

Seminararbeit: LoRaWAN

Tobias Sigmann

25. Mai 2019

Inhaltsverzeichnis

1	Einführung in Lora	3
2	Aufbau eines Lora-Netzwerk	3
2.1	Gateway	3
2.2	Netzwerkserver	4
2.3	Join-Server	4
2.4	End-Gerät	5
3	LoRaWAN Funktionsweise	5
3.1	Schichtenmodell	5
3.2	Netzwerkbeitritt	6
3.2.1	OTAA	6
3.2.2	ABP	8
3.3	Protokoll	8
3.3.1	MAC-Kommandos	9
3.3.2	LoRa-Paketstruktur	10
3.4	Übertragungsart	13
3.4.1	Adaptive Data Rate	14
4	Lora Geräte Klassen	15
4.1	Klasse A	15
4.2	Klasse B	16
4.2.1	Klassenwechsel A nach B	16
4.2.2	Betrieb	17
4.2.3	Singel / Multicast	18
4.2.4	Beacon	18
4.3	Klasse C	19
5	Sicherheit	20
5.0.1	Schlüssel	20
5.0.2	Zähler	21

1 Einführung in Lora

2 Aufbau eines Lora-Netzwerk

LoRa ist ideal für IoT-Geräte, da per einfachem Netzwerkaufbau Datenaustausch mit dem Internet möglich ist. Um die von den End-Geräten gesendeten LoRa-Pakete auf IP/TCP Pakete umzusetzen wird ein Gateway als Schnittstelle benötigt, um die LoRa-Pakete in TCP/IP Pakete umzuwandeln und umgekehrt. Das Gateway implementiert aber keinerlei Logik. Hierzu ist ein Netzwerkservers zuständig der über die Gateways das Netzwerk kontrolliert und steuert. Gleichzeitig stellt er die Verbindung zu einem Applikationsserver her, in dem er die Daten weiterleitet.

Der Applikationsserver ist zuständig die gesendeten Nachrichten zu verarbeiten und sendet Daten an die Endgeräte.

Diese Architektur wurde gewählt um geringen Energieverbrauch zu ermöglichen, bei gleichzeitiger höher Endgeräteanzahl, hoher Signalqualität und entsprechender Netzwerksicherheit. [Tec15, S. 8 ff.]

2.1 Gateway

Die LoRa-Endgeräte werden sternförmig an die Gateways angeschlossen und stellen ein Teilnetz dar. Jedes Endgeräten kommuniziert direkt mit dem Gateway. Diese Art der Kommunikation wird “Single-Hop-Connection” genannt, da die gesendeten Daten ohne Umwege an das Gateway gesendet werden. Jedes Gateway ist mit mindestens einem Netzwerkservers verbunden.

Ein Endgerät kann gleichzeitig an mehreren Gateways senden. Der Netzwerkservers überprüft die Pakete auf Duplikate und stellt sicher, dass jedes Paket nur einmal an die Applikationsservers gesendet werden. Ein weiterer Vorteil ist das keine Endgeräteübergabe zwischen den Gateways bei Standortwechsel nötig sind. Da die Gateways mit allen in der Reichweite befindlichen Endgeräten kommunizieren müssen, wurde auf Einzelkommunikation verzichtet und auf Parallelkommunikation gesetzt. Dazu werden adaptive Datenraten und Mehrkanal-Multi-Modem-Transceiver verwendet.

Durch die genannt Eigenschaften der Gateways wird eine gute Skalierbarkeit erzielt.

Dadurch können neue Gateways die Anzahl der Endgeräte um das 6 bis 8-fach erhöhen. vgl. [Tec15, S.10]

2.2 Netzwerkservers

Der Netzwerkservers ist das “Herzstück“ eines jeden Lora-Netzwerkes. Er kann mit mehreren Gateways und mehreren Applikationsservers verbunden sein.

Die wichtigste Aufgabe des Netzwerkservers ist die Steuerung des LoRa-Netzwerkes. Der Servers verwaltet jedes Endgerät separat indem es mit ihm den zu verwendenden Funkkanal aushandelt und die Datenrate kontrolliert wenn ADR (Adaptiv Data Rate) aktiv ist. Außerdem steuert er den Netzwerkbeitritt der Endgeräte.

Weiterhin überprüft er die empfangen Pakete auf ihre Korrektheit, Integrität und wie schon erwähnt filtert er Duplikate. Dabei ermittelt er auch das Gateway mit dem besten Empfang zum jeweiligen Endgeräten und nutzt dieses dann um Daten an das Endgerät zu senden.

Es ist nicht immer möglich Daten direkt zu senden, da die Endgeräte nicht immer empfangsbereit sind. Um die Applikationsservers zu entlasten, puffert der Netzwerkservers die Daten und sendet diese zum nächst möglichen Zeitpunkt.

Eine weitere sehr wichtige Ausgabe ist es eine Schnittstelle für den Applikationsservers bereitzustellen um eine einfache und schnelle Kommunikation zu ermöglichen.

2.3 Join-Servers

Ein Join-Servers wird benötigt um den Beitritt mittels OTAA zu ermöglichen. Mehr zu OTAA kann in dem Kapitel OTAA gelesen werden. Wenn ein Endgerät dem Netzwerk beitreten möchte, leitet der Netzwerkservers die Anfragen an den Join-Servers weiter. Dieser führt dann die nötige Beitrittschritte aus, wie z.B. ableiten von Schlüsseln oder Senden der nötigen Einstellungen. Dafür ist der NwkKey und der AppKey notwendig, da diese zum verschlüsseln der Nachrichten benötigt werden. Aus Sicherheitsgründen dürfen diese Schlüssel nie über das LoRa-Netz übertragen werden, sondern müssen bei der Programmierung des Endgerätes vorgegeben werden. [SOR17b, S. 9 f.]

Der Join-Servers kann mit mehreren Netzwerkserversn verbunden werden und jeder

Netzwerkserver kann mehrere Join-Server haben.

2.4 End-Gerät

Endgeräte sind Geräte die Informationen mittels LoRa empfangen oder senden. Jedes Endgerät ist mit einem bestimmten Applikationsserver verbunden.

Jedes Endgerät muss zur korrekten Funktion mehrere wichtige Informationen haben.

