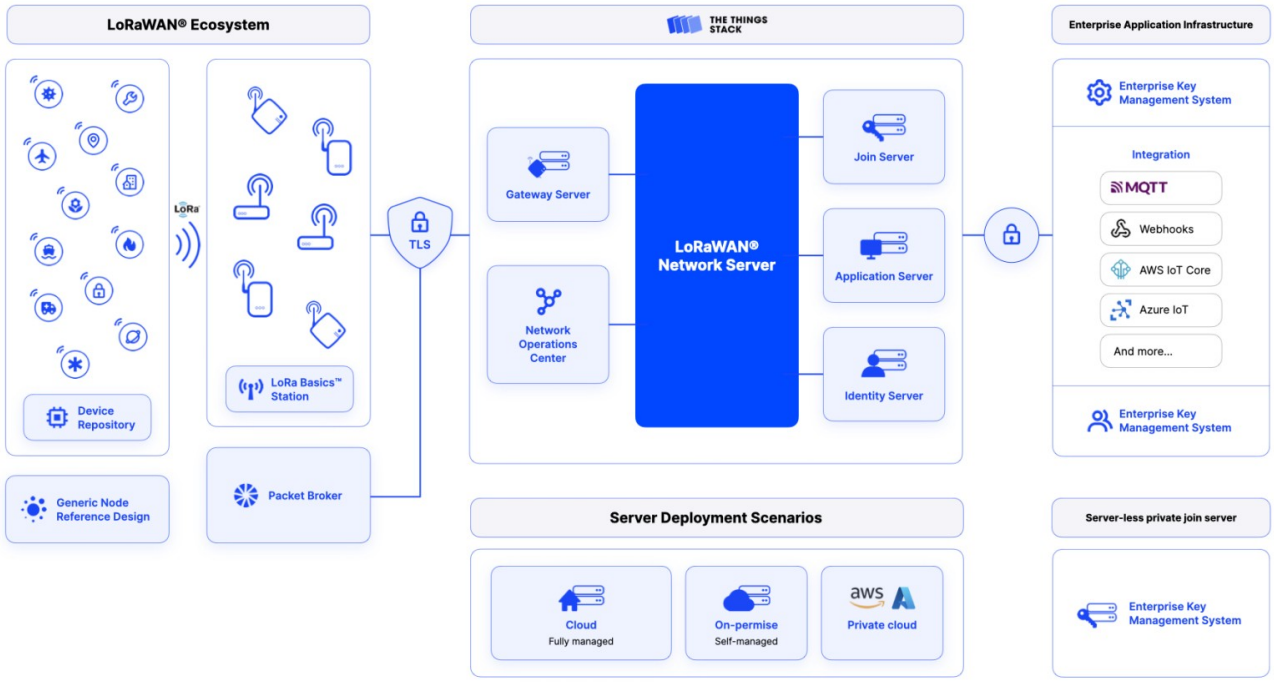
LoRa & TTN

Diese Seite beschreibt den Einsatz von LoRa und The Things Network (TTN), um Daten von Geräten einzusammeln, um sie anschließend in eine Datenbank abzulegen, und sie final zu visualisieren.

Die gesamte Kette der Datenübertragung wir im folgenden Bild zusammengefasst, Details finden sich in den weiteren Abschnitten



