Seminararbeit: LoRaWAN

Tobias Sigmann

26. Mai 2019

Inhaltsverzeichnis

1 Einführung

Warum noch eine weiteres Funknetz? Reichen die bestehenden Funknetzt nicht aus?

Heutige freie Funknetze sind GSM, Wi-Fi, Bluetooth, usw.

In jedem Funknetz sind drei Parameter wichtig: Bandbreite/Übertragungsrate, Energiebedarf und Reichweite. Wi-Fi hat eine hohe Bandbreite, hoher Energiebedarf und eine begrenzte Reichweite. GSM hat eine höhere Bandbreite, einen hohen Energiebedarf und mittlere Reichweite. Bluetooth hat geringe Bandbreite, eine niedringen Energiebedarf und eine niedrige Reichweite.

Für sehr hohe Reichweiten und sehr niedringen Energiebedarf gab es bisher noch keine Lösung. Hier setzen nun Low-Power-Wide-Area-Netzwerk-Technologien (LPWAN) wie LoRa, Sigfox und NarrowBand-IoT (NB-IoT) an. Diese Netzwerke sind für IoT, Smart-City-Applikationen und andere LPWAN-Anwendungen geeignet. Beispiele sind smarte Mülleimer, Straßenlaternen, Parkplätze aber auch privat für SmartHomes geeignet. Der Vorteil einer sehr hohe Reichweiten (bis zu 20km) und sehr niedringen Energiebedarf (Batterielaufzeit bis zu mehreren Jahren) hat den Nachteil einer sehr gerigen Bandbreite (zwischen 0.3 und 50 kbps).

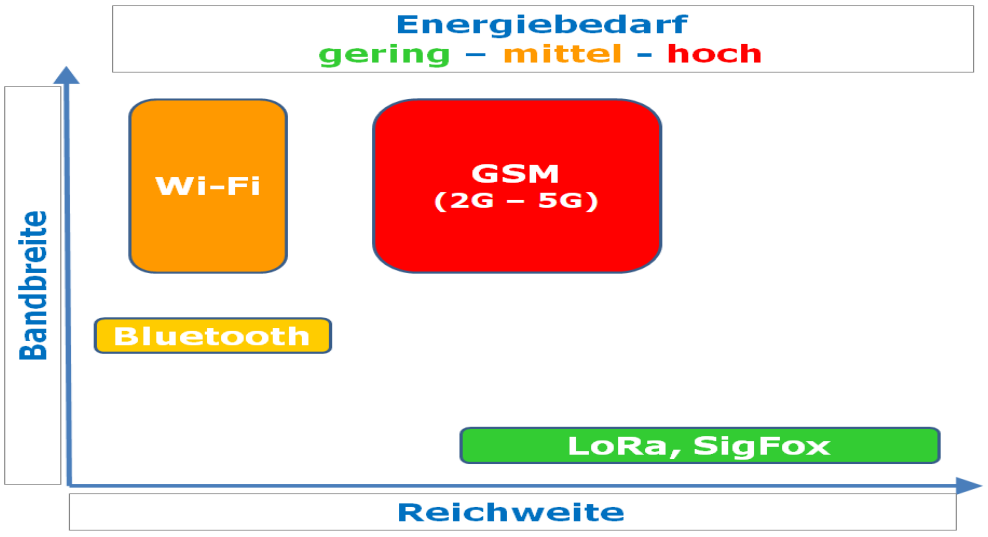

Abbildung 1: Funknetzvergleich

Im folgenen wird der Aufbau und die Funktion von LoRaWAN 1.1 beschreiben. LoRaWAN steht für “Long Range Wide Area Network”. [Wor18]

2 Aufbau eines LoRa-Netzwerk

LoRa ist ideal für IoT-Geräte, da per einfachem Netzwerkaufbau Datenaustausch mit dem Internet möglicht 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 Netzwerkserver zuständig der über die Gateways das Netzwerk kontrolliert und steuert. Gleichzeitig stellt er die Verbindung zu einem Applikationsserver her, indem 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 hoher Endgeräteanzahl, hoher Siganlqualität und entsprechnder Netzwerksicherheit. [Wor18, S. 8 ff.]

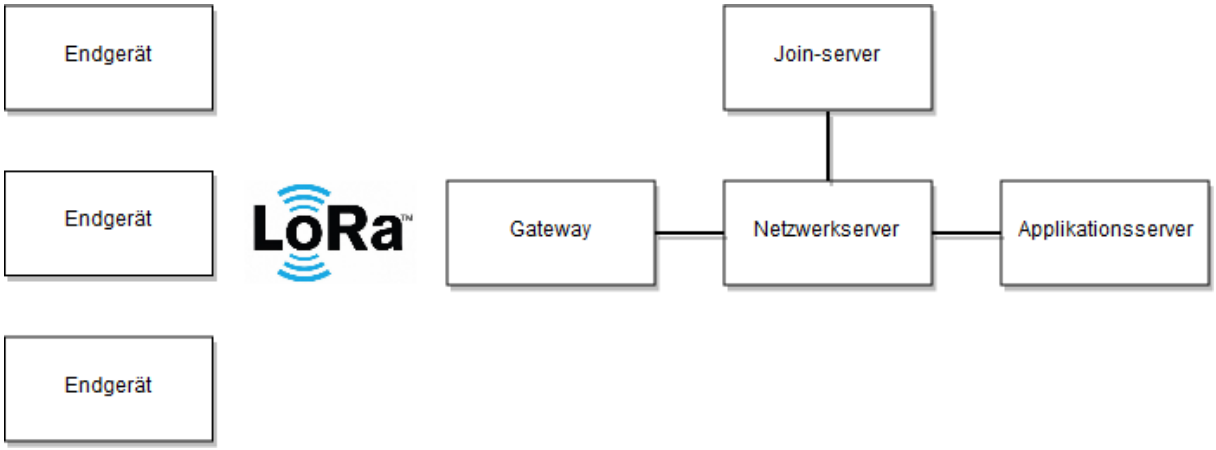

Abbildung 2: Beispiel eines LoRa-Netzwerks

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 Netzwerkserver verbunden.

