B geplante Empfangsfenster, um Downlink-Nachrichten vom Netzwerkserver zu empfangen.
- Mit Hilfe von zeitsynchronisierten Beacons, die vom Gateway übertragen werden, öffnen die Geräte regelmäßig Empfangsfenster. Die Zeit zwischen zwei Beacons wird als Beacon Period bezeichnet. Das Gerät öffnet Downlink-"Ping-Slots" zu geplanten Zeiten für den Empfang von Downlink-Nachrichten vom Netzwerkserver.

Quelle: [4]

Geräteklasse B



- Die Beacon Period beträgt 128 Sekunden

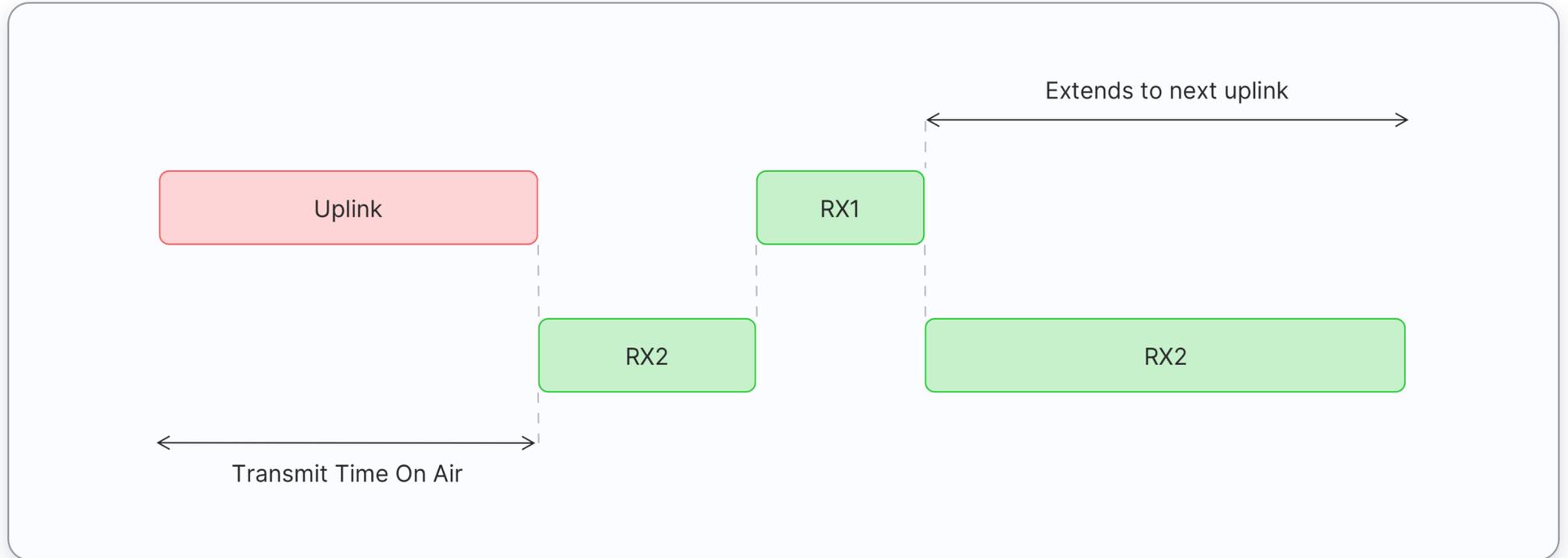
Quelle: [4]

Geräteklasse C

- Geräte der Klasse C erweitern die Klasse A, indem sie Empfangsfenster offen halten, wenn sie nicht senden, wie in der Abbildung unten dargestellt. Dies ermöglicht eine Kommunikation mit geringer Latenz, verbraucht aber ein Vielfaches an Energie im Vergleich zu Geräten der Klasse A.

Quelle: [4]

Geräteklasse C



Quelle: [4]

Aktivierung der End-Devices

- Jedes End-Device muss in einem Netz registriert werden, bevor es Nachrichten senden und empfangen kann. Dieser Vorgang wird als Aktivierung bezeichnet.
- Es sind zwei Aktivierungsmethoden verfügbar:
- Over-The-Air Activation (OTAA) - die sicherste und empfohlene Aktivierungsmethode für End-Devices.
 - Während der Aktivierung wird dem Gerät eine dynamische Geräteadresse zugewiesen und die Keys zur Verschlüsselung erzeugt.

Quelle: [4]

Aktivierung der End-Devices

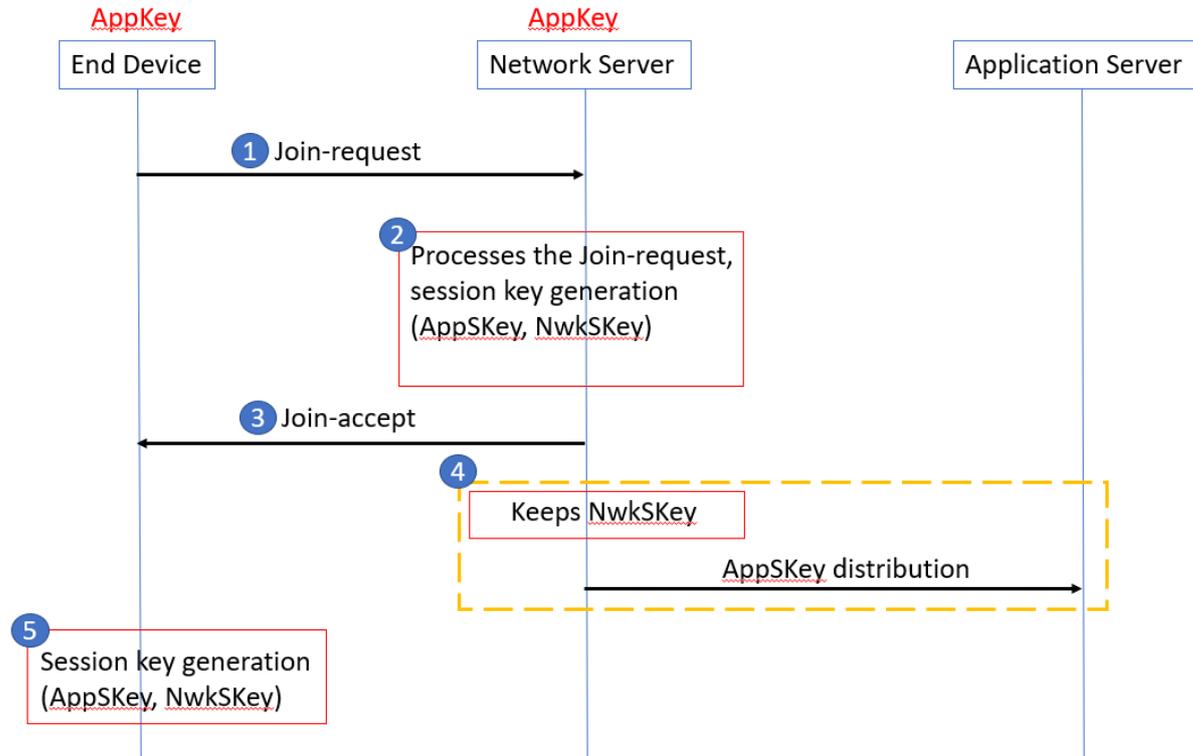
- Aktivierung durch Personalisierung (ABP) - erfordert die harte Codierung der Geräteadresse und der Sicherheitsschlüssel im Gerät. ABP ist weniger sicher als OTAA.
- Ein ABP-Gerät in einem öffentlichen Netzwerk muss die vom Netzwerkservers bereitgestellte DevAddr verwenden. Dies bedeutet, dass das Gerät an dieses Netzwerk gebunden ist und nicht in ein anderes Netzwerk verschoben werden kann.

Quelle: [4]

OTAA Aktivierung LoRaWAN Version 1.0.x

- 'join request' vom End-Device zum Network-Server
- 'join-accept' vom Network-Server zum End-Device
- Vor der Aktivierung müssen **AppEUI**, **DevEUI** und **AppKey** dem End-Device zugewiesen sein.
- Der **AppKey** ist ein AES-128 bit Schlüssel, er wird auch als 'root key' bezeichnet. Der gleiche **AppKey** muss im End-Device und im Network-Server bekannt sein. Der **AppKey** wird nicht übertragen.
- Die 'join request' Nachricht mit einer beliebigen Datenrate über die regionalen 'Join Kanäle' gesendet werden. In Europa sind dies 868,10 MHz, 868,30 Mhz oder 868,50 Mhz.