- DevEUI: Globale Endgeräte_ID die eindeutig für jedes Endgerät definiert ist. Vergleichbar mit der MAC-Adresse eines TCP/IP Gerätes.
- JoinEUI: Globale Adresse des Join-Servers an den die Join-Anfrage gehen soll. Wird nur für OTAA Geräte benötigt.
- NwkKey und AppKey: Werden verwendet um spätere Schlüssel abzuleiten und die Kommunikation während der Beitrittsprozedur in ein Netzwerk abzusichern. Dafür müssen sie sowohl dem Join-Server als auch dem Endgerät bekannt sein.

[SOR17c, S.47 ff.]

3 LoRaWAN Funktionsweise

Im folgenden Kapitel wird näher auf die Funktionsweise von LoRaWAN eingegangen. Speziell, liegt der Fokus auf dem Netzwerkeitritt, das verwendete Protokoll und wie die Daten physikalisch Übertragen werden.

3.1 Schichtenmodell

Das Schichtenmodell lässt sich in zwei Teile unterteilen. Der LoRa Part ist der unterste und kümmert sich um die physikalische Übertragung der Pakete. Der LoRaWAN Part ist für die Steuerung des Netzwerkes, Definition der LoRaWAN-Klassen und das Überprüfen / Verschlüsseln der Daten zuständig.

Die unterste Schicht des LoRa Parts ist für die Anwendung der richtigen Frequenzen zuständig. In Europa muss das freiverfügbare ISM-Band 868 verwendet werden, in den Vereinigten Staaten das Band 915 und in Asien das Band 430 . [Tec15, S.7]

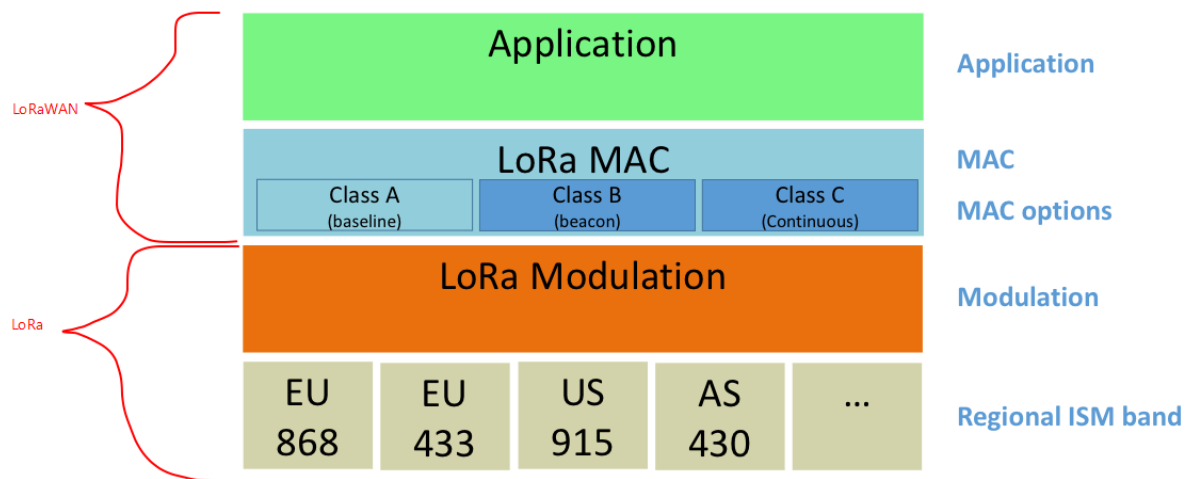


Abbildung 1: LoRaStack [Tec15, S.7]

Die darüber liegende Schicht nennt man LoRa Modulation. Sie wandelt die binären Pakete in LoRa-Signale um, so dass der Empfänger diese korrekt und effizient empfangen und wiederherstellen kann. Mehr dazu im Kapitel Übertragungsart.

Über LoRa Modulation liegt die erste LoRaWAN Schicht die LoRa MAC genannt wird. MAC steht für “Media Access Protokoll”. Dieses Protokoll wird zur Steuerung des LoRa-Netz verwendet. Diese Schicht ist außerdem für die Implementierung der einzelnen Endgeräteklassen und für das Übertragen der Steuerungskommandos zuständig. Mehr zu den Endgeräteklassen kann im Kapitel Lora Geräte Klassen und im Kapitel Protokoll gelesen werden.

Die oberste Schicht nennt sich Applikationsschicht und verpackt, verschlüsselt und authentifiziert die Nutzdaten einer Nachricht.

3.2 Netzwerkbeitritt

Endgeräte sind immer einem bestimmten Netzwerk zugeordnet. Es gibt zwei Wege um ein neues Endgeräte zu einem bestehenden Netzwerk hinzuzufügen.

3.2.1 OTAA

Die sicherste aber auch aufwendigste Methode heißt OTAA “Over-the-Air Activation”. Hierbei muss bei jedem Netzwerkbeitritt die Join-Prozedur ausgeführt werden. Dafür müssen folgende 4 Konstanten im Programm vorhanden sein: DevEUI, JionEUI, NwkKey

und AppKey.

Näheres zu den Konstanten kann im Kapitel End-Gerät nachgelesen werden.

Der NWKSKKEY ist für die Verschlüsselung der Datenpakete bis zu Gateway zuständig. Dieser Key wird vom Netzwerkservers erzeugt und muss manuell in den Code eingetragen werden. [GAS17, S.3]

Der letzte Wert heißt APPSKKEY und sichert die Kommunikation vom Endgerät zu dem Applikationsserver ab. Der Schlüssel wird genau wie der NWKSKKEY vom Netzwerkservers erzeugt und verwaltet. [GAS17, S.3]

Als erstes muss das End-Gerät eine Join- oder Rejoin-Nachricht an das Gateway senden. Die Nachricht besteht aus der JoinEUI, dem DevEUI und einer DevNonce. Mit der DevNonce sollen Replayattack verhindert werden. Die Nonce startet beim ersten Join-Request mit 0 und wird bei jedem Join-Request um eins erhöht. Die Nonce muss in einem nichtflüchtigen Speicher gespeichert werden, um den Werte unter allen Bedingungen zu sichern. Ansonsten wäre ein manueller Reset des Zählers im Netzwerkservers erforderlich. Sendet ein Endgerät einen Join-Request mit zu kleinem DevNonce, wird die Nachricht ignoriert und es ist nicht möglich dem Netzwerk beizutreten.