Ein Endgerät kann gleichzeitig an mehreren Gateways senden. Der Netzwerkserver überprüft die Pakete auf Duplikate und stellt sicher dass jedes Paket nur einmal an die Applikationsserver 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 sech bis acht-fach erhöhen. vgl. [Wor18, S.10]

2.2 Netzwerkserver

Der Netzwerkserver ist das "Herzstück" eines jeden LoRa-Netzwerkes. Er kann mit mehreren Gateways und mehreren Applikationsserver verbunden sein.

Die wichtigste Aufgabe des Netzwerksserver ist die Steuerung des LoRa-Netzwerks. Der Server 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 und Integrität. Wie auch 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 Applikationsserver zu entlasten, puffert der Netzwerkserver die Daten und sendet diese zum nächst möglichen Zeitpunkt.

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

2.3 Join-Server

Ein Join-Server wird benötigt um den Beitritt mittels OTAA zu ermöglichen. Mehr zu OTAA kann in dem Kappitel 3.2.1 OTAA gelesen werden. Wenn ein Endgerät dem Netzwerk beitreten möchte, leitet der Netzwerkserver die Anfragen an den Join-Server 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. [YEG17, S. 9 f.]

Der Join-Server kann mit mehreren Netzwerkservern 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 Beitrittprozedur in ein Netzwerk abzusichern. Dafür müssen sie sowohl dem Join-Server als auch dem Endgerät bekannt sein.

[SOR17, S.47 ff.]

3 LoRaWAN Funktionsweise

Im folgenden Kapitel wird näher auf die Funktionsweise von LoRaWAN eingegangen. Speziell liegt der Fokus auf dem Netzwerkbeitritt, 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 Pakte. 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 . [Wor18, 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. Mehre dazu im Kapitel 3.4 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 4 LoRa Geräte Klassen und im Kapitel 3.3 Protokoll gelesen werden.

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

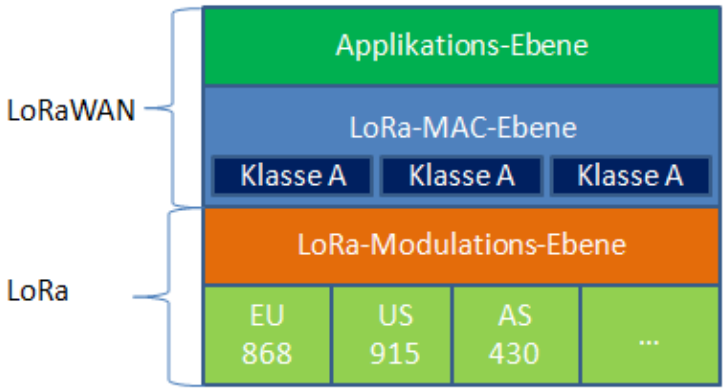

Abbildung 3: Schichtenmodell

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 2.4 End-Gerät nachgelesen werden.

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

Der letzte Wert heißt APPSKEY und sichert die Kommunikation vom Endgerät zu dem Applikationsserver ab. Der Schlüssel wird genau wie der NWKSKEY vom Netzwerkserver 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 im einem nichtflüchtigen Speicher gespeichert werde, um die Werte unter allen Bedingungen zu sichern. Ansonsten wäre ein manueller Reset des Zählers im Netzwerkserver 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 Replayatacken zu verhindern und muss größer als die zuletzt gesendete sein. Ansonsten wird die Nachricht vom Endgerät ignoriert. Außer-

dem 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 andern Zeitpunkt gesendet, wird diese nicht empfangen, da das Endgerät nicht empfangsbereit ist.

Mehr Informationen zu den Ableitungen der Schlüssel finden Sie in dem Kapitel 5 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 Indentifizierung des Endgeräts verwendet. Die Adresse wird vom Netzwerkserver 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-Prozdeur 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 Netzwerkserver 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 batteriebetrieben Endgeräte für drahtlos Kommunikation. Um energieeffizent zu sein setzt LoRa hauptsächlich auf zwei Punkte: die Modulationstechnik und Adaptive Datenrate (ADR). Auch die "One-Hop" Architektur trägt zur Energieeffizenz bei. Die Modulationsart wird in Kapittel 3.4 Übertragungsart beschrieben. [CBHF17, S,1 f]

Damit der Netzwerkserver 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 Netzwerkserver und die Endgeräte von Bedeutung sind, werden diese nicht an den Applikationsserver gesendet, sondern vom Netzwerkserver herausgefiltert. Im folgenden wird näher auf die MAC-Kommandos und die Paketstruktur eingegangen. Ein Uplink ist ein Paket das 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. MAC steht für "Medium Access Control". Da der Netzwerkserver das Netzwerk steuert, werden diese Kommandos nur zwischen dem Netzwerkserver und dem Endgerät verwedet.

MAC-Kommandos werden mittels eines "Command Indentifier" (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 Netzwerkserver 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 ver-

wendet. Dieser Puffer wird beim nächsten Uplink mit gesendet. Sollte der Puffer zu klein sein, werden keine weiteren Antworten eingetragen. Deshalb muss der Netzwerkserver vorher die maximale Antwortgröße errechnen und die MAC-Kommandos dementsprechend in Frames aufteilen. [SOR17, 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 um die Verbindung zum Netzwerk zu überprüfen.
0x02	LinkCheckAns	Bestätigt das vom Endgerät gesendete LinkCheckReq-Kommando
0x03	LinkADRReq	Wird zu 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	Das Gateway teilt dem Endgerät die neuen Pingslotparameter fest
0x05	RXParamSetupAns	Das Endgerät bestätig die neuen Parameter
0x06	DevStatusReq	Gateway fragt den Status des Endgeräts ab
0x06	DevStatusAns	Endgerät sendent seinen Status
0x07	NewChannelReq	Gatway weist das Endgerät an den Channel zu wechseln
0x07	NewChannelAns	Endgerät bestätigt Channelwechsel
0x08	RXTimingSetupReq	Gatway weist das Endgerät an an die Empfangsfensterverzögerung zu änder
0x08	RXTimingSetupAns	Endgerät bestätigt RXTimingSetupReq