OTAA Aktivierung LoRaWAN Version 1.0.x



Quelle: [2]

OTAA Aktivierung LoRaWAN Version 1.0.x

- Der Join Prozess wird immer vom End-Device gestartet.
 - Das End-Device sendet dafür folgende Daten zum Network-Server
8 bytes 8 bytes 2 bytes
AppEUI DevEUI DevNonce (Number used once)
 - **AppEUI** – Ein global eindeutiger 64-Bit 'application identifier' im IEEE EUI64 Adressraum
 - **DevEUI** – Ein global eindeutiger 64-Bit 'device identifier' im IEEE EUI64 Adressraum
 - **DevNonce** – Ein eindeutiger, zufälliger 2 Byte Wert, vom End-Device erzeugt. Der Network-Server verfolgt anhand des DevNonce die 'join-requests' vom End-Device. Bekommt der Network-Server einen 'join-request' der schon einmal von diesem Gerät benutzt wurde, weist er diesen mit 'reject' ab.
 - Die Daten des 'join request' sind nicht verschlüsselt.

OTAA Aktivierung LoRaWAN Version 1.0.x

- Der Join-Server bearbeitet den 'join request' und generiert zwei 'session keys', (**NwkSKey** and **AppSKey**) und die 'join accept' Nachricht, falls das End-Device dem Netzwerk beitreten darf.

3 bytes	3 bytes	4 bytes	1 byte	1 byte	16 bytes (optional)
AppNonce	NetID	DevAddr	DLSettings	RXDelay	CFList

- **AppNonce** – Eine zufällige und eindeutige ID vom Network-Server. Wird vom End-Device benutzt, um AppSKey und NwkSKey daraus abzuleiten.
- **NetID** – Ein 24-bit eindeutiger 'network identifier'.
- **DevAddr** – Eine 32-Bit 'device address' – vom Network-Server zugewiesen, um das End-Device im aktuellen Netzwerk zu identifizieren.
- **DLSettings** – Ein 1 Byte Feld das die 'downlink settings', die das Gerät benützen soll, beschreibt.

OTAA Aktivierung LoRaWAN Version 1.0.x

- **RxDelay** – Die Verzögerung zwischen TX und RX.
- **CFList** – Ein optionale Liste mit Frequenzen die das End-Device benutzen darf. Ist Regionsspezifisch.
- **MIC** -- (Message Integrity Code) wird mit dem AppKey über alle Felder der 'join accept' Nachricht berechnet und an die 'join accept' Nachricht angehängt.
- Die 'join accept' Nachricht ist mit dem AppKey verschlüsselt.
- Der Network-Server sendet die 'join accept' Nachricht als ganz normalen Downlink.
- Falls der 'join request' nicht akzeptiert wird, wird keine Nachricht an das End-Device gesendet.
- Der Network-Server merkt sich den NwkSKey und schickt den AppSKey zum Application-Server

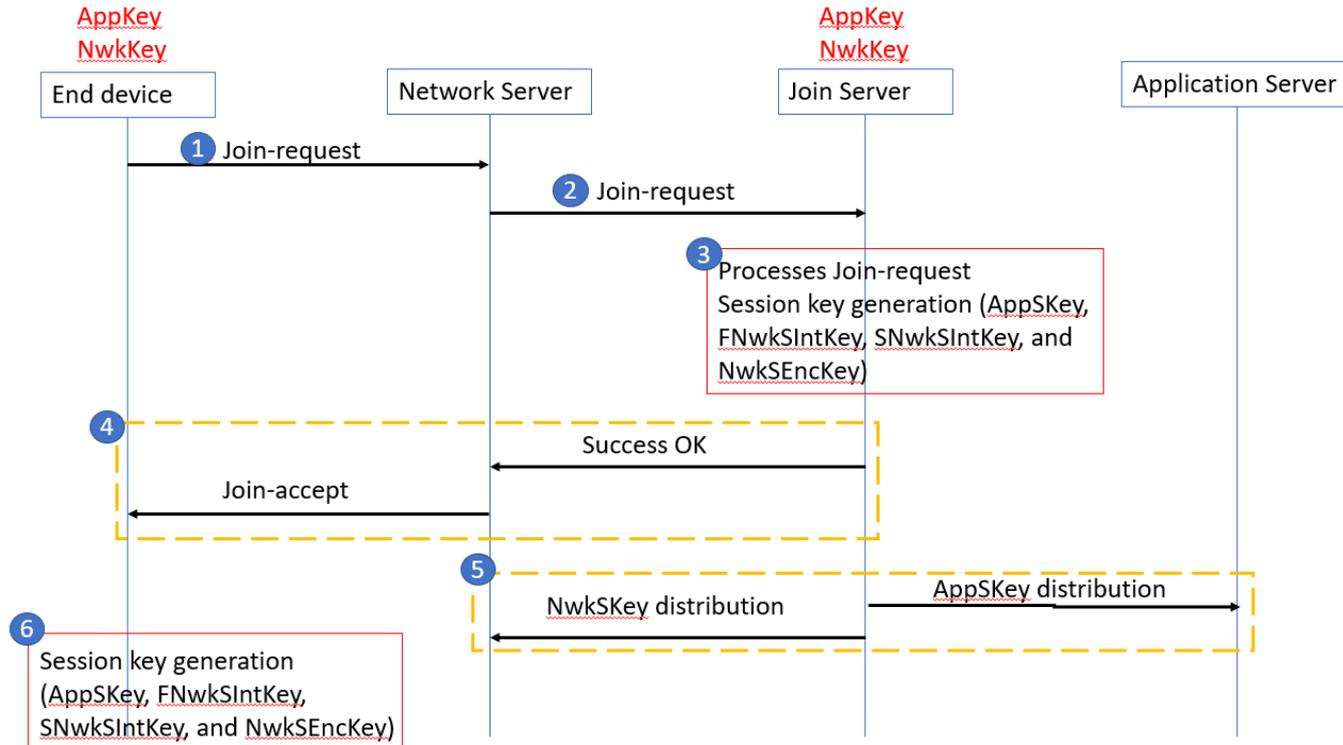
OTAA Aktivierung LoRaWAN Version 1.0.x

- Das End-Device ist jetzt im Netzwerk aktiviert.
- Folgende Informationen werden im End-Device gespeichert
 - **DevAddr** – Eine 32-Bit 'device address' – vom Network-Server zugewiesen, um das End-Device im aktuellen Netzwerk zu identifizieren.
 - **NwkSKey** – Der 'network Session Key' wird vom End-Device und vom Netzwerkserver verwendet, um den Message Integrity Code (MIC) aller Datennachrichten zu berechnen und zu überprüfen, um die Nachrichtenintegrität sicherzustellen. Der NwkSKey wird auch zum Verschlüsseln und Entschlüsseln von Nutzdaten mit MAC-Befehlen verwendet.
 - **AppSKey** – Der 'Application Session Key' wird zum Verschlüsseln und Entschlüsseln der Anwendungsdaten (payload) vom / zum 'application server' verwendet. Damit wird die Vertraulichkeit der Daten sichergestellt.

OTAA Aktivierung LoRaWAN Version 1.1

- 'join request' vom End-Device zum Network-Server
- 'join accept' vom Network-Server zum End-Device
- Vor der Aktivierung müssen **JoinEUI**, **DevEUI**, **AppKey** und **NwkKey** müssen im End-Device gespeichert sein.
- **AppKey** und **NwkKey** sind AES-128 Bit Schlüssel und werden als **root keys** bezeichnet.
- Ein MIC (Message Integrity Code) wird mit dem NwkKey über alle Felder der 'join request' Nachricht berechnet und an die 'join request' Nachricht angehängt.
- **AppKey** und **NwkKey** werden **niemals** über das Netzwerk gesendet.

OTAA Aktivierung LoRaWAN Version 1.1



Quelle: [2]

OTAA Aktivierung LoRaWAN Version 1.1

- Der Join Prozess wird immer von End-Device gestartet.
 - Das End-Device sendet dafür folgende Daten zum Network-Server
8 bytes 8 bytes 2 bytes
JoinEUI DevEUI DevNonce
 - **JoinEUI** – Ein global eindeutiger 64-Bit 'application identifier' im IEEE EUI64 Adressraum. Der 'Join Server' bearbeitet den 'join request' und erzeugt die Session Keys.
 - **DevEUI** – Ein global eindeutiger 64-Bit 'device identifier' im IEEE EUI64 Adressraum
 - **DevNonce** – Ein eindeutiger, zufälliger 2 Byte Wert, von End-Device erzeugt. Der Network-Server verfolgt anhand des DevNonce die 'join-requests' vom End-Device. Bekommt der Network-Server einen 'join request' der schon einmal von diesem Gerät benutzt wurde, weist er diesen mit 'reject' ab. Das **DevNonce** soll 'replay attacken' verhindern.
 - Die Daten des 'join request' sind nicht verschlüsselt.