Das Gateway antwortet mit einer Accept-Nachricht, besteht aus einer JoinNonce, einem NetzwerkID Net_ID, einer Geräteadresse DevAddr, einem Einstellungsfeld DLSettings, einer Zeitangabe wie lange zukünftig auf eine Antwort nach dem Senden gewartet werden muss (hier RxDelay) und einer optionalen Netzwerkparameterliste CFList.

Die JoinNonce wird verwendet um Replayattacken zu verhindern und muss größer als die zuletzt gesendete sein. Ansonsten wird die Nachricht vom Endgerät ignoriert. Außerdem wird die Nonce benutzt um Schlüssel wie AppSKey herzuleiten. Für jedes Endgerät wird eine eigene JoinNonce geführt, sie darf sich nicht wiederholen. Jedes Endgerät merkt sich die letzte JoinNonce und tritt auch nur bei, wenn diese größer ist als die letzte.

Die Join-Accept Nachricht wird vom Endgerät nach JOIN_ACCEPT_DELAY1 oder JOIN_ACCEPT_DELAY2 nach dem Senden des Request erwartet. Wird die Join-Accept Nachricht zu einem anderen Zeitpunkt gesendet, wird diese nicht empfangen, da das Endgerät nicht empfangsbereit ist.

Mehr Informationen zu den Ableitungen der Schlüssel finden Sie in dem Kapitel Sicherheit.

3.2.2 ABP

Die einfachste Art des Beitritts heißt ABP was für “Activation by Personalization” steht. Hierbei muss lediglich vor Inbetriebnahme des Endgerätes fünf Konstanten definiert werden.

Als erstes wird die DevAdr(Geräteadresse) angegeben. Diese Adresse existiert nur einmal im Netzwerk und wird für die Identifizierung des Endgeräts verwendet. Die Adresse wird vom Netzwerkservers erzeugt und muss im Programmcode eingetragen sein.

Die anderen vier Werte sind die verwendeten Schlüssel zur Kommunikationsverschlüsselung: FNwkSIntKey, SNwkSIntKey, NwkSEncKey und AppSKey. Damit kann die Join-Prozedur übersprungen werden. Allerdings werden immer dieselben Schlüssel verwendet was zu einem Sicherheitsproblem werden kann.

Nach Beitritt muss das ResetInd MAC-Kommando im FOpt Feld gesendet werden. Dieses Kommando teilt dem Netzwerkservers mit, dass das Endgerät online ist und die Standardübertragungsart, -frequenz und -kanal verwendet werden. Weiter Erläuterungen folgen im nächsten Unterkapitel.

Das Netzwerk muss mit einem ResetConf-Kommando antworten. In diesem teilt es dem Endgerät die unterstützten LoRa-Versionen mit. Danach kann die Kommunikation beginnen.

3.3 Protokoll

Das LoRaWAN Protokoll ist optimiert für batteriebetriebenen Endgeräte für drahtlos Kommunikation. Um energieeffizient zu sein setzt LoRa hauptsächlich auf zwei Punkte: die Modulationstechnik und Adaptive Datenrate (ADR). Auch die Öne-HopÄrchitektur trägt zur Energieeffizienz bei. Die Modulationsart wird in Kapitel Übertragungsart beschrieben. [CB⁺17, S,1 f]

Damit der Netzwerkservers das LoRa-Netz steuern kann, werden Mac-Kommandos eingesetzt. Mit diesen Kommandos tritt man dem Netzwerk bei, kommuniziert mit den Endgeräten und steuert Frequenzen, Kanäle und vieles mehr. Da die Kommandos nur für den Netzwerkservers und die Endgeräte von Bedeutung sind, werden diese nicht an den Applikationsservers gesendet, sondern vom Netzwerkservers herausgefiltert. Im Folgenden

wird näher auf die MAC-Kommandos und die Paketstruktur eingegangen. Ein Uplink ist ein Paket dass vom Endgerät, das bildlich gesprochen “unter“ dem Gateway sitzt, an das Gateway gesendet wird. Ein Downlink ist dem entsprechend ein Paket das vom Gateway an das Endgerät gesendet wird.

3.3.1 MAC-Kommandos

MAC-Kommandos werden ausschließlich für die Steuerung und Pflege des Netzwerkes verwendet. steht für “Medium Access Control“. Da der Netzwerkservers das Netzwerk steuert, werden diese Kommandos nur zwischen dem Netzwerkservers und dem Endgerät verwendet.

MAC-Kommandos werden mittels eines “Command Identifier“ (CID) gesendet. Das ist eine 8 Bit Zahl die das MAC-Kommando repräsentiert. Nach dem CID folgen mögliche Variablen die dem Kommando mitgegeben werden. Die Anzahl und Länge der Variablen ist nicht beschränkt.

Die MAC-Kommandos müssen zwingend ausgeführt werden. Die Nachrichten werden in derselben Reihenfolge wie sie angekommen bestätigt oder beantwortet. Somit weiß der Netzwerkservers welche Kommandos bereits ausgeführt wurden. Alle Kommandos die in einem Frame gesendet wurden, müssen auch in einem Frame bestätigt / beantwortet werden. Zur Speicherung der Reihenfolge wird ein Puffer für die MAC-Kommando-Antworten verwendet. Dieser Puffer wird beim nächsten Uplink mit gesendet. Sollte der Puffer zu klein sein, werden keine weiteren Antworten eingetragen. Deshalb muss der Netzwerkservers vorher die maximale Antwortgröße errechnen und die MAC-Kommandos dementsprechend in Frames aufteilen. [SOR17c, S.29 ff.]

Im Folgenden ist eine Liste aller MAC-Kommandos angegeben:

CID	Kommandoname	Funktion
0x01	ResetInd	Wird bei dem Beitritt mittels ABP vom Endgerät gesendet. Informiert das Netzwerk über den Beitritt.
0x01	ResetConf	Wird vom Gateway gesendet und bestätigt das ResetInd-Kommando.
0x02	LinkCheckReq	Wird vom Endgerät verwendet die Verbindung zum Netzwerk zu überprüfen.
0x02	LinkCheckAns	Bestätigt das vom Endgerät gesendete LinkCheckReq-Kommando
0x03	LinkADRRReq	Wird zur Steuerung der Endgeräte im ADR-Modus verwendet
0x04	DutyCycleReq	Das Gateway Teilt dem Endgerät mit wie oft die Verbindung getestet werden soll
0x04	DutyCycleAns	Das Endgerät bestätigt den neuen Testzyklus
0x05	RXParamSetupReq	...
0x05	RXParamSetupAns	...