LoRa Datenübertragung

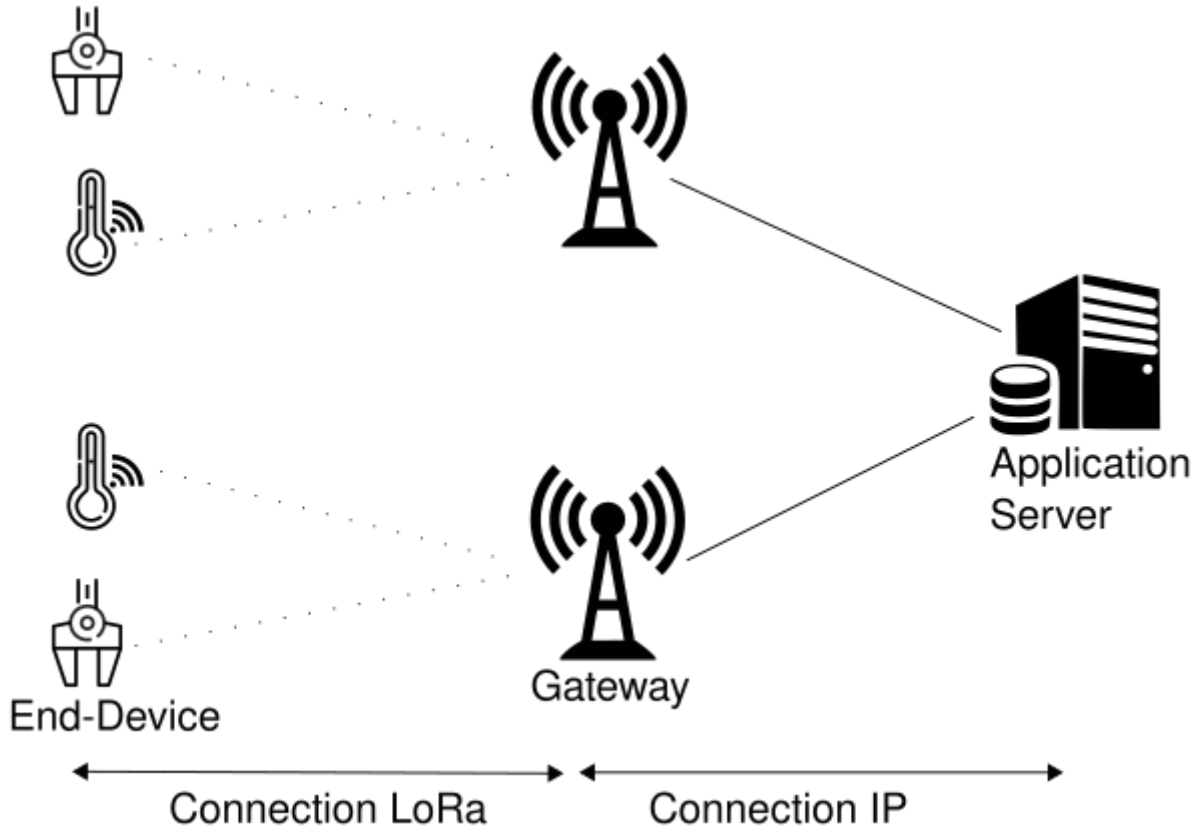
Im IoT Bereich hat sich das Übertragungsprotokoll LoRa (Long Range) etabliert.

Es funkt im freien Frequenzband von 433 bzw 866MHz, und ermöglicht Datenübertragung über Entfernungen von 2km innerorts bis über 40km im Freien.

Es werden allerdings nur wenige Bits übertragen, wodurch sich das Protokoll bestens zum Übermitteln von Status-Informationen, Sensor-Meßwerte... eignen. Das Protokoll ist prinzipiell bidirektional, sodass man damit auch Befehle an die Geräte senden kann.

Als Gegenstelle zum IoT Gerät mit seinem LoRa Modul (z.B. ESP TTGO) dient ein LoRa Gateway, welches typischerweise eine Brücke in's Internet aufbaut (per LAN Kabel, WLAN oder Mobilfunk),

und weiter zu einem Server, der die Daten weiter verarbeitet.



Diese Gateways kosten zwischen 100€ und 800€, weshalb es sinnvoll ist, die Gateways zu teilen. Damit erhöhnt sich die Abdeckung, bzw die Erreichbarkeit der Geräte.

The Things Network

Das Projekt The Things Network (TTN) verwaltet ein solches, weltweites, Netzwerk von Gateways. Man kann das Netzwerk kostenlos verwenden, aber auch eigene Gateways dem Netzwerk hinzufügen, was wir im Oberlab auch tun.

TTN Logo (Wikipedia)

Auch wenn alle Geräte Daten über die Gateways senden können, sind diese mit einem eigenen AES -Schlüssel gesichert, sodass nur der Eigentümer des Geräts die Daten im Klartext sehen kann. Andere Teilnehmer, sowie ebenfalls der Betreiber des Gateways, über den die Datenpakete laufen, sehen nur die verschlüsselte Kommunikation. Die Entschlüsselung erfolgt auf den TTN-Server, wo der Eigentümer die Daten abholen kann.

