TECHNISCHE UNIVERSITÄT
BERGAKADEMIE FREIBERG
Die Ressourcenuniversität. Seit 1765.

Fakultät für Mathematik und Informatik
Institut für Informatik
Lehrstuhl für Betriebssysteme und Kommunikationstechnologien

Bakkalaureatsarbeit

Entwicklung einer Sicherheitsinfrastruktur zur Bluetooth-Kommunikation zwischen Smartphone und Mikrocontroller

Marian Käsemodel

Angewandte Informatik
Vertiefung: Technik

Matrikel: 62 412

28. Juni 2021

Betreuer/1. Korrektor:
Prof. Dr. Konrad Froitzheim

2. Korrektor:
M.Sc. Jonas Treumer

Eidesstattliche Erklärung

Ich versichere, dass ich diese Arbeit selbstständig verfasst und keine anderen Hilfsmittel als die angegebenen benutzt habe. Die Stellen der Arbeit, die anderen Werken dem Wortlaut oder dem Sinn nach entnommen sind, habe ich in jedem einzelnen Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht. Diese Versicherung bezieht sich auch auf die bildlichen Darstellungen.

28. Juni 2021 Marian Käsemodel

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Themenstellung

1.2 Problemstellung

2 Grundlagen zu Bluetooth Low Energy

2.1 Überblick

Bluetooth ist ein Industriestandard für die Übertragung von Daten per Funk, dessen Intention die Reduzierung von Kabelverbindungen an mobilen sowie stationären Geräten war. Wichtige Eigenschaften der Technologie sind vor allem ein niedriger Energieverbrauch und niedrige Herstellungskosten der Hardware. Seit 1998 wird Bluetooth von der Bluetooth Special Interest Group (SIG) entwickelt und ist seit 2002 von der Organisation Institute of Electrical and Electronics Engineers (IEEE) standarisiert [1].

Es agiert im lizenzfreien ISM-Band (Industrial, Scientific and Medical Band) von 2,4 GHz. Zur Reichweite kann keine genaue Aussage getroffen werden, da diese von vielen Parametern wie beispielsweise der Sendeleistung und Einflüssen aus der Umwelt abhängt. Um trotzdem einen Eindruck zu gewinnen, kann für bestimmte Bedingungen und Konfigurationen die maximale Reichweite mithilfe eines Tools [2] der SIG ermittelt werden. Dabei variieren die Ergebnisse von ca. einem Meter bis hin zu mehr als 1000 Metern.

Grundlegend unterscheidet sich Bluetooth seit der Version 4.0 von 2010 in die zwei Systeme Basic Rate (BR) und Low Energy (LE), wobei LE darauf ausgelegt ist weniger Energie als BR zu benötigen. Die neuste Bluetooth-Version ist die Version 5.2, die wie jede Version abwärtskompatibel ist. Jedoch sind beide Systeme (BR und LE) bezüglich der Kommunikation miteinander inkompatibel. D.h. implementiert ein Gerät nur das BR-System, kann es keine Daten mit einem Gerät austauschen, das nur das LE-System unterstützt. Demnach ist für LE die Abwärtskompatibilität nur bis zur Version 4.0 gegeben. Desweiteren ist es möglich, dass ein Gerät über beide Systeme verfügt und so die meisten Nutzungsfälle abdeckt. Das BR-System kann mit den Erweiterungen Enhanced Data Rate (EDR) und Alternate Media Access Control and Physical Layer (AMP) genutzt werden, um eine höhere Datenrate zu erreichen. Die einzelnen Systeme und Erweiterungen können entsprechend ihrer Bluetooth-Version die in Tablle 1 dargestellten Datenraten erreichen.

System/Erweiterung	**max. Bitrate (Version 4.0)**	**max. Bitrate (Version 5.2)**
BR	1 Mb/s [3]	1 Mb/s [4]
BR/EDR	2 Mb/s bis 3 Mb/s [3]	2 Mb/s bis 3Mb/s [4]
802.11 AMP	24 Mb/s [5]	52 Mb/s [6]
LE	1 Mb/s [7]	2 Mb/s [8]

Tab. 1: Maximale Bitraten der Bluetooth-Systeme

Da Bluetooth Low Energy (BLE) ein zentraler Bestandteil dieser Arbeit ist, bezieht sich der Autor von nun an nur auf dieses und nicht mehr auf Bluetooth im Allgemeinen. D.h., dass Bluetooth Classic, welches BR/EDR und AMP beschreibt, nur noch behandelt wird, wenn das BR-System oder eine seiner Erweiterungen explizit erwähnt werden.