0x09	TxParamSetupReq	Gatway sendet neue Sendeeinstellungen an das Endgerät
0x09	TxParamSetupAns	Endgerät bestätigt TxParamSetupReq
0x0A	DlChannelReq	Wid vom Gateway gesendet und ändert die Empfangsfenstereinstellungen des RX1-Fensters
0x0A	DlChannelAns	Endgerät bestätigt DlChannelReq
0x0B	RekeyInd	Endgerät meldt seinen Beitritt
0x0B	RekeyConf	Gateway bestätigt Beitritt
0x0C	ADRParamSetupReq	Gatway ändert ADR_ACK_LIMT and ADR_ACK_DELAY
0x0C	ADRParamSetupAns	Endgerät bestätigt ADRParamSetupReq
0x0D	DeviceTimeReq	Endgerät fragt das aktuelle Datum und die aktuelle Zeit vom Gateway an
0x0D	DeviceTimeAns	Gateway beantwortet DeviceTimeReq
0x0E	ForceRejoinReq	Gateway weist das Endgerät an nach verbindungsabbruch sofort eine Rejoin-Anfrage zu senden
0x0F	RejoinParamSetupReq	Wird vom Gateway gesendet und stellt die Rejoin-Periode ein
0x0F	RejoinParamSetupAns	Endgerät bestätigt RejoinParamSetupReq
0x10	PingSlotInfoReq	Endgerät teilt dem Gateway die neue Pingslot-Periode mit
0x10	PingSlotInfoAns	Gateway bestätigt PingslotInfoReq
0x11	PingSlotChannelReq	Gatway definiert neuen Pingslot-Channel
0x11	PingSlotChannelAns	Endgerät bestätigt PingSlotChannelReq
0x12	BeaconTimingReq	Veraltet
0x12	BeaconTimingAns	Veraltet
0x13	BeaconFreqReq	Gatway ändert die Beaconfreuenz
0x13	BeaconFreqAns	Endgerät bestätigt BeaconFreqReq
0x20	DeviceModeInd	Endgerät teil Gateway mit ob es in Klasse A oder C arbeitet
0x20	DeviceModeConf	Gateway bestätigt DeviceModeInd

0x80-0xFF	Proprietary	Reserviert für Netzwerkerweiterungen

3.3.2 LoRa-Paketstruktur

Die Paketstruktur kommt wie beim ISO/OSI Schichtenmodel durch das "durchlaufen" des Stacks zustande. Die Paketstruktur wird hier Top-Down betrachtet. Als erstes werden die Felder der Modulationsschicht betrachtet.

Jedes Pakete besteht grundlegend aus zwei Felder: Präambel und PHYPayload. Falls es sich um einen Uplink-Paket handelt wird noch ein CRC Code hinzugefügt, also Preamble, PHYPayload, CRC. In diesem Fall spricht man von einem impliziten Paket oder vom implizitem Modus. Impliziter Modus bedeute, dass es kein Payload Header gibt. Payload Header gibt die Felderlänge und die CRC Längenangabe an. Diese sind somit zuvor fest zu definieren. Im expliziten Modus werden noch zwei Felder hinzugefügt, PHDR und PHDR_CRC. Somit sieht ein explizites 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 dann folgende Paketstruktur: Preamble, PHDR, PHDR_CRC, PHYPayload, CRC.

LoRa-Paket					
Feld	Preamble	PHDR	PHDR_CRC	PHYPayload	CRC
Größe in Byte	8	?	?	18-32	2

Abbildung 4: Aufbau einer LoRa-NAchricht

Die Preamble ist eine Startsequenz und teilt dem Empfänger mit, dass gleich Daten gesendet werden. Sie ist nur ein Signal ohne Informationen.

Da Teile des LoRaWAN-Protokolls geschützt sind, findet man über die PHDR und PHDR_CRC Felder nur sehr wenig Informationen. Allerdings geht hervor, dass der PHDR die Länge des PHYPayloads und die Zieladresse beinhaltet. Mit dem PHDR_CRC Feld wird die Korrektheit der empfangenen Werte sichergestellt. 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 zur Bestätigung der

Nachrichtenkorrektheit verwendet. 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 drei mögliche PHYPayloads. Entweder wird ein MACPaylod eingefügt, Join-Rejon-Request oder aber es werden die Join-Accept Nachricht darin transportiert. Um die Daten bzw. die MAC Kommandos richtig auszuwerten und um die Korrektheit zu überprüfen, werden einige Headers und zusätzliche Felder benötigt. Deshalb lässt sich das Feld PHYPayload in MHDR und MACPayload unterteilen. Falls der MACPaylod eine Join-Rejoin oder MACPayload Nachricht ist, wird noch ein MIC Feld hinzugefügt: MHDR, MACPayload und MIC. MIC steht für "Message Integrity Code" und mit ihr wird die Korrektheit der des MACPayloads und des MHDR ermittelt.

PHYPayload			
Feld	MHDR	MACPayload	MIC
Größe in Byte	1	13-27	4

Abbildung 5: Aufbau des PHYPayload-Felds

Das MHDR Feld definiert die Daten im MACPayload-Feld. Auch dieses wird Feld in Unterfelder unterteilt: MType, RFU und Major. MType steht für "Message Type". Das MType-Feld beschreibt die Art der Nachricht, z.B. Datennachrichten oder Join-Nachrichten, ...

MType	Description
000	Join-request
001	Join-accept
010	Unconfirmed Data Up
011	Unconfirmed Data Down
100	Confirmed Data Up
101	Confirmed Data Down
110	Rejoin-request
111	Proprietary

RFU steht für “Reserved for Future Usag“. Daher kann dieses Feld in der Version 1.1 und niedriger ignoriert werden. Das Major Unterfeld wird verwendet um das 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.

