

Seminararbeit: Lorawan

Tobias Sigmann

23. Mai 2019

Inhaltsverzeichnis

1	Einführung in Lora	4
2	Aufbau eines Lora-Netzwerk	5
2.1	Gateway	5
2.2	Netzwerkserver	6
2.3	Join-Server	6
2.4	End-Gerät	7
3	LoraWan Funktionsweise	7
3.1	Schichtenmodell	8
3.2	Netzwerkbeitritt	9
3.2.1	OTAA	9
3.2.2	ABP	10
3.3	Protokoll	11
3.3.1	MAC-Kommandos	11
3.3.2	LoRa-Paketstruktur	11
3.4	Übertragungsart	14
3.4.1	Adaptive Data Rate	16
4	Lora Geräte Klassen	17
4.1	Klasse A	18
4.2	Klasse B	20
4.2.1	Klassenwechsel A nach B	21
4.2.2	Betrieb	22
4.2.3	Singel / Multicast	23
4.2.4	Beacon	24
4.3	Klasse C	25
4.3.1	Wechsel von A nach C	26
5	Sicherheit	26
6	Live-Beispiel	28

1 Einführung in Lora

Lora ist ein Low Power, Wide Area (LPWA) Netzwerkprotokoll und somit sehr gut für batteriebetriebene kabellose Geräte geeignet. Deswegen wird Lora auch oft im Internet of Things (IoT) Bereich verwendet. Mittels der bidirektionalen Kommunikation ist es möglich Daten und Befehle über weite Strecken zu übertragen. Leider leidet darunter die Geschwindigkeit, sodass sich Lora nicht als WLAN Ersatz eignet. Trotzdem können zwischen 0.3 und 50 kbps erreicht werden. In Europa werden 863 MHz bis 870 MHz verwendet. Allerdings variiert der Frequenzbereich für andere Kontinente. Je nach Bedingungen können so bis zu 20km entfernte Endgeräte erkannt und mit diesen kommuniziert werden. Es ist sogar möglich den Standort des Gerätes zu bestimmen.

Eine Alternative zu Lora ist Sigfox, hierauf werde ich nicht weiter eingehen. LoRaWAN 1.1

[Tec15](Optimiert für Batterie Kapazität(Teilnehmer) Reichweite, Kosten mehrjährige Batterielaufzeit, kleine Datenmengen, große Reichweite, LPWAN (Low Power WAN)

Kriterien für Lora: Netzwerk Architektur, Reichweite, Batterielaufzeit, Interferenzrobustheit, Anzahl Knoten, Sicherheit, bidirektionale Kommunikation, verschiedene Anwendungsunterstützung

Orientiert für Mobile Adressierbare Endgeräte)

[AVTP⁺17](alternativen: Sigfox, Ingenu, Dash7

Klassen Kompromiss zwischen Reichweite, Performance(Latenz/ Durchsatz) und Energiebedarf

Energiesparend durch ADR (Adaptive Daten Rate))

Es wird folgen: Was ist lora, wo und wofür wird es benutzt, wie weit kann man senden und wie schnell...

2 Aufbau eines Lora-Netzwerk

Lora wird auch Deswegen gerne für IoT-Geräte verwendet, weil der Netzwerkaufbau ermöglicht die über Lora verwendeten Daten im Internet abzurufen und so ohne weiteres das Gerät mit dem Internet zu verbinden. Um die von den End-Geräten gesendeten LoRa-Pakete auf IP/TCP Pakete umzusetzen wird ein Gateway benötigt, das auf der einen Seite LoRa-Pakete empfängt/sendet und auf der anderen Seite TCP/IP Pakete verwendet. Das Gateway implementiert aber keinerlei Logik. Hierzu ist ein Netzwerkservers zuständig der durch die Gateways das Netzwerk kontrolliert und steuert. Gleichzeitig stellt er die Verbindung zu einem Applikationsserver her, in dem er die vom Gateway empfangenen Daten Weiterleitet.

Der Applikationsserver ist zuständig den die gesendete Nachrichten zu verarbeiten und gegebenenfalls selbst welche an die Endgeräte zu senden.

Diese Architektur wurde gewählt um die Laufzeit der Akku betriebenen Endgeräte, Anzahl der Endgeräte, Qualität Signals und Sicherheit des Netzwerkes möglichst hoch zu halten. [Tec15, S. 8 ff.]

2.1 Gateway

Das Teilnetz das aus dem Gateway und mehreren LoRa-Endgeräten besteht ist Sternförmig aufbau. Jedes Endgeräten kommuniziert direkt mit dem Gateway. Diese Art der Kommunikation wird auch “Single-Hop-Connection” zu Deutsch (Einfacher-Sprung-Verbindung) 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 ist zuständig die Pakete auf Duplikate zu überprüfen und nur einmalig an die Applikationsservers zu senden. Ein weiterer Vorteil ist das kein Übergabe der Endgeräte bei Standortwechsel zu andern Gateways nötig ist. Dadurch müssen die Gateways mit vielen Endgeräten kommuniziert. Um diese hohe Endgeräteanzahl zu ermöglichen wurde darauf verzichtet mit jedem Endgerät einzelne zu kommunizieren und stattdessen auf eine Parallele Kom-

munikation gesetzt. Hierzu werden adaptive Datenraten und Mehrkanal-Multi-Modem-Transceiver verwendet.

Durch die genannten 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 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 das Steuern des LoRa-Teils des Netzwerkes. Der Server verwaltet jedes Endgerät separat indem es mit ihm den zu verwendenden Funkkanal Aushandelt und die Datenrate kontrolliert wenn ADR(Adaptiv Data Rate) verwendet wird. Außerdem ist er bei dem Netzwerkbeitritt eines Endgerätes .beteiligt.