3.3.2 LoRa-Paketstruktur

Die Paketstruktur kommt wie beim ISO/OSI Schichtenmodell durch das "durchlaufen" des Stacks zustande. Da im Folgenden die Paketstruktur vom Groben ins Feine Betrachtend wird, werden hier als erstes die Felder der Modulationsschicht betrachtet.

Jedes Pakete besteht aus grundlegend aus 2 Felder Präambel und PHYPayload Falls es sich um einen Uplink-Paket handelt wird noch ein CRC Code hinzugefügt: Preamble, PHYPayload, CRC. In diesem Fall spricht man von einem implizit Paket oder von dem implizitem Modus. Implizit Modus bedeute dass es kein Payload Header gibt, der Felderlängen oder CRC längenangabe angibt. Diese sind somit feste zuvor definierte. Im expliziten Modus werden noch 2 Felder hinzugefügt, PHDR und PHDR_CRC. Somit sieht ein expliziteste Paket folgendermaßen aus: Preamble, PHDR, PHDR_CRC, PHYPayload. Auch hier gilt, im Falle eines Uplink-Paketes wird am Ende ein CRC Feld angefügt. Somit ergibt sich folgende Paketstruktur: Preamble, PHDR, PHDR_CRC, PHYPayload, CRC.

Die Preamble ist dafür gedacht dem Empfänger mitzuteilen dass gleich Datengesendet

werden. Deswegen wird hier nur ein Signal gesendet das ohne Informationen ist, aber von dem Empfänger wahrgenommen werden kann.

Da Teile des LoRaWAN Protokolls geschützt sind, finden sich über die PHDR und PHDR_CRC Felder kaum Informationen. Allerdings geht hervor, dass der PHDR die Länge des PHYPayloads und die Zieladresse beinhalten sollte. Das PHDR_CRC Feld wird benutzt um sicherzustellen dass die empfangenen Werte korrekt sind. Dies wird mittels des CRC Verfahrens überprüft.

Wie schon mehrfach erwähnt wird in Uplink-Nachrichten ein zusätzliches CRC Feld verwendet. CRC steht für Cyclisch Redundanz Check und wird verwendet um die Korrektheit der Nachricht zu bestätigen. PHDR, PHDR_CRC und das CRC Feld werden automatisch vom dem Funktransceiver (Modul aus Empfänger und Sender) hinzugefügt.

Die LoRa MAC Ebene fügt nun das PHYPayload Feld ein. PHYPayload steht für Physikalische Payload. Es gibt 3 Mögliche PHYPayloads Entweder wird ein MACPayload eingefügt, Join-Rejoin-Request oder aber es werden die Join-Accept Nachricht darin transportiert. Um die Daten bzw. die MAC Kommandos richtig auswerten zu können und um die Korrektheit überprüfen zu können werden einige Headers und zusätzliche Felder benötigt. Deswegen lässt sich das Feld PHYPayload weiter unterteilen in MHDR und MACPayload. Für den Fall das der MACPayload eine Join-rejoin oder MACPayload Nachricht ist, wird noch ein MIC Feld hinzugefügt(MHDR, MACPayload, MIC). MIC steht für Message Integrity Code und wird verwendet um die Korrektheit der des MACPayloads und des MHDR festzustellen.

Das MHDR Feld beschreibt wie die Daten im MACPayload Feld zu deuten sind. Wieder wird dieses Feld in Unterfelder unterteilt. MType, RFU und Major heißen die Unterfelder. Das MType Feld beschreibt die Art der Nachricht. z.B: kann hier angegeben werden ob es sich um Datennachrichten, Join-Nachrichten, ... handelt. RFU steht für "Reserved for Future Usage" zu Deutsch "für zukünftige Verwendung reserviert". Daher kann dieses Feld in der Version 1.1 und niedriger ignoriert werden. Das Major Unterfeld wird verwendet um die LoRa-Version der Nachricht zu definieren. Momentan ist nur der Wert 0 definiert. 0 steht für LoRaWAN R1. Die restlichen Werte sind für zukünftige Updates reserviert.

Mit der Unterteilung des MACPayload springen wir in dem LoRaStack noch eine Ebene

höher, in die Applikationsschicht. Enthalten im MACPayload Feld sind der Frameheader FHDR, der Frame Port FPort und der Frame Payload FRMPayload. Daten die gesendet werden sollen befinden sich in dem FRMPayload Feld. Wenn keine Daten gesendet werden, kann das FRMPayload Feld auch MAC-Kommandos enthalten. In dem Feld FPorts wird angegeben an welchen Port und somit an welche Teilapplikation des Applikationsserver die Daten geleitet werden sollen. Es gibt einige feste Ports. Port 0 ist reserviert um MAC-Kommandos im FRMPayload Feld entgegenzunehmen. Die Ports 0x01 bis 0xDF sind Anwendungsspezifische Ports und Port 0xE0 ist für das LoRaWAN Test Layer Protokoll reserviert. Falls ein anderer Port als die geraden genannten angegeben wird, wird die Nachricht verworfen.

Erneut kann der FHDR "Frame Header" in einzelne Felder unterteilt werden in DevAddr, FCtrl, FCnt, Fopts. In dem Feld DevAddr wird die Zieladresse der Nachricht vermerkt. Im Feld FCnt (Frame Counter) wird der jeweilige counterwert für die bisher gezählten Nachrichten übermittelt. Damit schützt man sich vor Replayattacks. Mehr zu den Counter kann im Kapitel Sicherheit gelesen werden. Im FOpt Feld können bis zu 5 MAC Kommandos parallel zu Daten übermittelt werden. Die Anzahl kommt auf die Menge der mitgelieferten Variablen an.

Das Letzte Feld das in Unterfelder unterteilt werden kann ist das FCtrl Feld. Hier wird das Verhalten des Gerätes gesteuert sowie Nachrichten bestätigt. Es gibt leichte Unterschiede für ein Uplink und für Downlink Nachrichten. Beide Nachrichtentypen haben ein ADR, ein ACK und ein FOptLen Feld. Im ADR wird definiert ob der Sendende bereit ist im Modus "Adaptive Data Rate" Daten zu senden, siehe Kapitel Adaptive Data Rate. Mit dem ACK Feld können empfangene Nachrichten bestätigt werden. Ob Nachrichten bestätigt werden müssen steht im MType Feld (Confirmed Data). In dem FOptLen Feld wird die Länge des FOpt Feldes mitsamt des Headers eingetragen. Wenn Das FOptLen 0 ist, ist kein FOpt Feld vorhanden.