MHDR			
Feld	MType	RFU	Major
Größe in Bit	3	3	2

Abbildung 6: Aufbau des MHDR-Felds

Mit der Unterteilung des MACPayload springen wir in dem LoRa-Stack noch eine Ebene höher, in die Applikationsschicht. Im MACPayload-Feld sind Frameheader FHDR, Frame-Port FPort und Frame-Payload FRMPayload enthalten. Nutzdaten befinden sich im FRMPayload-Feld. Werden keine Daten gesendet, entällt das FRMPayload-Feld MAC-Kommandos. In dem Feld FPorts wird angegeben an welchen Port und somit an welche Teilapplikation des Applikationsserver die Daten geleitet werden sollen. Es gibt einige fest definierte Ports. Port 0x00 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. Nachrichten mit anderen Port-Adressen werden verworfen.

MACPayload			
Feld	FHDR	FPort	FRMPayload
Größe in Byte	7 bis 22	0 bis 1	?

Abbildung 7: Aufbau des MACPayload-Felds

Auch der FHDR “Frame Header” wird in einzelne Felder unterteilt: DevAddr, FCtrl, FCnt und Fopts. In dem Feld DevAddr (Device Address) wird die Zieladresse der Nachricht vermerkt. Im Feld FCnt (Frame Counter) wird der jeweilige Zählerwert (Counter) für die bisher gezählten Nachrichten übermittelt. Hiermit schützt man sich vor Replayattacks. Mehr zu den Counter kann im Kapitel 5.2 Zähler gelesen werden. Im FOpt Feld können bis zu fünf MAC-Kommandos parallel zur Datenübertragung übermittelt werden. Die Anzahl kommt auf die Menge der mitgelieferten Variablen an.

FHDR				
Feld	DevAddr	FCtrl	FCnt	FOpts
Größe in Byte	4	1	2	0 bis 15

Abbildung 8: Aufbau des FHDR-Felds

Das letzte Feld, das in Unterfelder unterteilt werden kann, ist das FCtrl-Feld. Hiermit wird das Endgeräte gesteuert und Nachrichten bestätigt. Es gibt leichte Unterschiede für Uplink- und Downlink-Nachrichten. Beide Nachrichtentypen haben ein ADR, ein ACK und ein FOptsLen Feld. Das ADR-Feld definiert, ob im Modus "Adaptive Data Rate" Daten gesendet werden oder im Standardmodus. Siehe Kapitel 3.4.1 Adaptive Data Rate. Mit dem ACK Feld können empfangene Pakete bestätigt werden. Ob Nachrichten bestätigt werden müssen, ist im MType Feld (Confirmed Data) definiert. In dem FOptsLen Feld wird die Länge des FOpts Feldes mitsamt des Headers eingetragen. Ist das FOptsLen 0, so ist kein FOpts 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 Netzwerkserver dem Endgerät mitteilen, dass noch mehr Daten folgen.

FCtrl (Downlink)					
Feld	ADR	RFU	ACK	FPending	FOptsLen
Größe in Bit	1	1	1	1	4

Abbildung 9: Aufbau des FCTRL(Downlink)-Felds

Dahingegen hat ein Uplink-Paket ein ClassB. Hier teilt das Endgerät dem Gateway mit, dass es auf Funktionsklasse B wechseln will. Zusätzlich hat das Uplink-Paket ein ADRACKReq Feld. Dieses Feld wird verwendet um zu überprüfen, dass das Netzwerk noch antwortet. Die genaue Funktionsweise ist im Kapitel 3.4.1 Adaptive Data Rate erklärt.

Eine detailierte Beschreibung der LoRa-Paketstruktur findet man in der LoRaWA 1.1 Specification [SOR17].

FCtrl (Uplink)					
Feld	ADR	ADRACKReq	ACK	ClassB	FOptsLen
Größe in Bit	1	1	1	1	4

Abbildung 10: Aufbau des FCTRL(Uplink)-Feld

3.4 Übertragungsart

Um die entstandenen Pakete in Signale umzusetzen und diese effizient und gleichzeitig übertragen zu können nutzt LoRa Chrip-Spread-Spectrum (CSS). Bei dieser Frequenzmodulation wird Frequenzanstieg als Binär 1 und Frequenzabfall als Binär 0 codiert. Man spricht bei einem Bit von einem Chrip-Impuls. Durch aneinanderreihen der verschiedenen Impulse können mehrere Bits übertragen werden. Das entstandene Signal wird auch als Sub-Chrip bezeichnet. Da verschiedene Anstiegszeiten und Abfallzeiten verwendet werden, können mehre Signale auf derselben Frequenz parallel übertragen werden. Dies nennt man auch Spreading Factor. Die Parallelität wird durch Verwendung von verschiedenen Frequenzbereiche verbessert. CSS ist besonders für große Reichweiten geeignet und ideal für LoRa. Mit dem "Spreading Factor" wird die Signaldauer entsprechend dem Endgeräteabstand zum Gateway geregelt. Damit wird auch der Energieaufwand gering gehalten. Ähnlich einer Unterhaltung auf einer lauten Party. Man spricht nicht nur lauter, sonder auch besonders langsam und deutlich. [uSF17]

Mit Spreading Factoren und Frequenzen werden Channels erzeugt. Channels können beliebig benutzt werden. Es gibt allerdings zwei Regeln zu beachtet:

1. Channels werden per Pseudozufallszahl geändert

2. Sendezeit erfüllt die regionalen Bestimmungen

Mit dem Aloha-Protokoll wird festgestellt wann gesendet werden soll. Vorhandene Daten werden dann einfach gesendet. 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 auswerten 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

Adapive Data Rate oder kurz ADR wird verwendet um die optimalste Senderate und die optimale Sendepower für das Endgerät zu finden und so die optimalste Datenübertragung zu erhalten. ADR kann nur verwendet werden wenn im FHDR Feld des LoRa-Pakets das ADR Bit gesetzt ist, siehe 3.3 Protokoll. Die Steuerung mit ADR findet durch den Netzwerkserver statt. Sobald der Netzwerkserver bereit ist setzt er das Bit im Downlink-Paket. Ist das Endgerät auch bereit, so setzt es ebenfalls das ADR Bit und ADR kann verwendet werden. Ist ADR nicht möglich wird Standardübertragung verwendet.