Weiterhin überprüft er die empfangen Pakete auf ihre Korrektheit, Integrität und filtert Duplikate, die durch das Empfangen der gleichen Übertragung von einem Endgerät an verschiedenen Gateways, verursacht wurden. Dabei ermittelt er auch die Gateways, die den besten empfang zu den jeweiligen Endgeräten hat und nutzt dieses um Daten an die Endgeräte zu senden.

Es ist nicht immer möglich Daten direkt zu senden, da die Endgeräte nur manchmal empfangsbereit sind. Um die Applikationsserver zu entlasten, puffert der Netzwerkserver die Daten und sendet diese zum nächst möglichem Zeitpunkt.

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

2.3 Join-Server

Der Server kann mit mehreren Netzwerkservern verbunden werden und jeder Netzwerkserver kann mehrere Join-Server haben.

Ein Join-Server wird benötigt um den Beitritt mittels OTAA zu ermöglichen. Mehr zu OTAA kann in dem Kappitel OTAA gelesen werden. Wenn ein Endgerät dem Netzwerk beitreten möchte, leitend der Netzwerkserver die Anfragen an den Join-Server weiter. Dieser führt dann die nötige schritte des Beitritts aus wie z.B. ableiten von Schlüsseln oder Senden der nötigen Einstellungen. Um dies zu tun muss ihm der NwkKey und der AppKey bekannt sein, da diese zum verschlüsseln der Nachrichten verwendet werden aber aus Sicherheitsgründen nie über das Netzt übertragen werden dürfen. [SOR17a, S. 9 f.]

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 speichern.

- 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 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 da sie nie übertragen werden.

[SOR17b, 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.

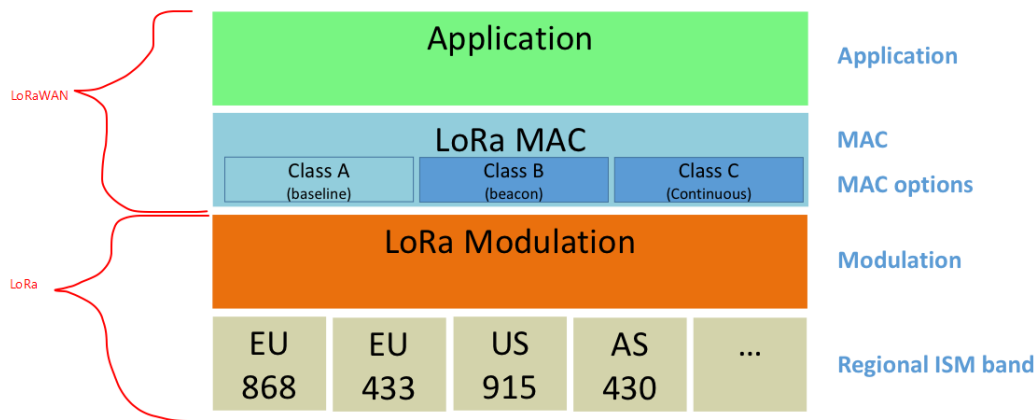


Abbildung 1: LoRaStack [Tec15, S.7]

3.1 Schichtenmodell

Das Schichtenmodell lässt sich in zwei Teile unterteilen. Der LoRa Teil ist der unterste und kümmert sich um die physikalische Übertragung der Pakete und er LoRaWAN Teil des Modells ist für die Steuerung des Netzwerkes, Implementierung der LoRaWAN-Klassen und das überprüfen / verschlüsseln der Daten zuständig.

Die unterste Schicht des LoRa Teils ist für die Anwendung der Richtigen Frequenzen zuständig. In Europa muss das ISM-Band 868 verwendet werden in den Vereinigten Staaten wird das Band 915 verwendet. [Tec15, S.7]

Die darüber liegende Schicht heißt LoRa Modulation und kümmert sich darum dass die Pakete so in die Frequenz "moduliert" werden, dass der Empfänger diese korrekt und effizient empfangen und wiederherstellen kann. Mehr dazu im Kapitel Übertragungsart

Über der LoRa Modulation Schicht liegt die erste LoRaWan Schicht, LoRa MAC. AC steht für "Media Access Protokoll". Dieses Protokoll wird verwendet um das Loranetz zu steuern und wie der Name schon sagt, Daten zu übertragen. Diese Schicht ist außerdem für die Implementierung der einzelnen Endgeräteklassen und für das Übertragen der Steuerungskommandos zuständig. Mehr zu den Klassen kann im Kapitel [namerefsec:klassen](#) und im Kapitel Protokoll gelesen werden.

Die oberste Schicht nennt sich Applikationsschicht und ist dafür zuständig

die Nutzdaten einer Nachricht passend zu verpacken, zu verschlüsseln und zu authentifizieren.

3.2 Netzwerkbeitritt

End-Geräte sind immer bestimmten Netzwerken zugeordnet. Es gibt zwei Wege um ein neue End-geräte zu einem bestehenden Netzwerk hinzuzufügen.

3.2.1 OTAA

Die sicherste aber auch aufwendigste Methode um ein End-Gerät mit einem Netzwerk zu verbinden heißt OTAA “Over-the-Air Activation”. Hierbei muss jedes Mal wenn einem Netzwerk beigetreten werden soll die Join-Prozedur ausgeführt werden. Hierfür müssen folgende 4 Konstanten Vorgegeben werden.

DevEUI, JionEUI , NwkKey, AppKey. erklären

Näheres zu dem DevEUI und JoinEUI kann im Kapitel ?? gefunden werden. der

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

Der letzte Wert heißt APPSKY und sichert die Kommunikation vom End-Gerät zu dem Applikationsserver ab. Der Schlüssel wird genau wie der NWKSKY vom Netzwerkservers erzeugt und verwaltet. [GAS17, S.3]

Als erstes muss das End-Gerät eine Join- oder Rejoin-Nachricht senden. Die Nachricht besteht aus der JoinEUI, dem DevEUI und einer DevNonce. Mit der DevNonce sollen Replayattack verhindert werden. Diese Nonce ist das beim ersten Join-Request 0 und sollte sich bei jedem Join-Request erhöhen. Unter anderem deswegen muss sie auch dann noch gespeichert werden wenn kein Strom zur Verfügung steht um die Nonce nicht nach jedem ausschalten zurückzusetzen. Falls von dem gleichen Endgerät eine Join-Request mit einer zu kleinen DevNonce gesendet wird, wird die Nachricht ignoriert und es ist nicht möglich dem Netzwerk beizutreten.