OTAA Aktivierung LoRaWAN Version 1.1

- In LoRaWAN 1.1 Ist die **AppEUI** durch **JoinEUI** ersetzt.
- Die 'join request' Nachricht mit einer beliebigen Datenrate über die regionalen 'Join Kanäle' gesendet werden. In Europa sind dies 868,10 MHz, 868,30 Mhz oder 868,50 Mhz.
- Die 'join request' Nachricht kann ein oder über mehrere Gateways zum 'network server' übertragen werden.
- Wenn der 'join request' nicht akzeptiert wird, wird keine Antwort zum 'end device' geschickt.

OTAA Aktivierung LoRaWAN Version 1.1

- Der 'network server' leitet die 'join request' Nachricht an den zuständigen 'join server' weiter.
- Der 'join server' bearbeitet den 'join request'. Der 'join server' generiert alle erforderlichen session keys (**AppSKey**, **FNwkSIntKey**, **SNwkSIntKey**, und **NwkSEncKey**) falls das 'end device' in das Netzwerk integriert wird.
- Wenn der 'join request' erfolgreich ist, sendet der 'network server' eine 'join accept' Nachricht mit folgendem Inhalt an das 'end device':

1 byte	3 bytes	4 bytes	1 bytes	1 bytes	16 bytes
JoinNonce	NetID	DevAddr	DLSettings	RXDelay	CFList

- **JoinNonce** – Ein gerätespezifischer Zähler der vom 'join server' vergeben ist und vom 'end device' benutzt wird, um die 'session keys' **FNwkSIntKey**, **SNwkSIntKey**, **NwkSEncKey** und **AppSKey** abzuleiten.
- **NetID** – Ein 24-bit eindeutiger 'network identifier'.
- **DevAddr** – Eine 32-Bit 'device address' – vom Network-Server zugewiesen, um das End-Device im aktuellen Netzwerk zu identifizieren.

OTAA Aktivierung LoRaWAN Version 1.1

- **DLSettings** – Ein 1 Byte Feld das die 'downlink settings', die das Gerät benutzen soll, beschreibt.
- **RxDelay** – Die Verzögerung zwischen TX und RX.
- **CFList** – Ein optionale Liste mit Frequenzen die das End-Device benutzen darf. Ist Regionsspezifisch.
- **MIC** -- Der Message Integrity Code (MIC) wird über alle Felder in der 'join accept' Nachricht mit dem NwkKey (für LoRaWAN 1.0 devices) oder JSIntKey (für LoRaWAN 1.1 devices) errechnet. Der errechnete MIC wird an die 'join accept' Nachricht angehängt.
- Die 'join accept' Nachricht wird dann mit dem **NwkKey** (falls ein 'join request' angefordert wurde) oder dem **JSEncKey** (bei einem 'rejoin request') verschlüsselt. Der 'Network Server' benutzt eine AES decrypt operation in ECB mode um die 'join accept' Nachricht zu verschlüsseln.
- Dann sendet der 'Network Server' die verschlüsselte 'join accept' Nachricht zum 'end device' als ganz normalen Downlink.
- Wenn der 'join request' nicht akzeptiert wird, wird keine Nachricht zum 'end device' gesendet.

OTAA Aktivierung LoRaWAN Version 1.1

- Der 'Join Server' sendet den **AppSKey** zum 'Application Server', und die drei network session keys (**FNwkSIntKey**, **SNwkSIntKey**, and **NwkSEncKey**) zum 'Network Server'.
- Das 'end-device' entschlüsselt die 'Join-accept' Nachricht mit einer AES decrypt operation. Das 'end device' benutzt **AppKey**, **NwkKey**, und **JoinNonce** um die 'session keys' zu generieren.
- Für **LoRaWAN 1.0.x** Geräte:
 - Der **AppSKey** ist vom **NwkKey** abgeleitet.
 - **FNwkSIntKey**, **SNwkSIntKey**, und **NwkSEncKey** werden auch vom **NwkKey** abgeleitet.
- Für **LoRaWAN 1.1** Geräte:
 - Der **AppSKey** ist vom **AppKey** abgeleitet.
 - **FNwkSIntKey**, **SNwkSIntKey** und **NwkSEncKey** werden vom **NwkKey** abgeleitet.
- Das End-Device ist jetzt im Netzwerk aktiviert.

OTAA Aktivierung LoRaWAN Version 1.1

Nach der Aktivierung sind folgende Informationen im End-Device gespeichert:

- **DevAddr** - Eine 32-Bit 'device address' – vom Network-Server zugewiesen, um das End-Device im aktuellen Netzwerk zu identifizieren.
- **FNwkSIntKey** – Ein 'network session key' der vom End-Device verwendet wird, um den MIC (teilweise) von allen Uplink Nachrichten zu berechnen.
- **SNwkSIntKey** – Ein 'network session key' der vom End-Device verwendet wird, um den MIC (teilweise) von allen Uplink Nachrichten und den MIC Downlink Daten Nachrichten zu berechnen.
- **NwkSEncKey** -- Ein 'network session key' der benutzt wird, um die Nutzdaten von MAC Kommandos zu verschlüsseln und zu entschlüsseln.
- **AppSKey** – Ein 'session key'. Vom 'Application Server' und vom 'end device' verwendet, um die 'application data' zu verschlüsseln und zu entschlüsseln. Damit wird die Vertraulichkeit sicher gestellt.

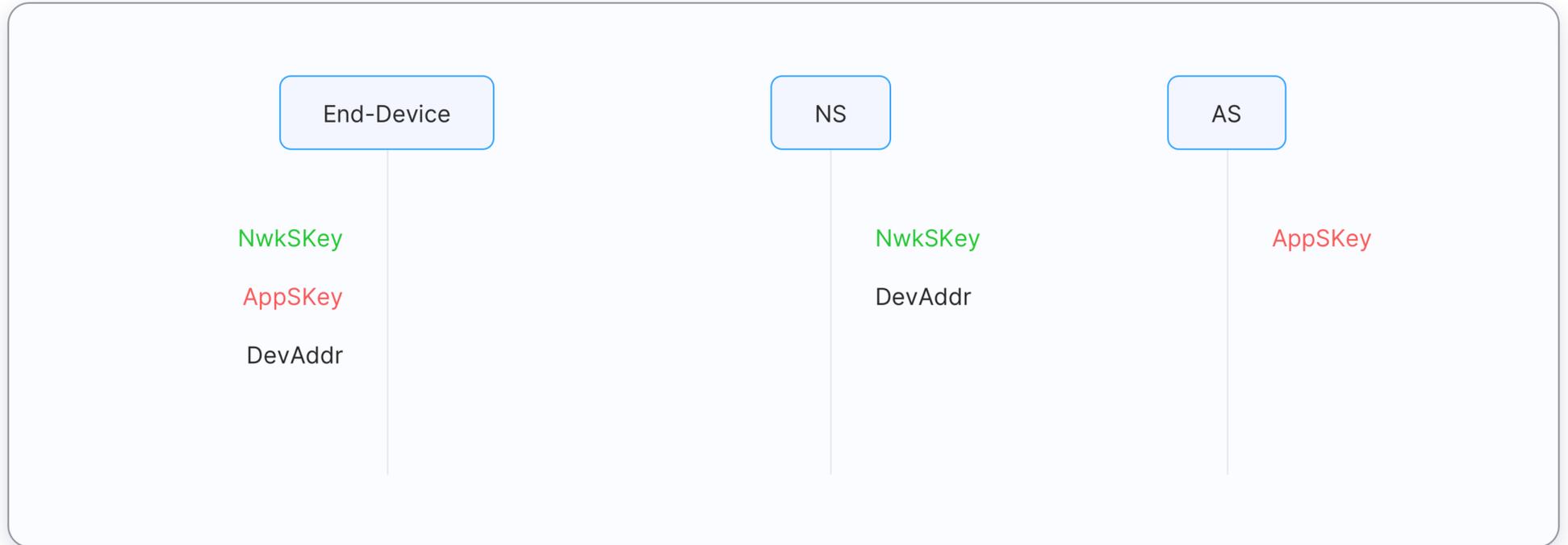
ABP – Activation By Personalization

- Bei der Aktivierung durch Personalisierung (ABP) wird ein End-Device direkt an ein vorab ausgewähltes Netzwerk gebunden, wobei das drahtlose Aktivierungsverfahren umgangen wird. Die Aktivierung durch Personalisierung ist die weniger sichere Aktivierungsmethode und hat außerdem den Nachteil, dass Geräte den Network-Provider nicht wechseln können, ohne die Schlüssel im Gerät manuell zu ändern. Ein Join-Server ist am ABP-Prozess nicht beteiligt.
- Ein mit der ABP-Methode aktiviertes End-Device kann nur mit einem einzigen Netzwerk arbeiten und behält während seiner gesamten Lebensdauer dieselbe Sicherheitssitzung bei.