Die Steuerung findet durch spezielle MAC-Kommandos statt. Standardgemäß wird die höchste Übertragungsstärke verwendet, die geringste Übertragungsrate und zufälliger Kannal. Diese Parameter werden bei Bedarf vom Netzwerkserver durch das LinkADRReq MAC–Kommando angepasst. Die neuen Werte der Parameter sind in den Variablen des MAC-Kommandos codiert. Sobald die Werte geändert wurden, muss periodisch überprüft werden ob das Netzwerk die Nachrichten noch empfängt. Deshalb wird bei jedem Uplink der ADR_ACK_CNT Zähler erhöht. Wenn dieser Zähler den Schwellenwert ADR_ACK_LIMIT überschreitet, wird das ADRACKReq Bit im Uplink gesetzt. Dieses signalisiert dem Netzwerkserver eine Nachricht zu senden um die Verbindung zu bestätigen. Falls dieser Downlink nicht in ADR_ACK_DELAY Frames empfangen wird, wird zuerst die Übertragungsstärke auf das Maximum gesetzt. Gegebenenfalls wird außerdem die Datenrate verringert um die Reichweite zu erhöhen. Die Datenrate wird pro ADR_ACK_DELAY Frames schrittweise weiter bis zum minimaler Datetrate verringert. Falls diese schon minimal ist, werden alle Kanäle benutzt. Dies wird solange probiert bis eine Verbindung hergestellt werden kann. [SOR17, S.19 f]

4 LoRa Geräte Klassen

Um maximal Energie zu sparen, aber trotzdem die Endgeräte passend zur Anwendung Daten empfangen können, wurden Geräteklassen eingeführt. Das Hauptmerkmal der Klassen sind die unterschiedlichen Empfangsmodie. Es gibt drei Klassen: A, B und C. Klasse A ist der Standard und in jedem Endgerät verhanden. Klasse B und C sind optional und können vorhanden sein und werden als “Higher Class End-Devices “ bezeichnet. [SOR17, S.10]

4.1 Klasse A

Klasse A wird auch “All End-Devicec” genannt. Sie zeichnen sich durch den geringsten Stromverbrauch aus. Die Kommunikation wird vom Endgeräte gestartet. Werden keine keine Daten gesendet, schaltet sich das Endgerät in einen sehr sparsamen Schlafmodus. Es wird auf ein Hardbeat oder ähnliches verzichtet. Dadurch kann das Endgerät so lange schlafen bis ein vordefiniertes Ereignis auftritt.

Klasse A erlaubt außerdem, dass das Endgerät andere Protokolle verwendet solange es keine LoRa-Daten sendet oder empfängt. [SOR17, S.11 ff.] Das Endgerät startet die Kommunikation indem es Daten an das Gateway sendet. Daraufhin sendet das Gateway zweimal Daten zum Endgeräte. Die Empfangsfenster werden RX1 und RX2 genannt.

Die Empfangsfenster RX1 und RX2 müssen mindestens solange geöffnet bleiben, dass sie eine beginnende Übertragung feststellen können. Wird keine Übertragung empfangen, wird das Empfangsfenster geschlossen. Empfangsfenster RX1 wird nach RECIEV_DELAY1 Zeiteinheiten +/- 20msec nach Beendigung des Uplinks geöffnet. Es wird dieselbe Frequenz und Datenrate verwendet, die auch beim Downlink verwendet wurde. RX2 wird nach RECIEV_DELAY2 Zeiteinheiten +/- 20msec nach Beendigung des Uplinks geöffnet. Allerdings ist die Datenrate und Frequenz nun fest definiert. Nur durch spezieller MAC-Kommandos kann dies verändert werden. Wird bei RX1 festgestellt dass keine weiteren Daten mehr übertragen werden müssen, kann auf das öffnen des RX2 Fensters verzichtet werden.

Für die Join-Prozedur wird immer die Klasse A verwendet. Die verwendete Frequenz entspricht den Empfangsfenstern RX1 oder RX2.

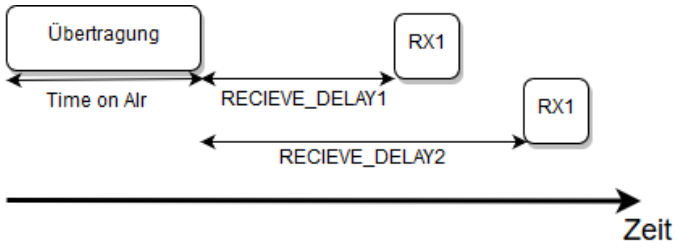

Abbildung 11: Sende/Empfangsmodell

4.2 Klasse B

Die Klasse B (B für BEACON) bietet bidirektionale Kommunikation mit einer deterministischen 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 periodischen Beacon ermittelt. Dieser Beacon wird regelmäßig vom Gateway gesendet und dient der Synchronisation der Endgeräte. Die Latenz ist einstellbar und kann bis zu 128 Sekunden dauern. [SOR17, 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, dass es sich in einer neuen Umgebung befindet. Dies ist der Fall , wenn es einen Beacon mit einer unbekannten ID empfängt. Durch eine Uplink teilt es dies dem Server mit. 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 auch die Klasse B für den Batteriebetrieb geeignet.

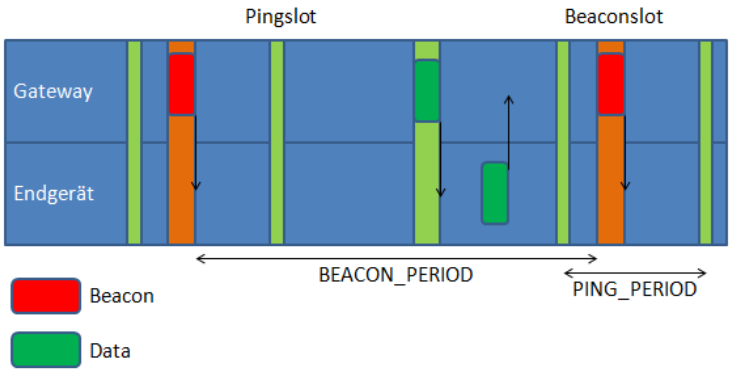

Abbildung 12: Sende/Empfangsmodell

4.2.1 Klassenwechsel A nach B

Um überhaupt einen Wechsel zu ermöglichen, muss der Netzwerkserver die Standard Pingslot Periode, die Pingslot Datenrate und den Pingslot Kanal kennen. Ale Endgeräte

treten in Klasse A dem Netzwerk bei. Der Wechsel in die Klasse B wird durch folgenden Prozess realisiert.