Die Accept Nachricht besteht aus einer JoinNonce, einem NetzwerkID Net_ID, einer Geräteadresse DevAddr, einem Einstellungsfeld DLSettings , einer Zeit-

angabe wie lange zukünftig auf eine Antwort nach dem senden gewartet werden muss, hier RxDelay und einer optionalen Liste an Netzwerkparameter CFList.

Die JoinNonce wird verwendet um Replayatacken zu verhindern und muss größer sein als die zuletzt gesendete um von dem Endgerät verarbeitet zu werden. Außerdem wird die Nonce benutzt um Schlüssel wie AppSKey herzuleiten. Für jedes Endgerät wird eine eigene Join Nonce geführt, sie sollte sich nicht wiederholen. Jedes Endgerät merkt sich die letzte JoinNonce und tritt auch nur bei wenn diese größer ist als die letzte empfangene.

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

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

erklären

3.2.2 ABP

Die einfachste Art des Beitritts heißt ABP was für "Activation by Personalization" zu Deutsch "Aktivierung durch Personalisierung" steht. Hierbei muss lediglich vor Inbetriebnahme des End-Gerätes 3 Konstanten definiert werden. Manche Hersteller "brennen" diese drei Werte fest in den Chip ein, sodass er nicht geändert werden kann. Falls es nicht möglich ist dem Hersteller die gewünschten Werte zukommen zu lassen, sind solche End-Geräte nur schlecht bis gar nicht für den Beitritt mittels ABP geeignet.

Als erstes muss die DevAddr(Geräteadresse) angegeben werden. Diese Adresse existiert nur einmal im Netzwerk und wird verwendet um das Endgerät zu identifizieren. Die Adresse wird vom Netzwerkserver erzeugt und muss manuell von dort kopiert werden.

Mit Hilfe dieser 3 Werte kann die Join-Prozedur übersprungen werden. Daher kann das Endgerät direkt einem LoRa-Netz beitreten wenn es angeschaltet wird und muss nicht erst alle Schlüssel neu ableiten und aushandeln. Allerdings ist diese Methode deswegen weniger sicher, da immer dieselben Schlüssel ver-

wendet werden.

Nach Beitritt muss das ResetInd Mac Kommando im FOpt Feld gesendet werden gesendet werden solange bis ein ResetConf Kommando erhalten wird. Nun ist das Gerät im Netzwerk und kann unter der eingestellten Adresse und mit dem eigestellten Schlüssel arbeiten. [SOR17b, S. 64]

3.3 Protokoll

Damit der Netzwerkservers das LoRa-Netz zu steuern kann wurden, wurden Mac-Kommandos eingesetzt. Mit diesen Kommandos lassen sich wie schon gesehen, dem Netzwerk beitreten, mit dem Endgerät kommunizieren und Frequenzen, Kanäle und vieles mehr zu steuern. 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 das vom Endgerät, das Bildlich gesprochen “unter“ dem Gateway sitzt, an das Gateway gesendet wird. Ein Downlink ist dem entspricht ein Paket das vom Gateway an den Servers gesendet wird.

3.3.1 MAC-Kommandos

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 längenangebe angibt. Diese sind somit feste zuvor definierte. Im expliziten Modus werden noch 2 Felder hinzugefügt, PHDR und PHDR_CRC. Somit sieht ein expli-

zitesten 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 dass 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 Usag" zu Deutsch "für zukünftige verwendung reservier". 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 werde 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 LoRa-WAN 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. Hamit 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 FOptsLen Feld wird die Länge des FOpts Feldes mitsamt des Headers eingetragen Wenn Das FOptsLen 0 ist, 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 Netzwerks server 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 ADRACKReq 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 LoRa-WA 1.1 Specification [SOR17b] gefunden werden.

Tabellen

einfü-

gen,

bele-

gen

maccomandos

einf-

gen

zuer-

set

3.4 Übertragungsart

Das LoRaWAN Protokoll ist optimiert für Batteriebetriebenen Endgeräte die drahtlos kommunizieren möchten. Um energieeffizient zu sein setzt LoRa hauptsächlich auf zwei Punkte. Die Modulationstechnik und eine Adaptive Datenrate (ADR). Auch die Öne-Hoparchitektur trägt zur energieeffizienz bei. Die Art wie LoRa signale Moduliert wird in kapittel Übertragungsart besprochen [CB⁺17](Um energieeffizient zu sein setzt LoRa hauptsächlich auf zwei Punkte. Die Modulationstechnik und eine Adaptive Dtanenrate (ADR)) //—

— Um die Enstandenen Pakete in Signale umzusezenun didese effizient und gleichzeitig übertragen zu können nutzt LoRa Chirp Spread Spectrum (CSS). Hierbei werden die Frequenz über eine gewisse Zeit hinweg verändert. Durch erkennen in welche richtig, ansetigen oder Abfalleb, die frequenz verändert wird, können 1 und 0 Codiert werden. Man spricht bei einem Bit von einem Chrip-Impuls. Durch aneinanderreihe der verschiedenen impulse ist es möglich mehere Bits nacheinander zu übertragen. Das entstandene Signal wird auch al Sub-Chrip bezeichnet. Durch verwenden von Unterschiedlichen anstiegs und

abfallszeiten ist es möglich mehrere Signale auf der Selben frequenz zu übertragen ohne das die Signale sich gegenseitig stören. Dies nennt man Spread Factor. Auserdem kann die Paralelität durch verschiedene Frequenzbereiche verbessert werden. CSS ist besonders für große reichweiten geeignet und somit nachprüfen auch bestens für Lora. Am besten ist das Signal wenn das endgerät nahe am Gateway ist. Je weiter es entfernt desto schlechter wird das sgnal. Um die kommunikation trotzdem zu ermöglichen wird der spreading Factor erhöht. Dies hat auch den Vorteil dass der Energieaufwand gering gehalten wird. Analog wie Menschen auf einer Party nicht immer versuchen lauter zu sprechen, sonder auch versuchen besonders langsam und deutlich zu sprechehn.