ABP in LoRaWAN 1.0.x

- Die **DevAddr** und die zwei session keys **NwkSKey** und **AppSKey** sind anstatt **DevEUI**, **AppEUI** und der **AppKey** sind im End-Device dauerhaft gespeichert. Jedes End-Device muss einen eindeutigen Satz **NwkSKey** und **AppSKey** haben. Dieselbe **DevAddr** und derselbe **NwkSKey** müssen dem 'Network Server' bekannt sein. Der **AppSKey** muss im 'Application Server' gespeichert sein.

ABP in LoRaWAN 1.0.x



Quelle: [2]

ABP in LoRaWAN 1.1

- Anstatt **DevEUI**, **JoinEUI**, **AppKey** und **NwkKey** müssen die **DevAddr** und die vier 'session keys' **FNwkSIntKey**, **SNwkSIntKey**, **NwkSEncKey** und **AppSKey** im 'end device' gespeichert sein.
- Die selben **DevAddr**, **FNwkSIntKey**, **SNwkSIntKey** und **NwkSEncKey** müssen auch im 'Network Server' gespeichert sein. Der **AppSKey** muss im 'Application Server' gespeichert sein.

ABP in LoRaWAN 1.1



Quelle: [2]

Network Server Konfiguration

- Damit ein LoRaWAN Netzwerk funktioniert, muss der Network Server konfiguriert werden.
- Die notwendigen Schritte sind auf den folgenden Seiten beschrieben.
 - Konfiguration eines evtl. vorhandenen Gateway
 - Aufnahme eines End-Device ins LoRaWAN Netzwerk

The Things Network

The screenshot displays the 'Home > Dashboard' page of The Things Stack. The interface includes a left-hand navigation sidebar with sections for 'Home', 'Applications', and 'Gateways'. The main content area is divided into several widgets:

- Active applications**, **Connected gateways**, and **Total end devices**: Each widget features a 'Unlock the Network Operations Center' message, a description, an 'Upgrade now' button, and filters for '7 days' and '30 days'.
- Top entities**: A table with tabs for 'All', 'Applications', 'Gateways', and 'End devices'. It lists 'Luga-Application' with a 'No recent activity' indicator.
- Notifications**: A section with a 'View all' link and a message: 'No notifications yet. Your latest notifications will appear here.'
- Documentation**: A list of links including 'Getting started', 'End devices', 'Gateways', 'Integrations', and 'The Things Stack'.
- Quick actions**: A grid of buttons for 'New application', 'Register end device in an application', 'New organization', 'New personal API key', and 'New gateway'.

At the bottom left of the dashboard, there is a 'Resources' dropdown and the version number 'v3.33.0.cc@020807'.

The Things Network

- Ein Praxisbeispiel mit ‚The Things Network‘ als Network-Server.
- Konfiguration eines Gateway für TTN
- Registrieren eines Gateway
- Registrieren eines End-Device

Registrieren eines Gateway

 LoRa ▾ LoRaWAN ▾ Forwarder ▾ Network ▾ System ▾ Server ▾ LogRead ▾ Home Logout

LoRa Configuration

Debug Level

Radio Settings

Stat Package Period (sec)

Frequency Plan

Static GPS coordinates ?

Enable Static GPS

Latitude

Altitude (m)

Longitude

Current Mode: **LoRaWAN Semtech UDP**

Konfiguration eines Gateway

DRAGINO LoRa LoRaWAN Forwarder Network System Server LogRead Home Logout

LoRaWAN Configuration

General Settings

Email

Gateway EUI

Primary LoRaWAN Server

Service Provider Server Address

Uplink Port Downlink Port

Secondary LoRaWAN Server

Service Provider

Packet Filter

Primary server Fport Filter ? DevAddr Filter ?

Secondary server Fport Filter DevAddr Filter

Add Filter

Server Name: Filter type: Filter Value

DELET Filter

Current Mode: **LoRaWAN Semtech UDP**

Registrieren des Gateway

THE THINGS STACK

Gateways > Register gateway

Home Applications Gateways

Search Ctrl K

Register gateway

Register your gateway to enable data traffic between nearby end devices and the network.
Learn more in our guide on [Adding Gateways](#).

Does your gateway have a LoRaWAN® Gateway Identification QR Code? Scan it to speed up onboarding.

Scan gateway QR code

Gateway EUI [ⓘ]

AA BB CC DD EE FF 00 11 Reset

Gateway ID [ⓘ] *

luga-gateway

Gateway name [ⓘ]

LUGA Gateway

Frequency plan [ⓘ] *

Europe 863-870 MHz (SF9 for RX2 - recommended) | v

+ Add frequency plan

Note: most gateways use a single frequency plan. Some 16 and 64 channel gateways however allow setting multiple within the same band.

Require authenticated connection [ⓘ]
Choose this option eg. if your gateway is powered by [LoRa Basic Station](#)

Share gateway information
Select which information can be seen by other network participants, including [Packet Broker](#)

Share status within network [ⓘ]

Share location within network [ⓘ]

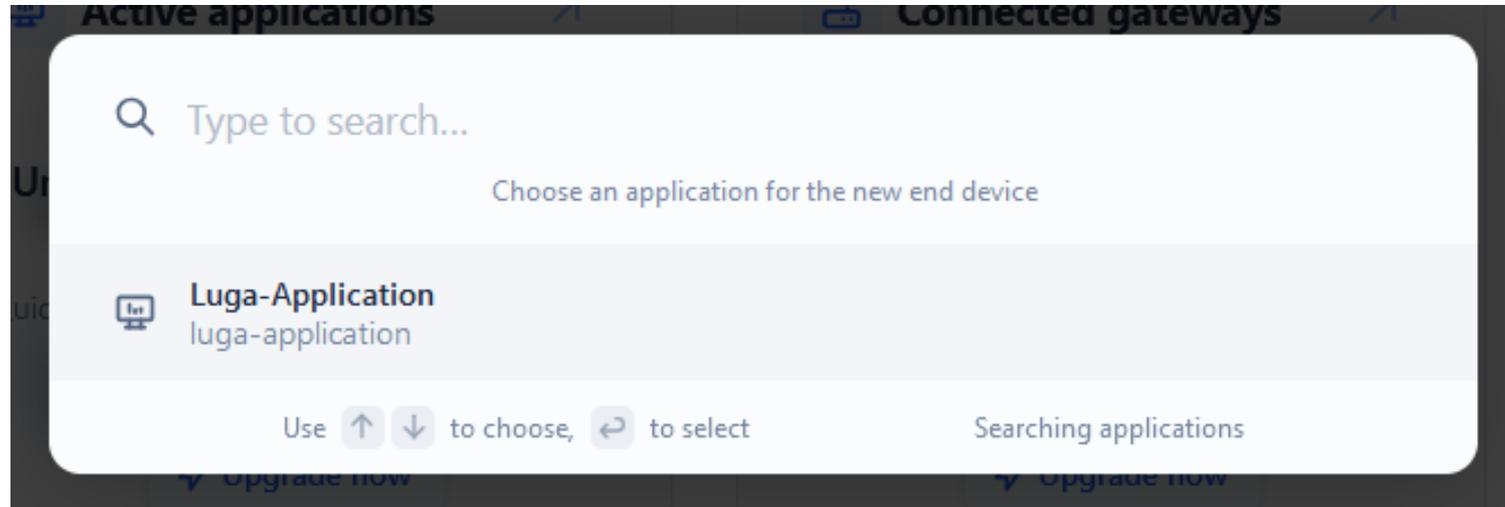
Register gateway

Das Gateway ist registriert

The screenshot displays the 'LUGA Gateway' overview page in The Things Stack. The interface includes a left sidebar with navigation options like 'Home', 'Applications', and 'Gateways'. The main content area is divided into several sections:

- General information:** Gateway ID (luga-gateway), Gateway EUI (AA BB CC DD EE FF 00 11), Frequency plan (Europe 863-870 MHz), and Created at (Jan 23, 2025 11:38:44).
- Network settings:** A list of settings with toggle switches: Require authenticated connection (Disabled), Public status (Enabled), Public location (Enabled), Packet Broker forwarding (Enabled), Status location updates (Disabled), and Enforce duty cycle (Enabled).
- Gateway status:** Shows 'Disconnected' and a message: 'This gateway has not made any connection attempts yet.' It includes a note about waiting for connection and a link to 'gateway troubleshooting documentation'.
- Network activity:** A section for monitoring activity, currently showing 'Packets per data rate' and 'Packets per channel (24 days)'.
- Location:** A section for the gateway's location, currently showing a 'Map' view.