Als erstes muss das Endgerätes beim LoRaWAN-Layer anfragen, ob ein Wechsel möglich ist. Der Layer sucht nun nach einem Beacon. Wird ein Beacon entdeckt, wird die BEACON_LOCKED Serviceprimitive zurückgeliefert. Wird kein Beacon empfangen, wird die BEACON_NOT_FOUND Serviceprimitive 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 wird im Endgerätes das ClassB Bit im FCtrl Feld des Uplinks auf 1 gesetzt. Der MAC-Layer öffnet die Pingslots- und Beacons-Empfangsfenster. Dabei muss mit der größtmöglichen Abweichung der internen Uhr gerechnet werden und demensprechend die Empfangsfenster angepasst werden. Diese darf pro Beacon nicht mehr als +/- 1.3msec liegen. [SOR17, S.73]. Der Inhalt des empfangenen Beacon wird mit der Information zur Signalstärke dem Gateway gesendet. Damit kann z.B. die innere Uhr synchronisiert oder eine Ortswechsel festgestellt werden.

4.2.2 Betrieb

Mit dem PingSlotChannelReq Mac-Kommando teilt der Netzwerkserver dem Endgerät mitteilen, dass die Pingslot Frequenz und die Datenrate zu ändern ist. 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 mehr als 2 Stunden kein Beacon empfangen wurde, kann die Synchronisation mit dem Netzwerk verloren gehen. Dadurch funktioniert die Kommunikation in Klasse B nicht mehr. Die Kommunikation startet dann neu in Klasse A. Wechsel in Klasse B kann wie beschrieben versucht werden. Dieser Prozess kann beliebig wiederholt werden.

Wird kein Beacon empfangen, wird in den Beacon-Less Modus gewechselt und eine Kommunikation kann weiterhin erfolgen. Dieser Modus orientiert sich ausschließlich an der

internen Uhr. Um die Abweichung auszugleichen werden die Empfangsfenster erweitert. Das bedeutet einen höheren Energieverbrauch, aber auch eine höhere Wahrscheinlichkeit noch Daten zu empfangen.

4.2.3 Singel / Multicast

In Klasse B können die Nachrichten als Singelcast- oder als Multicast-Nachrichten ausgetauscht werden. Eine Singelcast Nachricht wird an das Endgerät, das im DevAddr 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. Dort sind dann definierte Endgeräte enthalten. LoRaWAN gibt keine Methode vor wie die Adressen und Schlüssel verteilt werden. Die Verteilung muss entsprechend in der Applikationsebene, sprich im Programm der Endgeräte bzw. des Applikationsserver, implementiert werden.

In Multicast-Nachrichten sind keine MAC-Kommandos erlaubt. Nur Daten dürfen als Multicastnachrichten übertragen werden. Dies wurde eingeführt, da Multicastnachrichten nicht dieselbe Robustheit und Sicherheit wie Singelcastnachrichten haben. Die Nachrichten werden nicht bestätigt, um hohen Datentransfer zu verhindern. [SOR17, S.84]

4.2.4 Beacon

Wie schon erwähnt werden der Beacons verwendet um das Endgerät mit dem Netzwerk zu synchronisieren. Deswegen werden die Beacon periodisch gesendet. Die Zeit zwischen zwei Beacon wir BEACKON_Period genannt. Die Endgeräte öffnen Empfangsfenster um diese Beacons zu empfangen. Ein Beacon zu übertragen dauert BEACON_RESERVED. Das Beacon-Empfangsfenster wird BEACKON_GUARD früher geöffnet um sicher zu stellen das ein Beacon auch wirklich erkannt werden kann. Außerdem wird die Beakon_GUARD benutzt um sicherzustellen dass kein Pingslot mehr geöffnet ist. Deswegen muss diese Beakon_GUARD mindestens so lang sein wie ein maximaler Pingslot. Das hat den Vorteil, dass nicht darauf geachtet werden muss, ob ein Pingslot noch geöffnet ist. Vergleichbar ist dies mit der Distributed Coordination bei WLAN Anwendungen.

Während Beacon Empfang, kann kein Pingslot geöffnet werden.

Um gleichzeitiges Senden mehrerer Endgeräte nach Abschluss eines Beacon zu vermeiden, werden zufällige Wartezeiten und zufällige Pingslotperiode verwendet. Dadurch wird die Wahrscheinlichkeit einer Kollision verringert.

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 und 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$\hat{3}$2 enthalten. Das CRC Feld wird verwendet um die Korrektheit des Zeit- und des RFU-Feldes zu bestätigen. 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. Das GwSpezific Feld lässt sich unterteilen in InfoDesc und Info Felder. Das InfoDesc gibt an auf was sich das Infofeld bezieht. 0 gibt an, das 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 unbenützt. Die Koordinaten, im Info Feld enthalten 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. Falls nötig kann die Frequenz mit dem MAC-Kommando PingSlotChannelReq geändert werden.