Mann sichbrcht auch von Channels. Channels können beliebig benutzt werde. es gibt alerdings zwei regeln zu beachtn.

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 der selbern Frequenz senden möchten kommt es zu einer kollision. Dadurch kann das Gateway die empfangenen Dta-ne nicht mehr auswerden. Deswegen warten beide Endgeräte eine zufällige, unterschiedliche Zeit abe bist sie erneut senden.

[SOR17b](Knoten können zu jeder Zeit, auf beliebigen Kanälen, beliebig schnell, beliebig lange senden, solange folgende regeln befolgt werden.

- Channels werden per Pseudozufallszahl geändert
- Sendezeit erfüllt die Regionalen Bestimmungen

) [SOR17b](Geschwindigkeit ist kompromiss zwischen abstand/geschw. die untersch freuenzen bze. geschwindigkeiten beeinflussen sich nicht gegeseitig => keine inteferenz Die Datenrate ist einstellbar, jedoch wird die Reichweite bei höherer Datenrate gemindert. Ein Vorteil von Lora ist, das die einzelnen Datenraten nicht interferieren und so jedes Endgerät seine eigene Datenrate unabhängig von den anderen Verwenden kann. Außerdem wird die Datenrate

und die Sendeleistung für jedes Gerät separat gesteuert (ADR, Adaptive Data Rate))

[AVTP⁺17](Chirp Signal => Zeitliche Änderung in Trägerfrequenz(höhere Frequenz als Datenrate)(positiv chirp/negativ chirp)

Datensignal wird in Chirp Signal moduliert. Resultierende Signal ist breitbandiger als Datensignal. Maximale Datenrate auch mit Rauschen erreichbar.

Durch orthogonale SSspread Factor"mehrere Signale auf einem Chanel) [Tec15](normal FSK, schon sehr effizient. Lora "chirp spread spectrum modulation". Ist wie FSK aber größere Reichweite, robuster. Stammt aus dem Militär/raumfahrt.Lora als erstes für kommerziellen billigen Einsatz.

Spread spectrum => signale sind Orthogonal für versch. spreizraten, faktoriell mit datenrate => verschiedene Datenraten auf einem Kanal

Nähere Geräte sind schneller => höhere Datenrate => kürzere Übertragungsdauer und lassen somit mehr zeit für andere, => bessere Batterielaufzeit. Deswegen sind symmetrische up/downlinks nötig.) Frequenzhopping, spread spectrum, code-channels

soll ich

ver-

3.4.1 Adaptive Data Rate

hält-

Adaptive Data Rate oder kurz ADR wird verwendet um immer die optimale 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 LoraPaketes 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. 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 Applikationslayer gesteuert werden.

niss

zwi-

schen

data-

rat sf

und

ener-

gie

rein-

Die Steuerung findet durch spezielle MAC kommandos statt. Standardgemäß wird die höchste Übertragungsstärke verwendet allerdings auch die geringste Übertragungsrate. Falls diese gedrosselt werden soll wird vom Netzwerkservers das **LinkADRReq** MAC command benutzt. Mit diesem wird das Endgerät

ma-

chen?

warum?

informiert, dass es die data rate, transmit power, repetition rate or channel ändern soll. Was auf welchen wert geändert werden soll wird in die Parameter codiert. Sobald die Werte geändert wurden, muss periodisch überprüft werden ob das Netzwerk die Nachrichten noch bekommt. Deswegen wird jedes mal wenn der uplinkframecount erhöht wird, wird der ADR_ACK_CNT counter verwendet. Wenn dieser counter ein gewissen schwellenwert (ADR_ACK_Limit) überschreitet, wird das ADRACKReq bit gesetzt. Dieses signalisiert den Netzwerksverer das er mit einem Uplink ein Downlink senden muss um die Verbindung zu bestätigen. Falls dieser Downlink nicht in ADR_ACK_Delay frames empfangen wird, wird zuerst die übertragungsstärke auf max gesetzt. Falls möglich wird ausserdem die Datenrate verringert um die Reichweite zu erhöhen. Die Datenrate wird solange weiter jede ADR_ACK_Delay frames verringert bis diese minimal ist. Falls diese schon minimal ist müssen alle channels wiederverwendet werden. Dies wird solange probiert bis eine verbindung hergestellt werden kann.

[AVTP⁺17](Datenrate und Funkfrequenz(RF) werde passend zum Abstand angepasst

nahe Knoten => hohe Datenrate => kurze Sendezeit => weniger RF-Power kann nach Bedarf geändert werden

=> immer möglichst schnelle senden => weniger Energie

) [CB⁺17](crip signal)

4 Lora Geräte Klassen

Um maximal energie zu sparen aber trotzdem die möglichkeit dass die endgeräte agiel 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 A können werden als "high class Joinen End-Devices" genannt. nur in

[SOR17b](Geräte müssen mindestens A können, alle die mehr können wer- A be-
schrie-
ben?
wohin
mit
coun-

den auch "high class End-Devices" genannt) Vielleicht zu klein => in anderes Kapitel stopfen. Bei mehrfacher Übertragung wird nicht erhöht

Die Endgeräte sind je nach Kommunikationsart/Protokoll Art in drei Klassen (A, B und C) unterteilt.