Register end Device in an application



Register End Device

The screenshot shows the 'Register end device' page in The Things Stack. The page is titled 'Register end device' and includes a breadcrumb trail: 'Applications > Luga-Application > End devices > Register end device'. The main content area contains the following sections:

- Header:** 'Register end device' title, a search bar, and navigation tabs for 'Home', 'Applications', and 'Gateways'.
- Left Sidebar:** A navigation menu for 'Luga-Application' with options: 'Application overview', 'End devices' (selected), 'Live data', 'Payload formatters', 'Integrations', 'Collaborators', 'API keys', and 'General settings'. A note at the bottom of the sidebar states 'No top end devices yet'.
- Main Content:**
 - Introduction:** A message asking if the user has a LoRaWAN Device Identification QR Code and providing a 'Scan end device QR code' button and a 'Device registration help' link.
 - End device type:** A section with an 'Input method' dropdown set to 'Enter end device specifics manually'. Below it are three dropdown menus: 'Frequency plan' (Europe 863-870 MHz (SF9 for RX2 - recommended)), 'LoRaWAN version' (LoRaWAN Specification 1.0.2), and 'Regional Parameters version' (RP001 Regional Parameters 1.0.2).
 - Provisioning information:** A section with a 'JoinEUI' field containing '00 00 00 00 00 00 00 00' and a 'Reset' button. Below it is the text 'This end device can be registered on the network'. The 'DevEUI' field contains '02 00 00 00 00 00 00 01'. The 'AppKey' field contains '00 C1 7C E8 F1 C9 33 C8 CC 9C 12 34 31 41 E7 74' and a 'Generate' button. The 'End device ID' field contains 'ebyte-test-module'.
 - After registration:** A section with two radio buttons: 'View registered end device' (selected) and 'Register another end device of this type'.
 - Footer:** A blue 'Register end device' button.

ChirpStack

- Alternativ kann der Chirpstack als Network-Server dienen.
- In diesem Beispiel wurde eine Dockerinstanz auf einem Raspberry Pi installiert.
 - 1) Klonen Github Repository:
`git clone https://github.com/chirpstack/chirpstack-docker.git`
 - 2) Cd ./chirpstack-docker
 - 3) Container starten:
`docker compose up -d`
 - 4) Wenn der Stack läuft (`docker compose ps`, `docker compose logs`), mit dem Chirpstack verbinden:
`https://192.168.178.xx:8080`
 - 5) Mit User: ‚admin‘ und Passwort: ‚admin‘ anmelden.

ChirpStack

ChirpStack login

* Username / email:

* Password: 

ChirpStack – Add Gateway

ChirpStack

Search... ? admin

tenant

Tenants / tenant / Gateways

Gateways Add gateway Selected gateways

	Last seen	Gateway ID	Name	Region ID	Region common-name
No data					

Version: v4.11.0

ChirpStack – Add Gateway

The screenshot shows the ChirpStack web interface for adding a new gateway. The page title is "Add gateway" and the breadcrumb is "Tenants / tenant / Gateways / Add". The form is divided into three tabs: "General", "Tags", and "Metadata".

- General Tab:**
 - * Name:** A text input field containing "Dragino-GW".
 - Description:** A text area containing "Dragino LPS8v2 Gateway".
 - * Gateway ID (EUI64):** A text input field with a dropdown menu set to "MSB", a refresh icon, and a trash icon.
 - * Stats interval (secs):** A numeric input field set to "30".
 - Location:** A large blue area with a small white box containing a "+" and "-" sign, indicating a map or location selection interface.

The left sidebar contains navigation options for "Network Server" (Dashboard, Tenants, Users, API Keys, Device Profile Templates, Regions) and "Tenant" (Dashboard, Users, API Keys, Device Profiles, Gateways, Gateway Mesh, Applications). The "Gateways" option is currently selected.

ChirpStack – Add End-Device

- Damit ein End-Device im ChirpStack registriert werden kann, müssen folgende Schritte durchgeführt werden.
 - Add device profile
 - Add application
 - Add device mit Auswahl eines device profile

ChirpStack – Add device profile

The screenshot displays the ChirpStack web interface for adding a device profile. The page title is "Add device profile" and the breadcrumb is "Tenants / tenant / Device profiles / Add". The left sidebar shows the navigation menu with "Device Profiles" selected. The main content area contains the following form fields:

- Name:** EByte
- Description:** EByte Development Module E78-868LN225
- Region:** EU868 (Region configuration: EU868)
- MAC version:** LoRaWAN 1.0.3 (Regional parameters revision: RP002-1.0.2)
- ADR algorithm:** Default ADR algorithm (LoRa only)
- Flush queue on activate:**
- Allow roaming:**
- Expected uplink interval (secs):** 900
- Device-status request frequency (req/day):** 1
- RX1 Delay (0 = use system default):** 0

A "Submit" button is located at the bottom of the form. A "Select device-profile template" button is also visible in the top right of the form area.

ChirpStack – Add device profile

The screenshot displays the ChirpStack web interface. At the top left is the ChirpStack logo. A search bar and a user profile dropdown (admin) are at the top right. A left sidebar contains navigation menus for 'tenant' and 'Tenant', with 'Device Profiles' selected. The main content area shows the breadcrumb 'Tenants / tenant / Device profiles / EByte'. Below this, the profile name 'EByte' and its ID are shown, along with a 'Delete device profile' button. A tabbed interface includes 'General', 'Join (OTAA / ABP)', 'Class-B', 'Class-C', 'Codec', 'Relay', 'Tags', and 'Measurements'. The 'Join (OTAA / ABP)' tab is active, showing a 'Device supports OTAA' toggle switch (turned on) and a 'Submit' button. A 'Select device-profile template' button is also visible.

ChirpStack – Add application

The screenshot displays the ChirpStack web application interface. At the top left is the ChirpStack logo. To its right is a search bar with the placeholder text "Search...". Further right is a user profile dropdown menu showing "admin" with a downward arrow. Below the search bar is a breadcrumb navigation path: "Tenants / tenant / Applications".

The main content area is titled "Applications" and features a blue "Add application" button in the top right corner. Below the title is a table with two columns: "Name" and "Description". The table is currently empty, and a "No data" message with a folder icon is centered in the table area.

The left sidebar contains a navigation menu. The "tenant" dropdown is set to "tenant". Under "Network Server", there are links for "Dashboard", "Tenants", "Users", "API Keys", "Device Profile Templates", and "Regions". Under "Tenant", there are links for "Dashboard", "Users", "API Keys", "Device Profiles", "Gateways", "Gateway Mesh", and "Applications", which is currently selected and highlighted in blue.

ChirpStack – Add application

The screenshot displays the ChirpStack web interface. At the top left is the ChirpStack logo. To its right is a search bar with the text "Search...", a magnifying glass icon, a help icon, and a user profile icon labeled "admin". Below the search bar is a breadcrumb trail: "Tenants / tenant / Applications / Add". The main heading is "Add application". There are two tabs: "General" (active) and "Tags". The form contains a required field for "Name" with the value "LUGA Application" and a "Description" field with the value "LUGA Application". A blue "Submit" button is located at the bottom of the form. On the left side, there is a navigation menu with a "tenant" dropdown at the top. The menu items are: Network Server, Dashboard, Tenants, Users, API Keys, Device Profile Templates, Regions, Tenant (selected), Dashboard, Users, API Keys, Device Profiles, Gateways, Gateway Mesh, and Applications (highlighted in blue).