Es gibt verschiedene Übersichtskarten

- Die offizielle Karte der Gateways von TTN
- TTN Mapper Heatmap mit den Gateways und den Signalraten
- TTN Mapper Radar mit den Paketübetragungen, zeigt die Verbindungen zwischen Geräten und Gateways

Datenpakete

Um die übertragene Datenmenge weiter zu reduzieren, werden die Bits auf das Minimum reduziert, also zB 8 Bit um einen Integer-Zählerwert von 0 bis 255 abzubilden. Der TTN-Server konvertiert diese roh-Bits in eine Datenstruktur, mithilfe einer in Javascript geschriebenen Dekoder-Funktion (es gibt auch eine Enkoder-Funktion wenn man struturierte Daten in Bits wandeln will, um sie zum IoT Gerät zu versenden), zu finden unter “Payload Formatter”.

Die Funktion weist z.B. dem Feld “anzahl_wifi_geräte” den Wert aus dem 5. Byte des Datenpakets zu, bzw dem Feld “batterie_spannung” der Wert aus den 7. und 8. Bytes zu (zuerst nach Ganzzahl gewandelt, dann durch 1000 geteilt - von mV nach V umgerechnet). Die Felder werden unter “decoded_payload” in der JSON Datenstruktur abgelegt, siehe Data Formats Doku:

```
"decoded_payload": {
    "battery": 3754,
    "ble": 8,
    "ble_new": 8,
    "wifi": 1,
    "wifi_new": 0
  },
```


Applikationen

Alle Oberlab Projekte werden in der TTN Console über den Oberlab-Account verwaltet. TTN strukturiert die unterschiedlichen Projekte in “Applications”, z.B. Connecting Peaks. Innerhalb einer Applikation werden Geräte angemeldet - deren Datenströme sind dann im Kontext der Applikation abrufbar. Pro Applikation legt man ebenfalls eine Decoder- und Encoder Funktion an.

In der Console kann man die Live-Daten einsehen, die von TTN empfangen werden. Bei jedem Paket sind sowohl die roh-Daten als auch die dekodierten, strukturierten Daten sichtbar. Damit kann man prüfen, ob der Dekoder richtig funktioniert. Falls am Ende der Verarbeitungskette keine Daten in der Visualisierung (z.B. Grafana) ankommen, mach es Sinn den Datenverkehr live anzusehen, um zu verstehen wo die Kommunikation gestört ist.

Geräte

Geräte werden über ihre DevEUI identifiziert - eine *eindeutige* 16-stellige Hex-Zahl die man selber vergibt wenn man die Geräte einrichtet. Sie melden sich beim TTN an, indem sie die JoinEUI verwenden, ebenfalls eine 16-stellige Hex-Zahl die man pro Applikation angibt. Diese IDs müssen im Gerät bzw in der Firmware einprogrammiert werden. Passen sie nicht, ist keine Anmeldung im TTN-Netzwerk möglich. Es empfiehlt sich daher, die verschiedenen IDs im Projekt zu dokumentieren, eine nachträgliche Änderung geht nicht, man muss dazu das Gerät bei TTN löschen und wieder einrichten! Rechner zum erstellen von EUIs

Die End device ID ist der Name vom Gerät, wie er später in den Datenstrukturen sichtbar sein wird (z.B. *miesbach-pxc-01*). Die ID kann ebenfalls nur beim Einrichten gesetzt werden, später nicht mehr. Der Feld *Name* wird nur für die Auflistung in der TTN Console verwendet, in den Datenstrukturen taucht er nicht auf. Mehr Details in der TTN Doku zum Anmelden von Geräten.

Zugriff auf die Daten

Die empfangenen Daten können dann bei TTN über verschiedene Integrationen abgeholt werden, z.B.

- deren MQTT Server (TTN Doku zu MQTT)
- Node Red
- IFTTT
- Webhooks (z.B. Cayenne)

Bei den Oberlab-Projekten werden die TTN-Daten von deren MQTT Server auf den Oberlab MQTT Server (mqtt.oberlab.de) gespiegelt, und zur weiteren Verarbeitung zur Verfügung gestellt. z.B. können die MQTT Daten mittels Telegraf in eine InfluxDB Datenbank geschrieben werden, bevor sie mit Grafana visuell in Dashboards dargestellt werden.

Zur Integration mit den Integrationen sind API Keys notwendig, sie werden pro Applikation erstellt. Idealerweise sollten die Keys nicht mehrfach verwendet werden, sondern es sollte ein Key pro Applikation/Person/… erstellt werden. Keys können verschiedene Zugriffsrechte auf die Daten haben, je weniger Rechte desto sicherer!

Version #5
Erstellt: 14 Mai 2024 19:54:52 von Joel Hatsch
Zuletzt aktualisiert: 25 September 2024 21:33:19 von Joel Hatsch