Jede Klasse hat 3 Counter FCntUP(Pro uplink ++), FCNTDown(pro downlink außer port 0 => mach), AFCntDown(port ungleich 0 dann ++) (nur beschreiben wie diese grob funktionieren) Zähler sollen nicht flüchtig sein (Batterie wechseln kein reset) bei neuverbinden müssen alle Counter auf 0 gesetzt werden. Counter müssen auf beiden Seiten gleich gehalten werden (Synchron geführt) Wenn Nachricht empfangen ist muss der darin enthaltene Counter größer sein als der eigene.

die Counter Werte sollen so weit wie möglich nur einmal verwendet werden.

) [Tec15] (Asynchrone Knoten wegen Batterie => Event/Scheduler gesteuert verwendet ALOHA

Normal Netze müssen sich synchronisieren und Nachrichten abrufen. Lora partiell nicht => laut GSMA 3 bis 5-fach effizienter)

zur besseren Anpassung/ Anpassung an Batterie

EU: 10 Kanäle (8: 250bps bis 5.5kbps) (1: FSK 50kbps) (high rate Lora 114kbps)
)

4.1 Klasse A

Klasse A wird auch (All end-Device) genannt und zeichnet sich durch sehr geringen Stromverbrauch aus. Die Kommunikation kann bidirektional stattfinden, allerdings muss die Kommunikation von dem Endgerät gestartet werden. Das bietet die Möglichkeit, dass das Endgerät, wenn keine Daten gesendet werden müssen, in einen sehr sparsamen Schlafmodus wechselt. Um das Endgerät nicht zum Aufwachen zwingen zu müssen, wurde auf einen Hardbeat oder ähnliches verzichtet. Dadurch kann das Endgerät so lange "schlafen" wie es möchte. Somit ist die Klasse A auch die potenziell Stromsparende Endgeräteklasse. Die Klasse A erlaubt außerdem, dass das Endgerät andere Protokolle schickt

solange es keine LoRa Daten sendet oder empfängt.

Das Endgerät startet die Kommunikation in dem es Daten an das Gateway sendet(uplink). Daraufhin hat das Gateway die Möglichkeit 2 mal Daten zum Endgeräte senden(downlink). Die Downlinkfenster werden RX1 und RX2 genannt. Da die Kommunikation asynchron stattfindet, muss das Endgerät warten bis die uplinkphase abgeschlossen ist.

Die empfangsfenster RX1 und RX2 müssen mindestens solange geöffnet bleiben das sie eine beginnende Übertragung feststellen können. Falls keine Übertragung empfangen wird, wird das Fenster wieder geschlossen. Anderenfalls werden die Daten empfangen. Das Empfangsfenster RX1 wird nach RECIEV_DELAY1 zeiteinheiten +/- 20msec nach beendigung des Uplinks geöffnet. Es wird die selbe Frequenz und Datenrate verwendet die auch bei den uplink 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 nach RECIEV_DELAY2 zeiteinheiten +/- 20msec nach beendigung des Uplinks geöffnet. Allerdings ist die Datenrate und Frequenz fest. Nur Mittels spezieller MAC commands kann dies verändert werden.

Für alle join / rejoin Aktivitäten wird immer die Klasse A verwendet . Schon

[SOR17b](radio packet explicit mode, vom Gateway(1) zum Knoten(1), erklärt ausgelöst vom Netzwerkservers, auch Multikasts möglich, (Preamble, PHDR, oder PHDR_CRC, PHYPayload) Um Nachricht kurz zu halten kein CRC am Ende, woan- nach Receiver_Delay1 / Receiver_Delay2 kann empfangen werden (rx1, rx2) ders

Fenster müssen lange genug für Preamble auf bleiben=> wenn erkannt wird empfangen wenn nicht Fenster wieder zu. Es darf nur gesendet werden wenn beide Fenster zu sind. ==>Es ist auch erlaubt andere Protokolle zu sprechen wenn nicht gesendet oder gehört wird.<==) [SOR17b](Frequenz abhängig von Uplinkfrequenz, Datenrate abhängig von Uplinkdatenrate, wird nach Receiver_Delay 1 +/- 20 msec erwartet, Datenrate auch abhängig von Regionalen regeln, Standard: Datenrate = Uplinkdatenrate) [SOR17b](feste Frequenz/Datenrate, nach Delay2 +/- 20 msec, Frequenz/Datenrate mittels MAC änderbar) [SOR17b](Öffnungslänge muss für Preamble ausreichen, nach

RX1 + MIC und Authentizitätscheck muss nicht zwingen RX2 geöffnet werden, Sender muss in einem der beiden Fenster stattfinden, Falls Downlink über beide Fenster => Frames müssen gleich sein. Knoten dürfen nicht während empfangen/ zwischen RX1 und RX2 senden, andere Protokolle dürfen gesprochen werden wenn gesendet werden darf)

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 Synchron ablaufen. Außerdem muss festgestellt werden, ob das Endgerät bzw. das Gateway noch in Reichweite ist. Dies wird mittels eines periodischen "beacon" durchgeführt. 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. 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 selbstständig feststellen wenn es einen Beacon mit einer unbekannten ID bekommt und durch einen Uplink dem Server mitteilen dass es 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 "beacons" nicht schlafen kann, ist die Klasse B für den Batteriebetrieb gedacht.

[SOR17b] (wird verwendet wenn mehr Bedarf für Empfangsfenster ist. Hierzu ist ein Synchronsignal nötig => zu bestimmten Zeiten kann damit empfangen werden Gateway sendet Beacon für Synchronisation. Um Daten empfangen zu werden werden Empfangsslots => Pingslots verwendet, werden periodisch geöffnet und mittels Beacon synchronisiert. Normalerweise werden diese schnell geschlossen außer es wird etwas empfangen. Gateway dessen Beacon benutzt wird, wird nach Empfangsqualität ausgewählt. Wenn neuer/unbekannter Bea-

chon von einem anderen GAteway empfangen wird, wird der netzwerkserver benachrichted und dieser entscheidet welcher verwendet wird().passt rozte an).