Ein Downlinkpaket hat zusätzlich ein RFU Feld das nicht verwendet wird und ein FPending Feld. In diesem Feld kann das Gateway bzw. der Netzwerkservers dem Endgerät mitteilen, dass noch mehr Daten zu senden sind.

Dahingegen hat ein Uplinkpaket ein ClassB Feld indem das Endgerät dem Gateway mitteilt, dass es gerne auf Funktionsklasse B wechseln würde und ein ACKReq Feld.

Dieses Feld wird verwendet um zu überprüfen ob das Netzwerk noch antwortet. Die genaue Funktionsweise ist im Kapitel Adaptive Data Rate erklärt.

Eine noch genauere Darlegung der LoRa-Paketstruktur kann in den LoRaWA 1.1 Specification [SOR17c] gefunden werden.

3.4 Übertragungsart

Um die Entstandenen Pakete in Signale umzusetzen und diese effizient und gleichzeitig übertragen zu können nutzt LoRa Chirp-Spread-Spectrum (CSS). Hierbei werden die Frequenz über eine gewisse Zeit hinweg verändert. Durch messen in welche richtig, ansteigen oder abfalle, die Frequenz verändert wird, können 1 und 0 Codiert werden. Man spricht bei einem Bit von einem Chrip-Impuls. Durch aneinanderreihe der verschiedenen Impulsen ist es möglich mehrere Bits nacheinander zu übertragen. Das entstandene Signal wird auch al Sub-Chrip bezeichnet. Durch verwenden von Unterschiedlichen ansteigezeiten und abfallzeiten ist es möglich mehre Signale auf der selben Frequenz zu übertragen ohne das die Signale sich gegenseitig stören. Dies nennt man auch Spreading Factor. Außerdem kann die Parallelität durch verschiedene Frequenzbereiche verbessert werden. CSS ist besonders für große Reichweiten geeignet und somit auch bestens für Lora. Am besten ist das Signal wenn das Endgerät nahe am Gateway ist. Je weiter es entfernt desto schlechter wird das Signal. Um die Kommunikation trotzdem zu ermöglichen wird der “Spreading Factor“ erhöht. Dies hat auch den Vorteil dass der Energieaufwand gering gehalten werden kann. Analog wie Menschen auf einer Party nicht immer versuchen lauter sonder besonders langsam und deutlich sprechen. [SOR17a]

Diese Aufteilung durch den Spreading Factor und die Frequenz werden Channels erzeugt. Channels können beliebig benutzt werde. es gibt allerdings mpssen zwei regeln zu beachtet werden:

1. Channels werden per Pseudozufallszahl geändert
2. Sendezeit erfüllt die Regionalen Bestimmungen

Das Aloha Protokoll wird verwendet um festzustellen wann gesendet werden soll. Dabei wird einfach gesendet wenn Daten zum Senden vorhanden sind. Wenn nun zwei Sender

gleichzeitig auf dem selben Channel senden möchten kommt es zu einer Kollision. Dadurch kann das Gateway die empfangenen Daten nicht mehr auswerden und die Daten müssen erneut übertragen werden. Deswegen warten beide Endgeräte eine zufällige, unterschiedliche Zeit ab bis sie erneut senden.

3.4.1 Adaptive Data Rate

Adaptive Data Rate oder kurz ADR wird verwendet um immer die optimalste Senderate und die optimale Sendepower für das Endgerät zu finden und so schnellstmöglich die Daten zu senden. ADR kann nur verwendet werden wenn im FHDR Feld des LoRa-Paketes das ADR Bit gesetzt ist, siehe Protokoll. Die Steuerung durch ADR findet durch den Netzwerkservers statt. Sobald der Netzwerkservers bereit ist, setzt er das Bit im Downlink-Paket. Ist das Endgerät ebenfalls bereit setzt es ebenfalls das Bit und ADR kann verwendet werden. Falls es nicht möglich sein sollte ADR zu verwenden sollte es durch das Applikationsslayer gesteuert werden.

Die Steuerung findet durch spezielle MAC-Kommandos statt. Standardgemäß wird die höchste Übertragungsstärke verwendet und die geringste Übertragungsrate. Falls diese gedrosselt werden soll wird vom Netzwerkservers das LinkADRReq MAC –Kommando benutzt. Mit diesem wird das Endgerät informiert, dass es die Übertragungsstärke, Übertragungsrate oder den Übertragungskanal ändern soll. Die Werte sind in den Parameter codiert. Sobald die Werte geändert wurden, muss periodisch überprüft werden ob das Netzwerk die Nachrichten noch bekommt. Deswegen wird jedes Mal wenn ein Uplink empfangen wird, wird der ADR_ACK_CNT Zähler erhöht. Wenn dieser Zähler einen gewissen Schwellenwert (ADR_ACK_Limit) überschreitet, wird das ADRACKReq Bit im Uplink gesetzt. Dieses signalisiert den Netzwerkservers dass er mit einer Nachricht senden muss um die Verbindung zu bestätigen. Falls dieser Downlink nicht in ADR_ACK_Delay Frames empfangen wird, wird zuerst die Übertragungsstärke auf das Maximum gesetzt. Falls möglich wird außerdem die Datenrate verringert um die Reichweite zu erhöhen. Die Datenrate wird solange weiter, jedes ADR_ACK_Delay Frames, verringert bis diese minimal ist. Falls diese schon minimal ist müssen alle Kanäle benutzt werden. Dies wird solange probiert bis eine Verbindung hergestellt werden kann. [SOR17c, S.19f]

4 Lora Geräte Klassen

Um maximal Energie zu sparen aber trotzdem die Möglichkeit dass die Endgeräte agile Daten empfangen können wurden die Geräteklassen eingeführt. Das Hauptmerkmal der Klassen sind die unterschiedlichen Empfangsmodien. Es gibt 3 Klassen, A, B und C. Die Klasse A muss standartgemäß von jedem Endgerät implementiert werden. B und C sind Optional und müssen nicht vorhanden sein Alle Geräte die mehr als Klasse A unterstützen werden als “higher Class end-devices “ genannt. [SOR17c, S.10]

4.1 Klasse A

Klasse A wird auch (All end-Devicec) genannt, zeichnet sich durch den geringsten Stromverbrauch aus. Die Kommunikation wird von dem Endgeräte gestartet werden. Das bietet die Möglichkeit dass das Endgerät, wenn keine Daten gesendet werden müssen, in einen sehr sparsamen Schlafmodus wechselt. Um das Endgeräte nicht zum “aufwachen“ zwingen zu müssen, wurde auf einen Hardbeat oder ähnliches verzichtet. Dadurch kann das Endgerät so lange “schlafen“ wie es möchte. Klasse A erlaubt außerdem das das Endgerät andere Protokolle verwendet solange es keine LoRa Daten sendet oder empfängt. [SOR17c, S.11 ff.] Das Endgerät startet die Kommunikation in dem es Daten an das Gateway sendet. Daraufhin hat das Gateway die Möglichkeit 2 mal Daten zum Endgeräte senden. Die Empfangsfesnter werden RX1 und RX2 genannt.