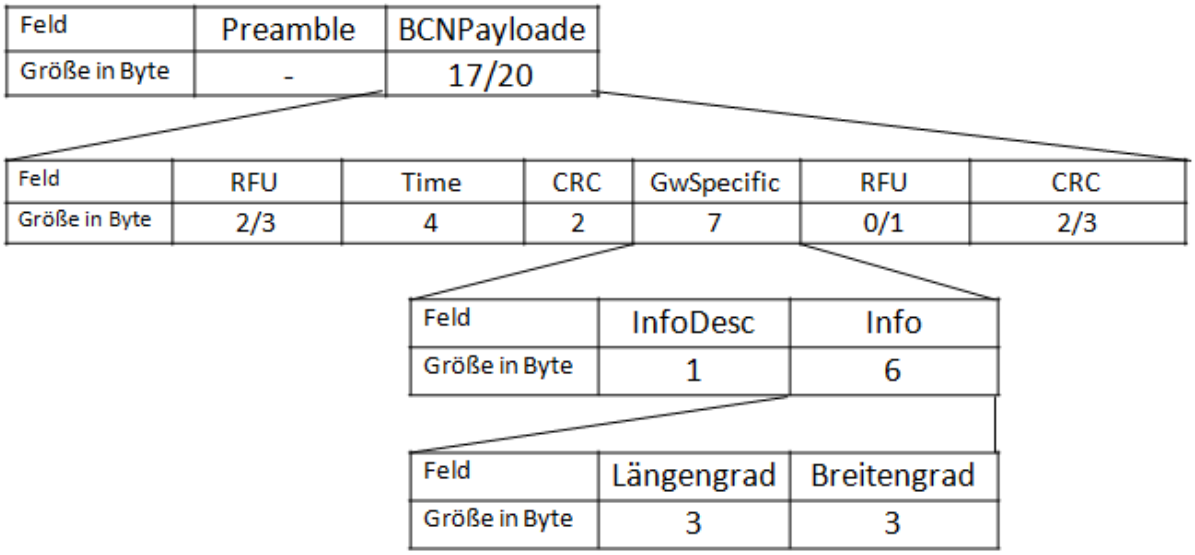

Abbildung 13: Aufbau eines Beacons

4.3 Klasse C

C steht für "CONTINUOUSLY Listening". Wie der Name schon sagt, ist hier dauernd ein Empfangsfenster geöffnet. Dafür wird fast latenzfrei übertragen. Dies bedeutet aber auch erhöhter Stromverbrauch. Klasse C ist somit nicht für den Batteriebetrieb geeignet.

Geräte, die Klasse C implementieren haben, sollen Klasse B nicht verwenden um Fehler zu vermeiden.

Diese Klasse verwendet die gleichen Empfangsfenster und die gleichen Frequenzen wie Klasse A. Der große Unterscheid besteht allerdings darin das RX2 immer dann geöffnet ist, wenn nicht gerade Daten gesendet werden oder RX1 geöffnet ist.

Auch in Klasse C können Multicastnachrichten gesendet werden. Hierbei gelten die gleichen Regeln wie bei Klasse B.

5 Sicherheit

Sicherheit in netzwerkfähigen Systemen ist ein sehr wichtiges und heiß disktiertes Thema. Da alle LoRa-Geräte drahtlos Daten übertragen, sind diese für alle empfangbar. Deshalb ist eine Verschlüsselung wichtig. Genauso wichtig ist es die Daten zu athentifiziern und somit zu überprüfen wer die Daten gesendet hat. Dritte können sogenannte Replayattack

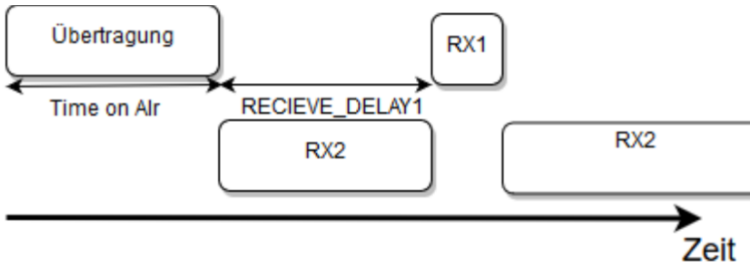

Abbildung 14: Sende/Empfangsmodell

durchführen und so versuchen die Kommunikation zu stören oder Fehlinformationen zu senden. Deswegen werden Zähler oder Nonce mit den Daten mitgeschickt. Im folgenden Text wird betrachtet wie LoRa diese Sicherheitsfeatures implementiert hat.

5.1 Schlüssel

Schon der Beitritt eines Endgerätes erfolgt verschlüsselt. Dafür werden zwei verschiedene Schlüssel verwendet. Die Join-Nachricht wird mit dem JSIntKey “Joining Session Encryption Key” verschlüsselt. Der Schlüssel wird mittels aes128 Verschlüsslung generiert. Zur Generierung wird NwkKey mit einem 16Byte langem Wort verschlüsselt. Das Wort beginnt mit der Konstante 0x06, danach folgt die DevEUI und zum Schluss wird mit Nullen 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. Mit Hilfe des JSIntKey wird außerdem der MIC-Code in der Join-Nachricht berechnet.

Alle MAC-Kommandos die an Port 0 gesendet werden, werden separat verschlüsselt. Hierzu wird der Network session encryption key “NwkSEncKey” verwendet. Durch die Verwendung mehrerer, verschiedener Schlüssel steigt die Sicherheit. Der NwkSEncKey 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: 3.2.1 OTAA.

Um die Integrität der Daten zu bestimmen werden zwei Schlüssel verwendet. Der FNwkSIntKey “der lang Forwarding Network Session Integrity key” wird benutzt um den MIC-Code für die Integritätbestimmung abzuleiten. Auch dieser Schlüssel wird, wie

JSIntKey abgeleitet. Jedoch besteht das Schüsselwort besteht aus 0x01, der JoinNonce , der JoinEUI und DevNonce. Der SNwkSIntKey “Serving Network session integrity key” stellt das Gegenstück dar und wird verwendet um den MIC-Code zu verifizieren. Hier besteht das Schlüsselwort aus 0x03, der JoinNonce, der JoinEUI und der DevNonce.

JSEncKey, JSIntKey, FNwkSIntKey, SNwkSIntKey, NwkSEncKey sind für jedes Entgerät unterschiedlich.