Das Netzwerk muss die standart ping-slot periode Datenrate und kanal kennen.

Um ein gerät auf klasse B zu kommen muss erst von Klasse A gewächstelt werden.

Entgeräte müssen Netzwerkserver über position nformieren. Dies kann über eine leere nachricht passieren oder eine normale(uplink).

Das beacon und die enthaltenen daten werden an die applikation geschiht. Der server kann den beacon auswerten. ziwischen beacon und uplink wird ran-dom time verwendet um kolisionen zu verhindern . änderungen an pingslot- periode .. muss mitgeteilt werden. Hierzu ist klasse A nötig => wechel zu A, wechel zu B.

Nachschuen
wie ge-
nau

Beacon wird genutzt um clockdrivt auszugleichen. Wenn kein beacon emp-fanen wird => Bacenless mode. Dieser wird bis zu 2 stunden beibehalten. Reines verlassen auf interne Uhr. Wenn beacon empfagne wird, wird zeit zu-rückgesetzt.)

das
funk-
tio-
niert

4.2.1 Klassenwechsel A nach B

Um einen Wechsel überhaupt zu ermöglichen muss der Netzwerkserber die de-vault ping-slot periodem die pingslot datenrate und den Pingslot channel ken-nen.

Ale endgeräte treten in Klasse A dem Netzwerk bei. Das wecheln in die klasse B wird durch folgenden Prozess realisiert.

Als erstes muss das Programm des ENdgerätes beim LoRaWAN layer an-frogen ob es möglich ist in klasse B zu wecheln. Der LAyer sucht nun nach einem baecon. Wird ein backen entdeckt, wird die BEACON_LOCKED Ser-visprimitive zurückgeliefert. Wenn kein Backoun empfangen wurde wir die BEACON_NOT_FOUND primitive zurückgegeben. Um diesen proess zu be-schleinigen kann das DeviceTimeReq MAC kommando verwendet werden. Da-mit wird das GAteway aufgefordert eien bAcon zu senden. Nun kann das end-

erklären

gerät in den modus B wächseln.

Als Zeites setzt der MAC Layer des engerätes das Class B Bit im FCtrl field Des Upliks auf 1. Dadurch ist er auch verandwortlich die Ping slots und für die Beacons zu öffnen. Dabei muss mit der größt möglichen abweiching der Internen Uhr gerechnet werden und demensprechend die Epfangsfenster angepasst werden. Diese darf pro Beacon nicht mehr als +/- 1.3msec liegen. Der Inhalt der Empfangenenen Beacons wird mit der Signalstärke and das Programm des Endgerätes zur weiteren Verarbeitung gesendet. Dmait kann z.B. dem LoRaWan layer angewiesen werden die Uhr nachzustellen.

[SOR17b](Endgerät fart LoRaWAN layer an. Layer sucht beacon. Mac command DeviceTimeReq um schneller bacon zu bekommen nutzen. Danach wird das ClassB feld auf 1 gesetzt. Bei den geöffneten fenstern wrd der maximal mögliche clockdrift berücksichtigt. Downlink läuft wie bei A ab.

)

4.2.2 Betrieb

Damit der NETzwerkserver dem Endgerät mittielen kann dass die pingslots frequen und/oder die Datnerate geändert werden soll gibt es den PingSlot-ChannelReq Mac kommando. Die werden sind in den argumenten enthalten.

Das Endgerät kann die Periode der Pingslots zu einer beliebigen Ziet ändern. Ist dies der Fall, so muss das ENdgerät in Köasse A wechseln mit mittels dem MAc kommando PingSlotChannelReq die geämderte periode Mitteilen . wird
Danach kann zuück in Kalsse B gewechselt werden. andere

Falls einige länger als 2 Stunden kein Beacon empfangen wird, kann die gespei-
synchronisation mit de Netzwerk verloren gehen. Dadruch funktoiniert die Kom- cher
munikation in Klasse B noicht merh und es wird in Klasse A gewechselt. Da 1/s
sich nun die Kommunikationsstrategie verändert muss mit einem Uplink in dem => hz
das CClassB Fled 0 ist, der Netzwerkserver informiert werden. Nun kann ver-
sucht werden eine verbindung mit der Klasse A aufzubauen. Das Programm
des Endgeätes kann versuchen wieder in Klasse B zu wecheln. Dieser prozess
kann sich immer weider wiederholen.

Um auch innerhalb der maximal 2 Stunden in denen 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 den Drift 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.

drift?

4.2.3 Singel / Multicast

Die Downlink der Klasse B unterscheidet sich nicht von denen der Klasse B. allerdings kann sich der Frequenzplan unterscheiden.

In Klasse B können die Nachrichten als Singelcast oder als Multicast Nachrichten verwendet werden. Eine Singelcast Nachricht wird an das Gerät das im DevAddr der Nachricht codiert ist gesendet. Im Multicastmodus wird das Paket an alle Endgeräte gesendet. Damit die möglich ist müssen sich die Geräte die selbe Multicast Adresse und die dazugehörigen Schlüssel teilen. Durch verschiedene Multicastadressen 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 also in der Applikationsebene sprich im Programm der Endgeräte oder Direkt bei der Personalisierung (Programmierung) erledigt werden.