ChirpStack – Add application

ChirpStack

Search...

admin

tenant

Tenants / tenant / Applications / LUGA Application

LUGA Application application id: 89654919-8c15-4a92-ac8d-64d331efee4d [Delete application](#)

[Devices](#) [Multicast groups](#) [Relays](#) [Application configuration](#) [Integrations](#)

[Add device](#) Selected devices

<input type="checkbox"/>	Last seen	DevEUI	Name	Device profile	Battery
No data					

ChirpStack – Add device

The screenshot shows the ChirpStack web interface for adding a new device. The page title is "Add device" and the breadcrumb trail is "Tenants / tenant / Applications / LUGA Application / Add device". The left sidebar contains navigation menus for "Network Server" (Dashboard, Tenants, Users, API Keys, Device Profile Templates, Regions) and "Tenant" (Dashboard, Users, API Keys, Device Profiles, Gateways, Gateway Mesh, Applications). The main form has three tabs: "Device" (selected), "Tags", and "Variables". The form fields include: a required "Name" field with the value "EByte E78-868LN22S"; a "Description" field with the value "EByte E78-868LN22S Development Module"; a "Device EUI (EUI64)" field with the value "0200000000000001" and a "Join EUI (EUI64)" field with the value "0000000000000000"; a "Device profile" dropdown menu with the value "EByte-Profile"; a "Device is disabled" toggle switch (currently off); and a "Disable frame-counter validation" toggle switch (currently off). A blue "Submit" button is located at the bottom left of the form.

tenant

Network Server

Dashboard

Tenants

Users

API Keys

Device Profile Templates

Regions

Tenant

Dashboard

Users

API Keys

Device Profiles

Gateways

Gateway Mesh

Applications

Tenants / tenant / Applications / LUGA Application / Devices / EByte E78-868LN22S

EByte E78-868LN22S device eui: 0200000000000001

Delete device

Dashboard Configuration OTAA keys Activation Queue Events LoRaWAN frames

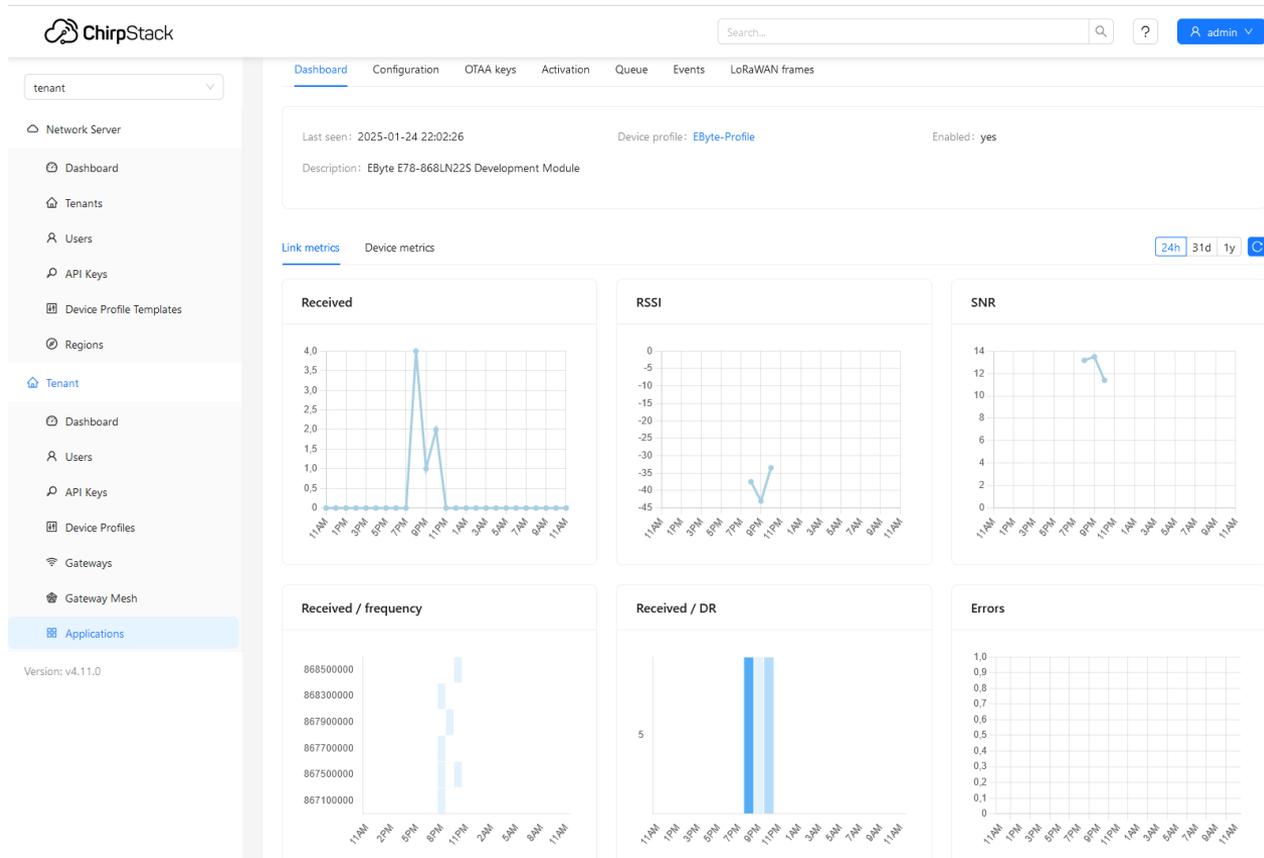
* Application key

B0C17CE8F1C933CBCC9C12343141E774

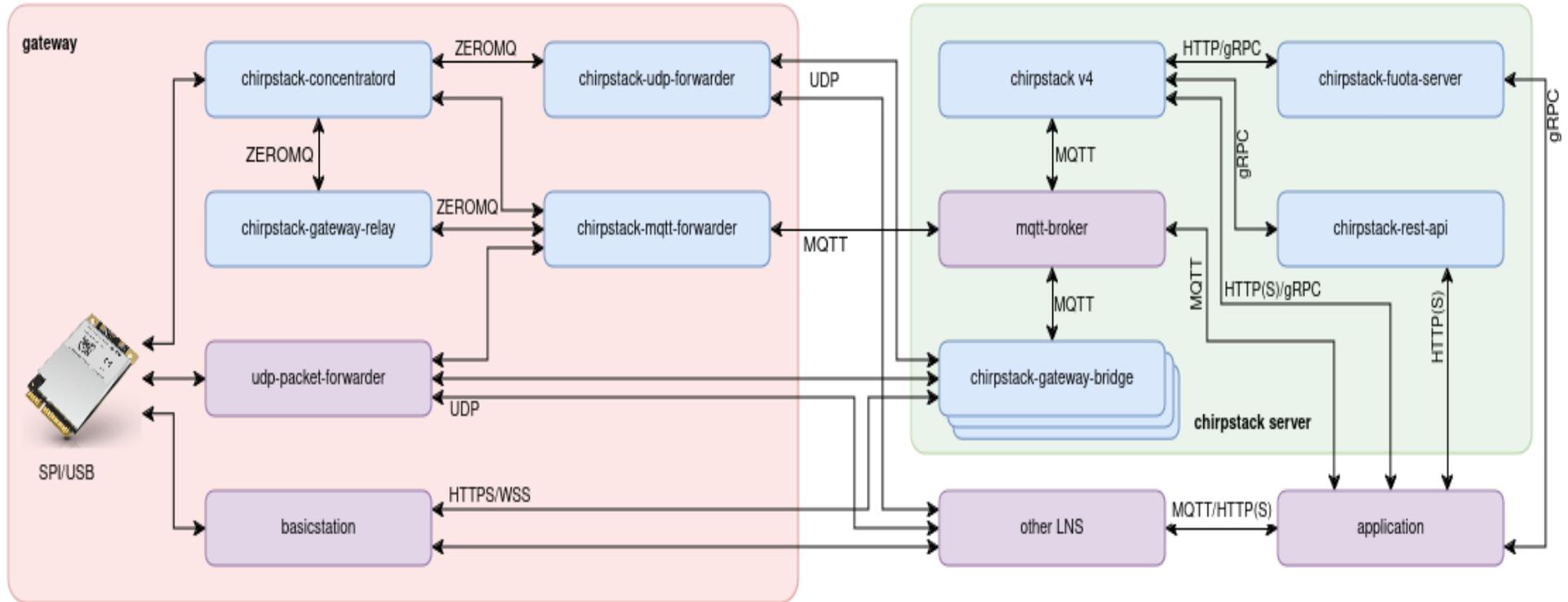
MSB C

Submit

ChirpStack – Application link metrics



Chirpstack Architecture



Network Server for Self Hosting

- The Things Stack, an open source LoRaWAN Network Server
<https://github.com/TheThingsNetwork/lorawan-stack>
- ChirpStack open-source LoRaWAN(R) Network Server
<https://github.com/chirpstack/chirpstack>
<https://github.com/chirpstack/chirpstack-docker>
- LoRaWAN 1.0.4 Specification in Depth For End Device Developers
<https://learn.semtech.com/course/view.php?id=6>
- LoRaWAN airtime calculator
<https://www.thethingsnetwork.org/airtime-calculator>