Auch die Nutzdaten werden verschlüsselt. Dies geschieht unter Verwunderung des AppSKeys. Der “Applikaction Session Key” wird vom Applikationsserver und dem Endgerät verwendet. Dieser Schlüssel wir vom Programmierer vorgegeben, da er mit dem Schlüssel des Applikationsserver übereinstimmen muss. [SOR17, S.50 ff]

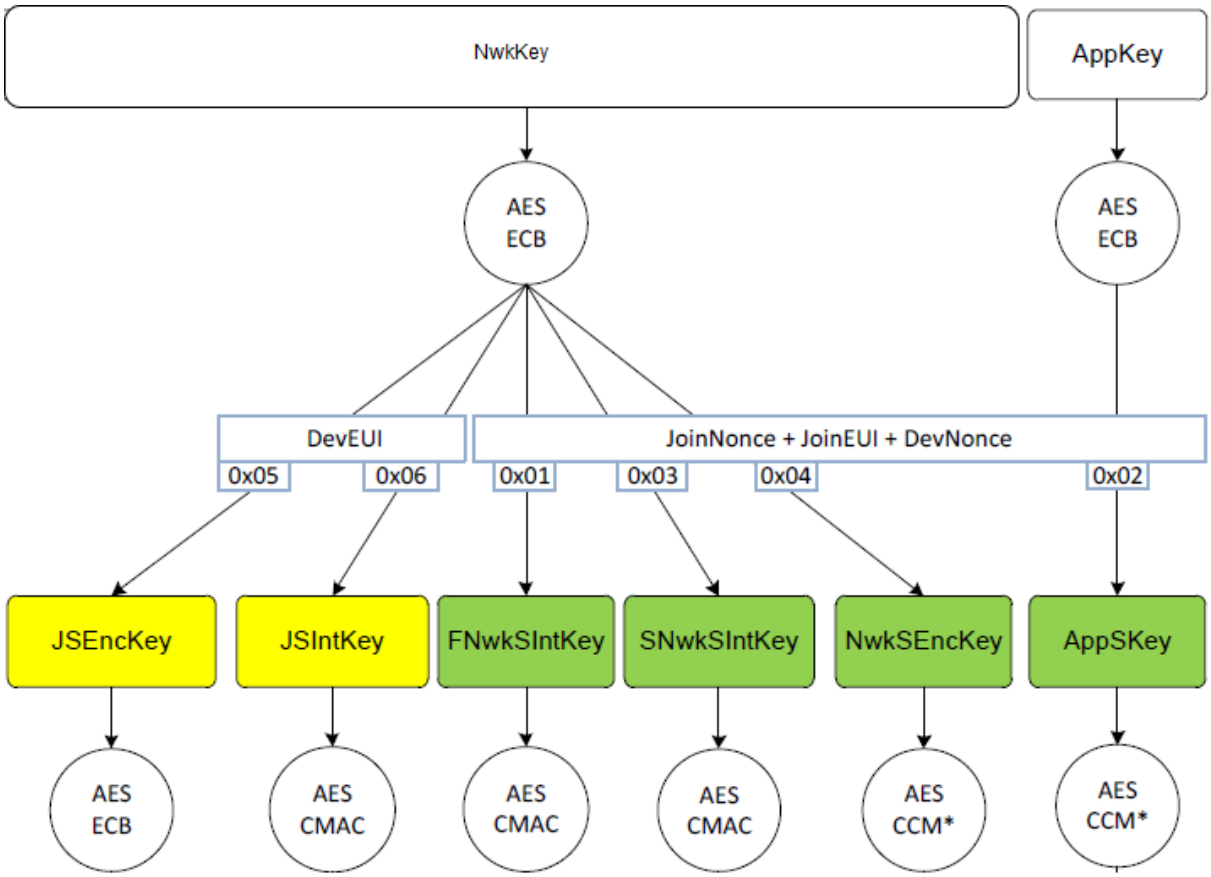

Abbildung 15: Schlüselableitung [SOR17, S. 63]

5.2 Zähler

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

Der FCntUp Counter wird für jeden Uplink erhöht. Der Stand wird in das FCnt Feld der Nachricht eingefügt. Da das Gatway den selben Zähler hat, kann es erkennen, ob die Nachricht schon einmal gesendet wurde.

Genau so wird mit den Downlink Counter verfahren. Im NFCntDown Zähler werden die MAC-Kommando Nachrichten gezählt. AFCntDown wird für alle anderen Nachrichten verwendet. [GAS17, S.22 ff]

6 Ausblick

Ich sehe ein große Zukunft für IoT-Anwendungen.

Der Bedarf an vernetzten batteriebetrieben Geräten zum Datenaustausch wird stetig steigen. Im Gegensatz zum hohen Investitionsaufwand beim 5G Netz sind die Kosten bei LPWAN gering. Besonders für Übertragung von Sensordaten sehe ich viele Möglichkeiten. Nicht nur solche, die unseren Komfort steigern, sondern auch Anwendungen z.B. im Umweltschutz und in der Landwirtschaft.

Ich könnte mir ein Netz von Sensoren zur frühzeitigen Hochwasserwarnung vorstellen. Aber auch Sensoren die in Gewächshäusern und auf Feldern die Bodenfeuchtigkeit, Sonneneinstarhlung und Temperatur (Frostwarner) messen und übermitteln und so schnelles Handeln ermöglichen.

Ich habe durch meine Seminararbeit großes Interresse an diesem Themengebiet gefunden und bin schon sehr gespannt wie sich dies weiter entwickelt.

Literatur

[CBHF17] Cheong, Phui San, Johan Bergs, Chris Hawinkel und Jeroen Famaey: *Comparison of LoRaWAN Classes and their Power Consumption*. 2017 IEEE Symposium on Communications and Vehicular Technology (SCVT), 2017.

[GAS17] GEMALTO, ACTILITY und SEMTECH: *LoRaWAN^{TM} SECURITYA WHITE PAPER PREPARED FOR THE LoRa ALLIANCE^{TM}*. 2017.

[SOR17] SORNIN, N.: *LoRaWAN^{TM} 1.1 Specification*. 2017.

[uSF17] Sveinn Finnsson, Kristoffer Olsson und: *Exploring LoRa and LoRaWAN*. 2017.

[Wor18] Workgroup, Technical Marketing: *LoRaWAN^{TM} Backend Interfaces 1.0 Specification*. 2018.

[YEG17] YEGIN, A.: *LoRaWAN^{TM} Backend Interfaces 1.0 Specification*. 2017.