In Multicastadressen sind keine MAC kommandos erlaubt. Nur Daten dürfen als Multicastnachricht übertragen werden. Dies wurde eingeführt da Multicastnachrichten nicht die selbe Robustheit wie Singelcastnachrichten haben. Die Nachrichten dürfen nicht acknologed werden. Das Fpending zeigt an das mehr Multicastnachrichten zu senden sind. [SOR17b] (separate Adresse für Multicast festgelegt durch layer oder manuell für gruppenmulticast nicht für MAC geeignet,)

unconfirmed

/ con-

firmed

4.2.4 Beacon

Wie schon erwähnt wird der Beacon verwendet um das Endgerät mit dem Netzwerk zu synchronisieren. Deswegen wird dieser Periodisch gesendet. Die Zeit zwischen zwei Beacons wird `BEACON_Period` genannt. Die Endgeräte öffnen Empfangsfenster um diese Beacons zu empfangen. Ein Beacon zu übertragen dauert `BEACON_RESERVED` lange. Das Beacon wird `Beacon_GUARD` früher geöffnet um sicher zu stellen das Beacon auch wirklich zu empfangen. Während versucht wird ein Beacon zu empfangen kann kein pingslot geöffnet werden. Ausserdem wird die `Beacon_GUARD` benutzt um sicherzustellen das kein Ping slot mehr geöffnet ist. Deswegen muss diese `Beacon_GUARD` mindestens so lang sein wie ein maximaler pingslot. Ein weiterer vorlesungsbezug? vorteil ist, dass nicht darauf geachtet werden muss wann ein pingslot geöffnet wird, da er sowieso im zweifelsfall fertig ist bevor ein beacon empfangen wird.

Um Synchronisierungen durch die Beacons zu vermeiden, wie alle endgeräte wollen sofort nach den beacon senden wollen, wird mittels zufälliger warte, pingslot zeiten und zufälliger pingslotanzahlen verhindert.

Beacons haben ihr eigenes Paketformat. Diese Pakete sind immer gleich lang. Dadurch kann auf header verzichtet werden was auch der Geschwindigkeit der verarbeitung zu gute kommt. Wie auch ein normales LoRaPaket, so besteht auch das erste feld des Beaconpaketes aus der Preamble nur das die des Beaconpaketes länger dauert was ein bemerkenswerter übertragungswahrscheinlicher macht. Danach folgt nur noch der BCNPayload. Der BCNPayload 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 Timefeld ist die zeit seit 00:00:00, Sunday 6th of January 1980 (start of the GPS epoch) modulo 2^{32} enthalten. das CRC feld wird verwendet um die korrektheit des Zeit und des RFU feldes zu versichern. Die andere hälfte ist Gatewayspezifisch. Sie enthält das GWSpecific feld und ein RFU feld das auch 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 be-

zieht.0 GPS coordinate of the gateway first antenna 1 GPS coordinate of the gateway second antenna 2 GPS coordinate of the gateway third antenna 3:127 RFU 128:255 Reserved for custom network specific broadcasts. Sonstige sich im infolled koordinaten enthalten kann dieses unterteilt werden in Längen und Breitengrad.

Auch Klasse A kann den beacon somit nutzen um herauszufinden von welchem gateway es gerade Daten empfängt und um somit eventuelle Standortwechsel festzustellen.

In Europa werden die Beacons auf einer festen Frequenz übertragen die sich nicht ändert außer über das MAC Kommando PingSlotChannelReq. Auf anderen Kontinenten kann es sein dass Frequenzhopping angewendet wird.

regionale
parameter
erwähnt?

4.3 Klasse C

C steht für CONTINUOUSLY Listening. Wie der Name schon sagt wird hier unaufhörlich ein empfangssender geöffnet. Dadurch wird es ermöglicht fast Latenzfrei zu übertragen. Dies bedeutet aber auch dass der Stromverbrauch am höchsten ist und somit nicht für den Batteriebetrieb geeignet. Das Gateway kann immer Daten senden außer wenn das Endgerät gerade Daten sendet. Hier sind Geschwindigkeit von bis zu 50mb möglich.

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 den gleichen Funktionen 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. Also auch während. Außerdem stehen die gleichen MAC Kommandos und zwei zusätzliche zur Verfügung.

Suche
normal

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

net?

[SOR17b] (öffnet RX1 und RX2 Fenster wie in Klasse A. Immer wenn nicht gesendet wird oder RX1 offen ist, ist RX2 offen. Multicast ist auch möglich.)

4.3.1 Wechsel von A nach C

Da es kein ClassC Fled in einem LoRapaket gibt, wurde für das umschalten in ClassC mode MAc kommandos eingeführt. Das endgerät sendet das Device-ModeInd commando. ALs parameter kann es 0 für Klasse A und 2 Für klasse C angeben. Der Netzkerkserver kann mit DeviceModeConf welches den wert der klasse enthät indas gewächst wurde.

5 Sicherheit

Weche

Sicherheit in netzwerkfähigen Systemen ist ein sher wichtiges und heiß diki- felder
tiertes thema. Da LoRa daten Über die Lpft überträgt, ist es extrem wichtig was
sich und die Dtaen zu schützen. Da Luft als Medium benutzt wird könnten ist wie
alle in der Nähe befindlichen geräte die gesendeten Daten mithören. Aber ge- ver-
nauso kann ein Endgerät sich als ein andres ausgeben und in seinem Namen schlüs-
Daten an ein Fremden Server send. Um zu verhindern das Gesnedene Daten selt?
mitgelsen werden müssen diese verschlüsselt werde. Um zu verhindern das je-
mand anderst so tut als wäre er das engerät müssen die Dtane Autentifiziert
werden. Slest jetzt könnten z.B. Join-request mitgeschnitten werden und von
dem (böken) endgerät wiederholt werden um das (gut) endgerät daran zu hin-
dern aktiv dem Netzwerk beizutreten. Deswegen wurden zähler eingebaut. Im
folgenden wird sich nächer damit beschäftigt welche mechanismen es gibt die
gennten probleme zu umgehen.

Oberflächlich gesehen bietet Lora eine end-to-end Sicherheit an, indem es
die Signale zweimal verschlüsselt. Die erste Verschlüsselung dient dazu die ge-
sendeten Daten vor eventuellen Mithörern zu verschlüsseln, also pm Endgerät
bis zu Gateway zu verschlüsslen. Die Verschlüsselung geschieht mit einem 128-
bit Network-Session-Key. Die zweite Verschlüsselung wird bis zur endgültigen
Weiterverarbeitung der Daten auf z.B. einen Server verwendet und ist ein 128
bit Application-Session-Key.