Die Empfangsfesnter RX1 und RX2 müssen mindestens solange geöffnet bleiben das sie eine beginnende Übertragung feststellen können. Falls keine Übertragung empfangen wird, wird das Empfangsfesnter wieder geschlossen. Empfangsfenster RX1 wird nach RECIEV_DELAY1 zeiteiheiten +/- 20msec nach Beendigung des Uplinks geöffnet. Es wird die selbe Frequenz und Datenrate verwendet die auch bei den Downlink verwendet wurde. Wenn festgestellt in RX1 festgestellt wurde das keine Weiteren Daten mehr empfangen werden müssen kann auf das öffnen des RX2 Fensters auch verzichtet werden.

RX2 wird wird nach RECIEV_DELAY2 Zeiteinheiten +/- 20msec nach Beendigung des Uplinks geöffnet. Allerdings ist die Datenrate und Frequenz fest. Nur mittels spezieller MAC Kommandos kann dies verändert werden.

Für die Join-Prozedur wird immer die Klasse A verwendet. Die verwendete Frequenz

entspricht der Empfangsfenster RX1 oder RX2.

4.2 Klasse B

Die Klasse B (B für BEACON) bietet bidirektionale Kommunikation mit einer deterministischem Downlink Latenz. Um diese Latenz zu gewährleisten, muss die Kommunikation synchronisiert ablaufen. Außerdem muss festgestellt werden, ob das Endgerät bzw. das Gateway noch in Reichweite ist. Dies wird mittels eines periodischem Beacon ermittelt. Dieser Beacon wird regelmäßig vom Gateway gesendet und dient der Synchronisation der Endgeräte. Zeitpunkten gesendet werden realisiert. Die Latenz ist einstellbar und kann bis zu 128 Sekunden. [SOR17c, S.66 ff.]

Die Endgeräte öffnen in regelmäßigen Abständen ein Empfangsfenster das Pingslot genannt wird. Ein Downlink der in einem Pingslot gesendet wird, wird Ping genannt. Da immer das Gateway mit dem besten Empfang die Daten an das Gateway sendet, muss das Endgerät selbständig feststellen wenn es einen Beacon mit einer unbekannten ID bekommt und durch eine Uplink dem Server mitteilen das es sich in einer neuen Umgebung ist. Dadurch lernt der Server wo sich das Endgerät befindet und kann das Gateway mit dem besten Empfang wählen. Obwohl das Endgerät durch die periodischen Beacon nicht "schlafen" kann, ist die Klasse B für den Batteriebetrieb gedacht.

4.2.1 Klassenwechsel A nach B

Um einen Wechsel überhaupt zu ermöglichen muss der Netzwerkserver die Standard Pingslot Periode, die Pingslot Datenrate und den Pingslot Kanal kennen. Alle Endgeräte treten in Klasse A dem Netzwerk bei. Das wechseln in die Klasse B wird durch folgenden Prozess realisiert.

Als erstes muss das Programm des Endgerätes beim LoRaWAN Layer anfragen ob es möglich ist in Klasse B zu wechseln. Der Layer sucht nun nach einem Beacon. Wird ein Beacon entdeckt, wird die BEACON_LOCKED Serviceprimitive zurückgeliefert. Wenn kein Beaconempfangen wurde wird die BEACON_NOT_FOUND primitive zurückgegeben. Um diesen Prozess zu beschleunigen kann das DeviceTimeReq MAC-Kommando verwendet werden. Damit wird das Gateway aufgefordert einen Beacon zu senden. Nun

kann das Endgerät in den Modus B wechseln.

Als Zweites setzt der MAC Layer des Endgerätes das Class B Bit im FCtrl Feld des Uplinks auf 1. Der MAC Layer ist auch verantwortlich die Pingslot und für die Beacons Empfangsfenster zu öffnen. Dabei muss mit der größten möglichen Abweichung der internen Uhr gerechnet werden und dementsprechend die Empfangsfenster angepasst werden. Diese darf pro Beacon nicht mehr als $\pm 1.3\text{msec}$ liegen. [SOR17c, S.73] Der Inhalt des empfangenen Beacon wird mit der Signalstärke dem Programm des Endgerätes zur weiteren Verarbeitung gesendet. Damit kann z.B. die innere Uhr nachgestellt werden oder Ortswechsel festgestellt werden.

4.2.2 Betrieb

Damit der Netzwerkservers dem Endgerät mitteilen kann dass die Pingslot Frequenz und die Datenrate geändert werden soll gibt es den PingSlotChannelReq Mac-Kommando. Die neuen Werte sind in den Argumenten enthalten.

Das Endgerät kann die Periode der Pingslot zu einer beliebigen Zeit ändern. Ist dies der Fall, so muss das Endgerät in Klasse A wechseln und mittels dem MAC-Kommando PingSlotChannelReq die geänderte Periode mitteilen. Danach kann zurück in die Klasse B gewechselt werden.

Falls einige länger als 2 Stunden kein Beacon empfangen wird, kann die Synchronisation mit dem Netzwerk verloren gehen. Dadurch funktioniert die Kommunikation in Klasse B nicht mehr und es wird in Klasse A gewechselt. Nun kann versucht werden eine Verbindung mit der Klasse A aufzubauen. Das Programm des Endgerätes kann versuchen wieder in Klasse B zu wechseln. Dieser Prozess kann sich immer wieder wiederholen.