- [1] <https://blog.semtech.com/>
- [2] <https://www.thethingsnetwork.org/docs/lorawan>
- [3] https://media.elv.com/file/251814_lpwan.pdf
- [4] <https://www2.htw-dresden.de/~jvogt/projektseminar/public/>
- [5] <https://learn.semtech.com/course/view.php?id=6>

Become a LoRaWAN expert

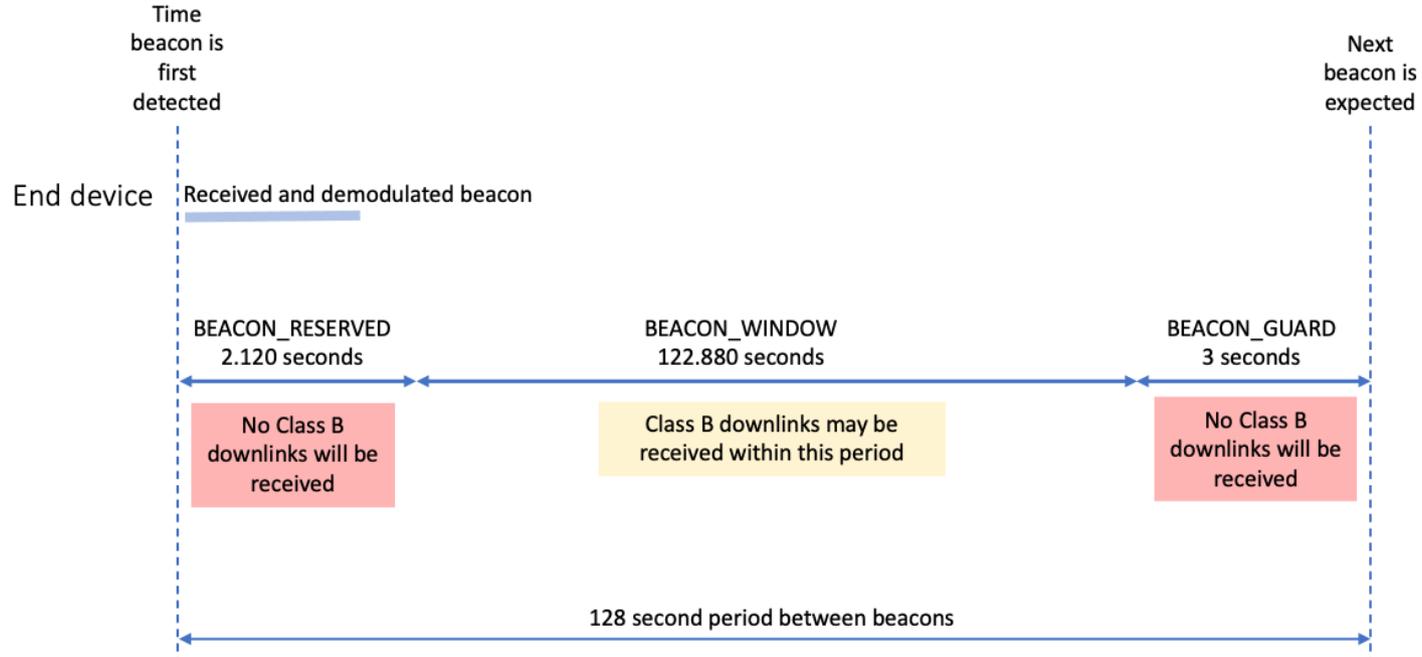
- <https://www.univ-smb.fr/lorawan/en/>

Quelle: [4]

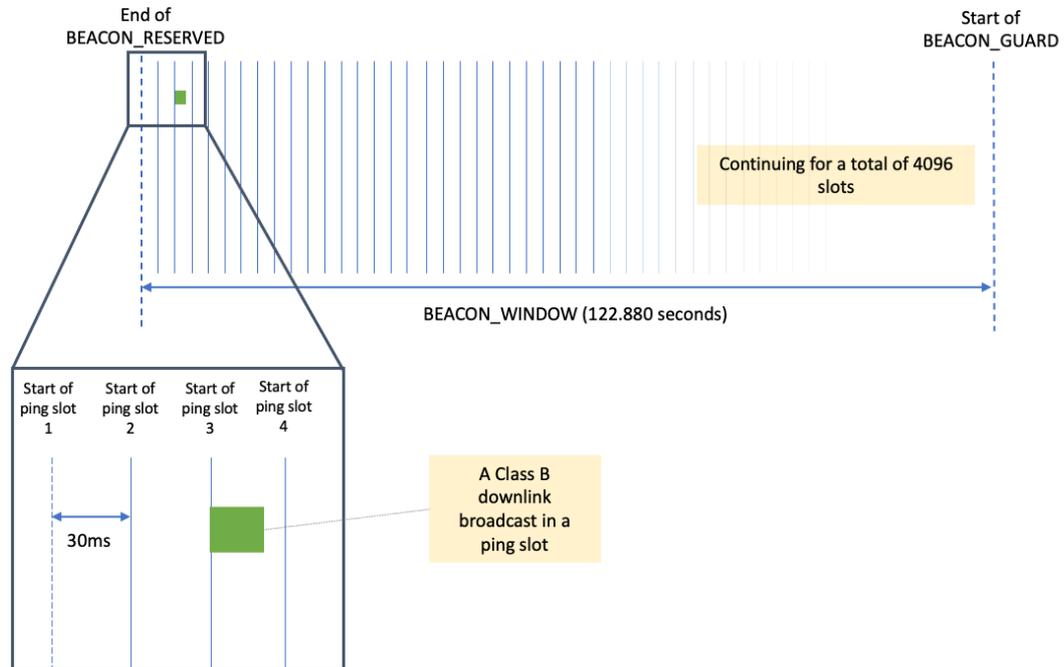
Glossar

- ECB — AES electronic codebook mode encryption
- ISM Band - Industrial, Scientific and Medical Band