Nächer betrachtd benutzt Lore eine ganze Reihe an Schlüssel und Zähler
verwendet um die Kommunikation abzusichern. Da die verwendeten Schlüssel

bei OTAA-Aktivierung variiert wird hier eine viel höhere Sicherheit geboten als bei ABP-Aktivierung wo alle Schlüssel von Anfang an vorgegeben werden. Infolgedessen wird im folgenden Text auf die Sicherheit unter Verwendung von OTAA bezogen.

Jedes Endgerät hat seine eigenen NwkKey (Netzwerkschlüssel) und AppKey (Applikationsschlüssel). Sobald einem Netzwerk beigetreten wurde wird aus dem NwkKey der FNwkSIntKey, SNwkSIntKey und NwkSEncKey abgeleitet. Aus dem AppKey wird zusätzlich der AppSKey abgeleitet. Die Schlüssel müssen so gespeichert werden, dass es nicht möglich ist diese auf irgendeiner Weise aus dem Speicher zu holen außer für das Endgerät selber. Zusätzlich werden Join-Keys abgeleitet. JSInitKey und JSEncKey.

Der FNwkSIntKey ist einzigartig für ein Endgerät und heißt Forwarding Network session integrity key. Der Schlüssel wird verwendet um ganze oder Teile der MIC felder in den LoRaPaketen zu berechnen. mic

Serving Network session integrity key heißt abgekürzt SNwkSIntKey. Dieser Schlüssel wird verwendet um die Integrität des MIC codes zu überprüfen. Zusätzlich wird er auch verwendet um Teile des MIC codes zu berechnen. Dieser Schlüssel ist spezifisch für ein Endgerät.

NwkSEncKey oder lang Network session encryption key, ist für jede Netzwerksitzung einzigartig und wird verwendet um empfangene oder gesendete MAC kommandos zu ent- oder verschlüsseln.

Der AppSKey wird auch Application session key und wird einem Endgerät zugeordnet. Er wird vom Gateway und vom Endgerät verwendet um Daten die zum Applikationsserver geschickt werden sollen zu verschlüsseln.

pad fügt so viele 0en das die Länge an Vielfaches von 16 ist

$$\text{AppSKey} = \text{aes128_encrypt}(\text{NwkKey}, 0x02 \mid \text{JoinNonce} \mid \text{NetID} \mid \text{DevNonce} \mid \text{pad16})$$

$$\text{FNwkSIntKey} = \text{aes128_encrypt}(\text{NwkKey}, 0x01 \mid \text{JoinNonce} \mid \text{NetID} \mid \text{DevNonce} \mid \text{pad16})$$

$$\text{SNwkSIntKey} = \text{NwkSEncKey} = \text{FNwkSIntKey}.$$

Jedes Gerät hat 3 verschiedene Frame counter um die Anzahl der gesendeten und empfangenen Frames mitzuzählen der FCntUP counter zählt die Uplinkframes, der FCntDown zählt die MAC-downlinkframes und der AF-

CntDown welcher alle downlinkframes zählt die Nutzdaten enthalten.

Wenn ein Gerät dem Netzwerk beitrifft, werden zuerst die Counter auf 0 gesetzt. Beide Seiten einer Kommunikation halten die Zähler gleich. Beim Senden wird der aktuelle Counterwert in das FCnt-Feld eingetragen. Werden Übertragungen wiederholt, so wird der Counter nicht erhöht weiderholung

Durch das Verwerfen von Nachrichten mit zu kleinem Counterwert, wird schon verhindert, dass Pakete von einem Angreifer aufgenommen und zu einem späteren Zeitpunkt wiederabgespielt werden. drin?

Sogar bei den Join oder Accept-Nachrichten besteht die Gefahr einer Replay-Attacke. Da hier dem Netzwerk noch nicht beigetreten wurde, können die Zähler nicht verwendet werden. Hier wird eine Nonce in die Join-Pakete codiert. Diese Nonce zählt auf die gleiche Weise hoch wie die Counter. Die gegnerische Seite der Kommunikation muss die Nonce tracken und darf nur Pakete mit einer Nonce akzeptieren, die höher ist als die letzte Nonce. wie sieht

[GAS17] (Netzwerkserver hat AppKey daraus werden AppSKey und NwkS-key erzeugt) [Tec15] (Applikationsverschlüsselung (Schutz der Daten für Mitleser)) Netzwerk (Authentifizierung der Knoten) AFS, Key Exchange IEEE 802.11 [SOR17b] (symmetrischer Schlüssel => nur einer benötigt, Sessionkey ist abgeleitet von Knoten-rootkey. JoinServer stellt Verbindung der Keys her.) aus. frequencyhopping

6 Live-Beispiel

wenn vorhanden.

7 Ausblick

Die verwendete Frequenz entspricht der RX1 bzw. RX2 aus dem Kapitel Klasse ich zu A. wo kommt das her weit?

Sobald dem Netzwerk erfolgreich beigetreten wurde, werden die benötigten Schlüssel aus den vorher gesetzten Werten abgeleitet. Genaueres dazu in Kapitel Sicherheit.

Literatur

- [AVTP⁺17] ADELANTADO, FERRAN, XAVIER VILAJOSANA, PERE TUSET-PEIRO, BORJA MARTINEZ, JOAN MELIÀ-SEGÚI und THOMAS WATTEYNE: *Understanding the Limits of LoRaWAN*. <https://ieeexplore.ieee.org/abstract/document/8030482>, September 2017. Eingesehen am 09.04.2019.
- [CB⁺17] CHEONG, PHUI SAN, JOHAN BERGS, , CHRIS HAWINKEL und JEROEN FAMAHEY: *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): *LoRaWANTM 1.1 Backend(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 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.