Um auch innerhalb der minimal 2 Stunden in den kein Beacon empfangen wurde eine Kommunikation zu ermöglichen wird jedes Mal wenn ein Beacon verloren geht in den Beacon-Less Modus gewechselt. Dieser Modus orientiert sich ausschließlich an der internen Uhr. Um die Abweichung auszugleichen werden die Empfangsfenster immer früher begonnen und immer später beendet. Das bedeutet einen höheren Energieverbrauch aber auch eine höhere Wahrscheinlichkeit noch Daten zu empfangen obwohl die Uhren des Gateways und des Endgerätes auseinanderlaufen.

4.2.3 Singel / Multicast

In Klasse B können die Nachrichten als Singelcast oder als Multicast Nachrichten verwendet werden. Eine Singelcast Nachricht wird an des Endgerät das im DevAfdR Feld der Nachricht codiert ist gesendet. Im Multicastmodus wird das Paket an mehrere Endgeräte gesendet. Damit diese möglich ist müssen sich die Endgeräte dieselbe Multicast Adresse und die dazugehörigen Schlüssel teilen. Durch verschiedene Multicastadresse ist es möglich sogenannte Multicastgruppen zu erzeugen die nicht alle sondern nur ein Teil aller Endgeräte beinhalten. LoRaWAN gibt allerdings keine Methode vor wie die Adressen und Schlüssel verteilt werden. Diese Aufgabe muss in der Applikationsebene sprich im Programm der Endgeräte bzw. des Applikationsserver oder die Schlüssel und Adresse werden fest in den Programmcode programmiert.

In Multicastnachrichten sind keine MAC-Kommandos erlaubt. Nur Daten dürfen als Multicastnachrichten übertragen werden. Dies wurde eingeführt da Multicastnachrichten nicht dieselbe Robustheit wie Singelcastnachrichten haben. Die Nachrichten dürfen nicht bestätigt werden da sonst eine große Last für das Netz entstehen würde. [SOR17c, S.84]

4.2.4 Beacon

Wie schon erwähnt werden die Beacons verwendet um das Endgerät mit dem Netzwerk zu synchronisieren. Deswegen werden die Beacon Periodisch gesendet. Die Zeit zwischen zwei Beacon wird BEACKON_Period genannt. Die Endgeräte öffnen Empfangsfenster um diese Beacons zu empfangen. Ein Beacon zu übertragen dauert BEACON_RESERVED lange. Das Beacon-Empfangsfenster wird BEACKON_GUARD früher geöffnet um sicher zu stellen das Beacon auch wirklich erkannt werden kann. Außerdem wird die Beacon_GUARD benutzt um sicherzustellen dass kein Pingslot mehr geöffnet ist. Deswegen muss diese Beacon_GUARD mindestens so lang sein wie ein maximaler Pingslot. Das hat den Vorteil, dass nicht darauf gedacht werden muss wann ein Pingslot geöffnet wurde, da er im Zweifelsfall fertig ist bevor ein Beacon empfangen wird. Vergleichbar ist dies mit der Distributed Coordination die bei WLAN Anwendung findet.

Während versucht wird ein Beacon zu empfangen kann kein Pingslot geöffnet werden

Um Synchronisierungen durch die Beacon zu vermeiden, wie "alle Endgeräte wollen

sofort nach den Beacon senden wollen“, wird mittels zufälliger Wartezeiten und zufälliger Pingslotperiode verhindert. Durch den Zufall wird die Wahrscheinlichkeit einer Kollision geringer.

Beacons haben ihr eigenes Paketformat. Diese Pakete sind immer gleich lang. Dadurch kann auf Header verzichtet werden was auch der Verarbeitungsgeschwindigkeit zugutekommt. Wie auch ein normales LoRa-Paket, so besteht auch das erste Feld des Beacon-Paketes aus der Preamble. Danach folgt der BCNPayload. Der BCNPayload(Beacon Payload) lässt sich unterteilen in RFU, Time, CRC, GWSpecific, RFU, CRC. Die zwei CRC Felder weisen schon auf die logische Unterteilung in zwei Hälften hin. Der erste Teil enthält Beacon spezifische Informationen (time und CRC). In dem Time Feld ist die Zeit seit 00:00:00, 01.01.1980 Modulo 2^{32} enthalten. das CRC Feld wird verwendet um die Korrektheit des Zeit und des RFU Feldes zu versichern. RFU steht wieder für “Reserved for Future Usage“ und wird nicht verwendet. Die andere Hälfte ist Gatewayspezifisch. Sie enthält das GWSpecific Feld und ein RFU Feld. Beide Felder sind durch ein zweites CRC Feld abgesichert ist. Das GWSpezifisch Feld lässt sich unterteilen in InfoDesc und Info Felder. Das InfoDesc gibt an auf was sich das Infofeld bezieht. 0 gibt an dass die GPS-Koordinaten der ersten Antenne folgen. 1 steht für die GPS-Koordinaten der zweiten Antenne und 2 steht für die GPS-Koordinaten der dritten Antenne. Die Werte 3 bis 127 sind noch solange sich im Info Feld Koordinaten enthalten kann dieses unterteilt werden in Längen- und Breitengrad.

Auch Klasse A kann den Beacon nutzen, um herauszufinden von welchem Gateway es gerade Daten empfängt und um eventuelle Standortwechsel festzustellen.

In Europa werden die Beacons auf einer festen Frequenz übertragen die sich nicht ändert. Nur über das MAC-Kommando PingSlotChannelReq.

4.3 Klasse C

C steht für CONTINUOUSLY Listening. Wie der Name schon sagt ist hier dauernd ein Empfangsfenster geöffnet. Dafür wird es ermöglicht fast Latzen frei zu übertragen. Dies bedeutet aber auch dass der Stromverbrauch am höchsten ist und somit die Klasse C nicht für den Batteriebetrieb geeignet.

Geräte die Klasse C implementieren sollen auch nicht die Klasse B implementieren das

es sonst zu Fehlern kommen kann.

Diese Klasse verwendet die gleichen Empfangsfenster mit der gleichen Frequenz wie in Klasse A. Der große Unterschied besteht allerdings darin, dass RX2 immer dann geöffnet ist, wenn nicht gerade Daten an das Gateway gesendet werden oder RX1 geöffnet ist.

Auch in Klasse C ist es, wie in B, möglich, Multicastnachrichten zu senden. Hierbei gelten die gleichen Regeln wie bei Klasse B.