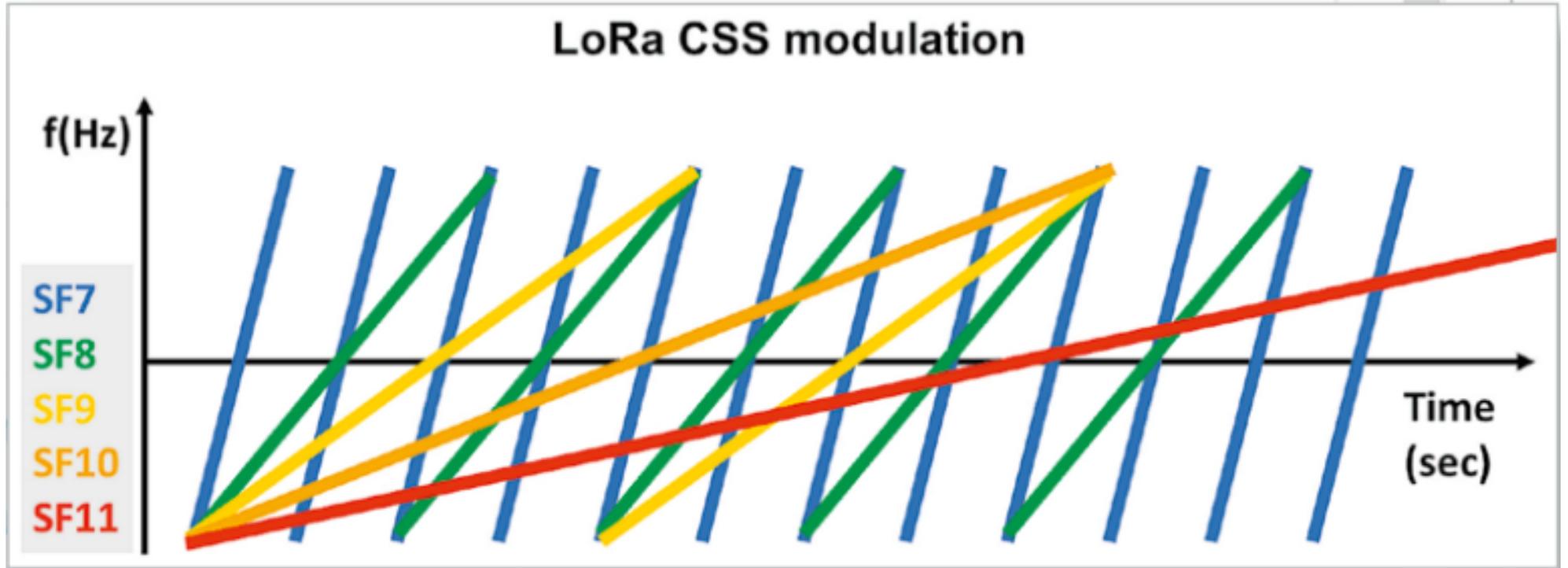
Anhang



Anhang

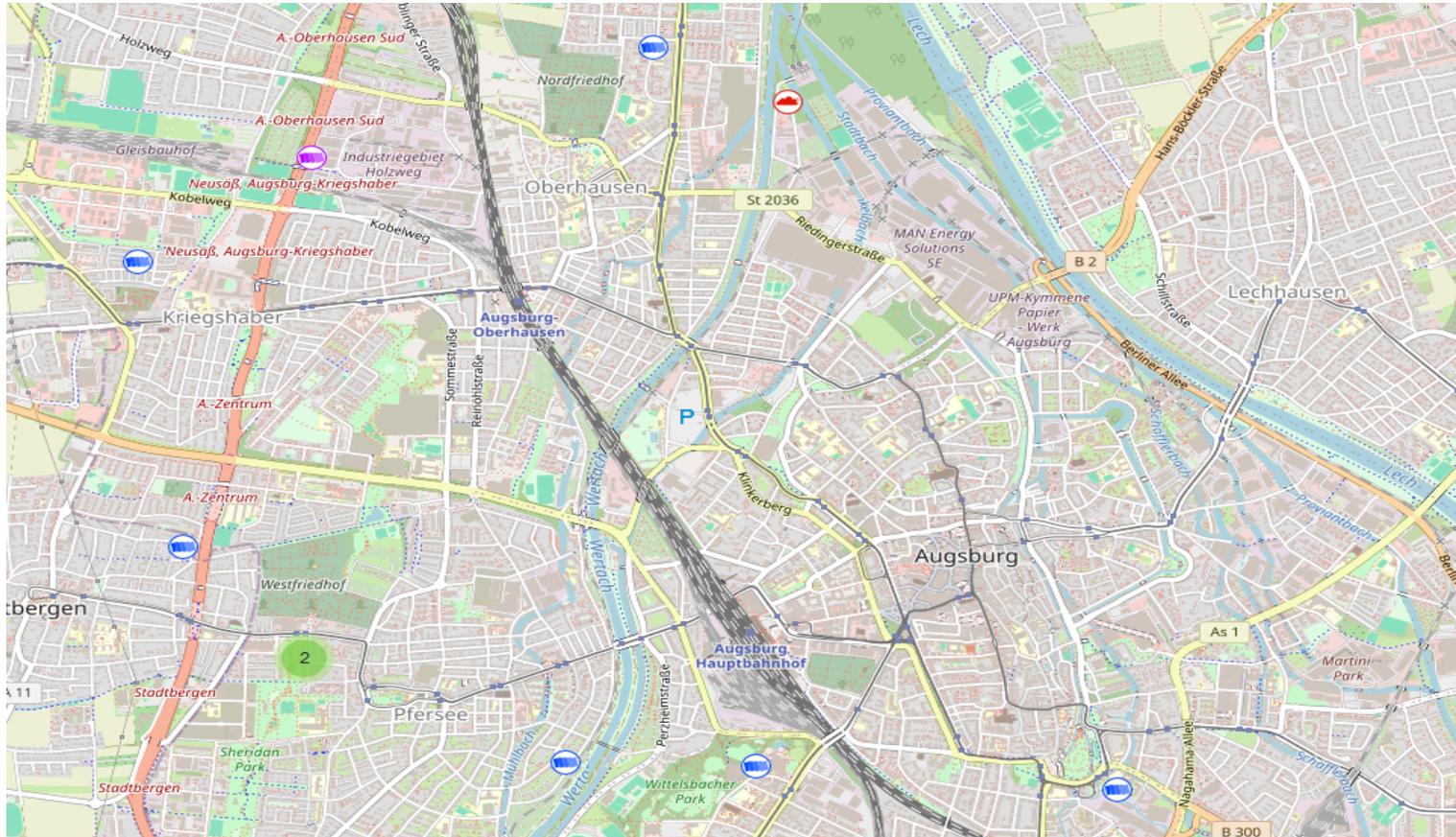


LoRa CSS Modulation



Quelle: [3]

Gateway Map Augsburg



Beispiele LoRaWAN Devices



Dragino Gateway LPS8N

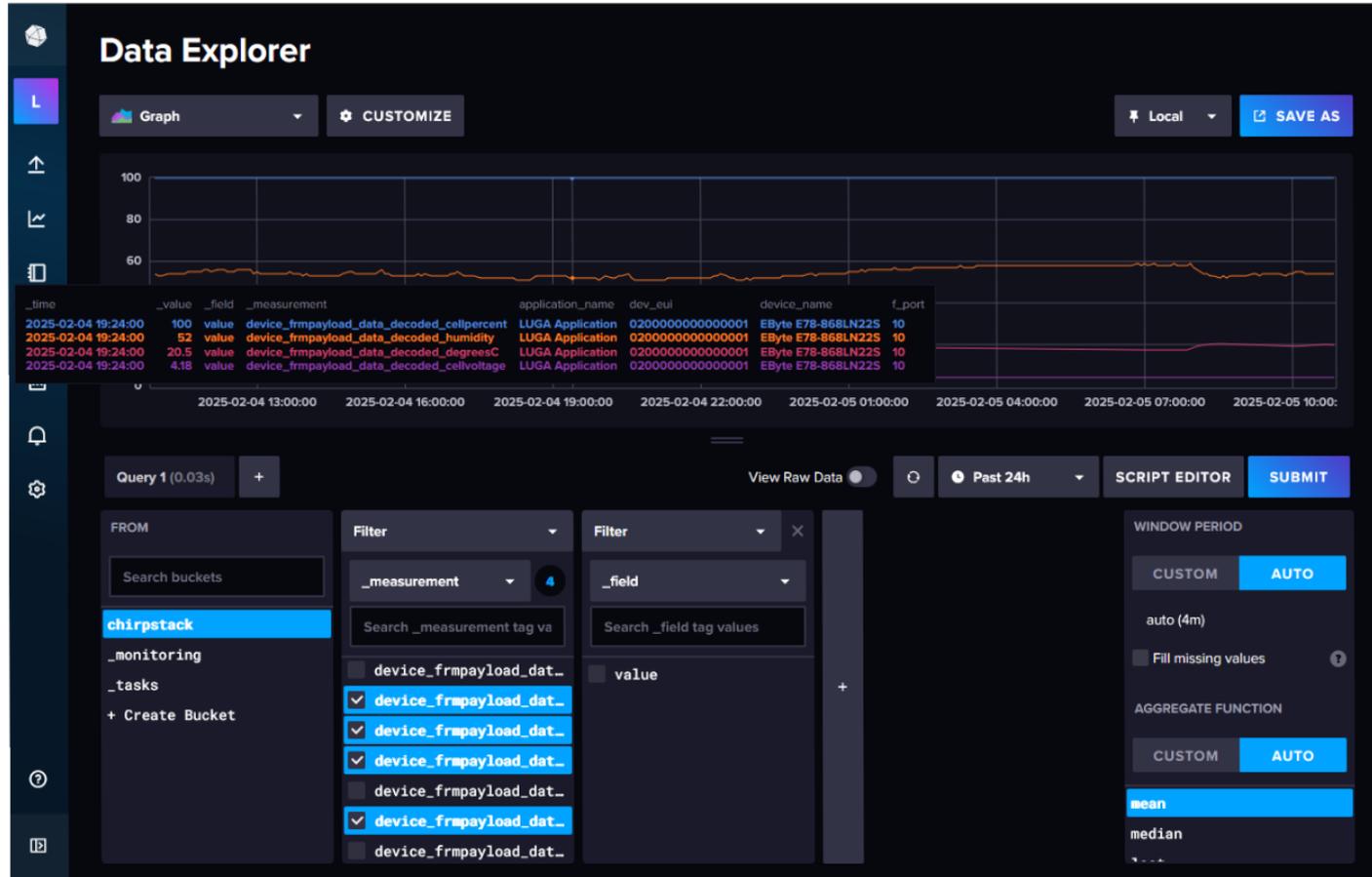


Dragino Node LHT65N – Temp, Hum

LoRaWAN Gateway Beispiel-Installation



Influxdb Data Explorer



Grafana Dashboard