5 Sicherheit

Sicherheit in netzwerkfähigen Systemen ist ein sehr wichtiges und heiß diskutiertes Thema. Da alle LoRa-Geräte über das Medium "Luft" die Daten übertragen, sind diese auch für alle empfangbar. Deswegen ist es wichtig, die Daten zu verschlüsseln, um zu verhindern, dass Dritte die Daten mitlesen. Genauso wichtig ist die Datenauthentifizierung und somit zu überprüfen, wer die Daten gesendet hat. Auch verschlüsselte Daten können von Dritten mitgelesen werden. Allerdings können diese nichts mit den verschlüsselten Daten anfangen. Das einzige, was Dritte mit den Daten tun können, ist eine sogenannte Replayattack durchzuführen, indem sie das empfangene Datenpaket nochmal senden und so versuchen, die Kommunikation zu stören oder Fehlinformationen zu streuen. Um Replayattack entgegenzuwirken, werden Zähler oder Nonce mit den Daten mitgeschickt. Im folgenden Text wird betrachtet, wie LoRa diese Sicherheitsfeatures implementiert hat.

5.0.1 Schlüssel

Schon der Beitritt eines Endgerätes muss verschlüsselt sein. Dafür werden zwei verschiedene Schlüssel verwendet. Die Join-Nachricht wird mit dem JSIntKey (Joining Session Encryption Key) verschlüsselt. Der Schlüssel wird mittels der aes128 Verschlüsselung generiert. Speziell wird der NWkKey mit dem deinem 16Byte langen Wort verschlüsselt. Das Wort besteht am Anfang aus der Konstante 0x06, danach folgt die DevEUI und zum Schluss wird mit =en aufgefüllt. Die Accept-Nachricht wird ebenfalls verschlüsselt. Allerdings wird hier der JSEncKey (Joining Session Encryption Key) verwendet. Dieser wird auf die gleiche Weise erzeugt, allerdings wird bei dem Verschlüsselungswort 0x06 durch 0x05 ersetzt. JSIntKey wird außerdem verwendet, um den MIC-Code in der Join-Nachricht zu

berechnen.

Alle MAC-Kommandos die an Port 0 gesendet werden, werden separat verschlüsselt. Dies bietet eine Höhere Sicherheit, selbst wenn Daten entschlüsselt werden können, so bleibt es unmöglich falsche MAC-Kommandos zu verschlüsseln und zu einem Endgerät zu senden. Hierzu wird der Network session encryption key (NwkSEncKey) verwendet. Dieser wird auf die selbe weise erzeugt wie JSIntKey. Allerdings besteht hier das Schlüsselwort aus 0x04, der JoinNonce, der JoinEUI und dem DevNonce. Diese Werte werden bei dem Beitritt mittels OTAA erzeugt. Siehe: ??.

Um die Integrität der Daten zu bestimmen werden zwei schlüssel verwendet. Der FNwkSIntKey oder lang Forwarding Network session integrity key wird benutzt um den MIC-Code für die Integritätsbestimmung abzuleiten. Auch dieser Schlüssel wird wie JSIntKey abgeleitet aber das Schlüsselwort besteht aus 0x01, der JoinNonce, der JoinEUI und DevNonce. Der SNwkSIntKey (Serving Network session integrity key) stellt das gegenstück da und wird verwendet den MIC-Code zu testen ob nichts verändert wurde. Hier besteht das Schlüsselwort aus 0x03, der JoinNonce, der JoinEUI und der DevNonce.

JSSEncKey, JSIntKey, FNwkSIntKey, SNwkSIntKey, NwkSEncKey sind für jedes Endgerät unterschiedlich.

Momentan sind nur die MAC-Kommentare vor dem Mitlesen geschützt. Natürlich müssen auch die mitgelieferten Daten geschützt werden. Dies geschieht unter Verwendung des AppSKeys. Der Applikation Session Key wird von dem Applikationsserver und dem Endgerät verwendet. Dieser Schlüssel wird vom Programmierer vorgegeben, da er mit dem Schlüssel des Applikationsservers übereinstimmen muss. [SOR17c, S.50 ff]

5.0.2 Zähler

Jedes Endgerät hat drei verschiedene Counter. Ein Framecounter für Uplinkframes FCntUp und zwei für Downlinkframes NFCntDown und AFCntDown.

Der FCntUp counter wird für jeden Uplink der gesendet wird erhöht. Der stand wird in das FCnt feld der Nachricht mit eingefügt. Da das Gateway den selben zähler hat, kann es erkennen ob die Nachricht schon einmal gesendet wurde in dem es nur Nachrichten mit einem größeren FCnt Wert als der eigene FCntUp Zähler enthält.

Genau so wird mit den Downlink counter verfahren, bloß auf dem Endgerät. Im NF-

CntDown Zähler werden die MAC-Kommando Nachrichten gezählt. AFCntDown wird für alle anderen Nachrichten verwendet. [?, S.22 ff]

Literatur

- [CB⁺17] CHEONG, PHUI SAN, JOHAN BERGS, , CHRIS HAWINKEL und JEROEN FAMAËY: *Comparison of LoRaWAN Classes and their Power Consumption*. <https://ieeexplore.ieee.org/abstract/document/8240313>, November 2017. Eingesehen am 09.04.2019.
- [GAS17] GEMALTO, ACTILITY und SEMTECH: *LoRaWANTM SECURITY WHITE PAPER PREPARED FOR THE LoRa ALLIANCETM*. <https://lora-alliance.org/resource-hub/lora-alliance-security-whitepaper>, Februar 2017. Eingesehen am 09.04.2019.
- [SOR17a] SORNIN, N. (Herausgeber): *Exploring LoRa and LoRaWAN(todo)*. Lora-Alliance, <https://tools.ietf.org/pdf/rfc8376.pdf>, 1.1 Auflage, Oktober 2017. Eingesehen am 09.04.2019.
- [SOR17b] SORNIN, N. (Herausgeber): *LoRaWANTM 1.1 Backend(todo2)*. Lora-Alliance, <https://tools.ietf.org/pdf/rfc8376.pdf>, 1.1 Auflage, Oktober 2017. Eingesehen am 09.04.2019.
- [SOR17c] SORNIN, N. (Herausgeber): *LoRaWANTM 1.1 Specification*. Lora-Alliance, <https://tools.ietf.org/pdf/rfc8376.pdf>, 1.1 Auflage, Oktober 2017. Eingesehen am 09.04.2019.
- [Tec15] TECHNICALMARKETINGWORKGROUP1: *A technical overview of LoRa® and LoRaWANTM*. <https://lora-alliance.org/resource-hub/what-lorawantm>, November 2015. Eingesehen am 09.04.2019.