Die Architektur eines Bluetooth-Systems unterteilt sich in einen Host und in einen oder mehrere Controller. Ein Host ist eine logische Entität, definiert als alle Schichten unterhalb der nicht zu Bluetooth gehörigen Profile (Protokolle) und oberhalb des Host-Controller-Interface (HCI). Ein Controller ist eine logische Entität, definiert als alle Schichten bzw. Funktionsblöcke unterhalb des HCI. Der Aufbau setzt sich immer aus genau einem primären Controller und optional aus sekundären Controllern zusammen. Dabei kann die Rolle des primären

Controllers entweder durch einen BR/EDR-Controller, einen LE-Controller oder durch eine Kombination aus BR/EDR- und LE-Controller eingenommen werden, während die Rolle eines sekundären Controllers nur durch einen AMP-Controller besetzt werden kann. In Abb. 1 sind einige Varianten skizziert.

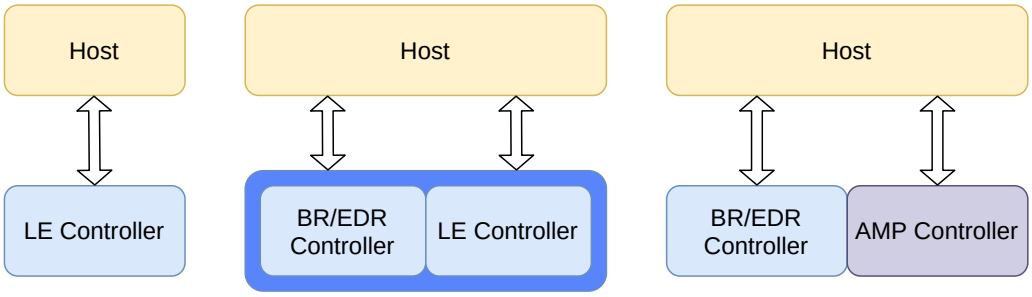

Abb. 1: Kombinationen aus Host und Controller [9]

Zur Veranschaulichung der Architektur bezüglich eines LE-Systems ist in der Abb. 2 die Zusammensetzung aus Host und Controller mit deren Schichten bzw. Protokollen festgehalten, die in den folgenden Sektionen 2.4 und 2.5 thematisiert werden. Über dem Host befindet sich die Anwendungsschicht.

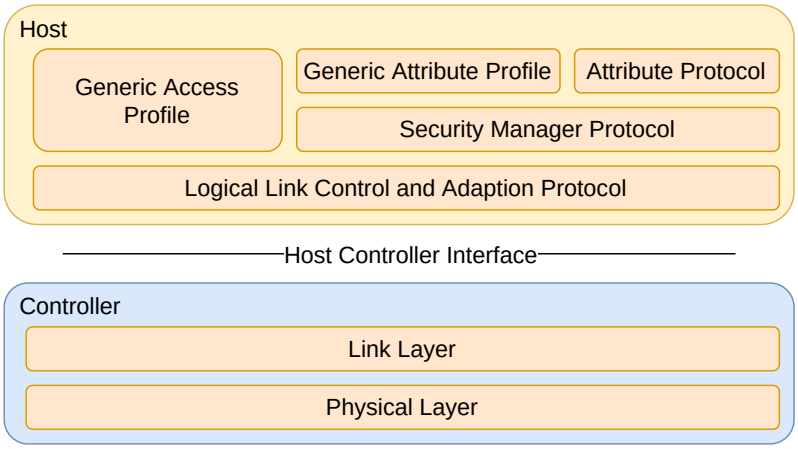

Abb. 2: Bluetooth Low Energy Architektur von Host und Controller [10]

2.2 Topologie

Ein Ad-Hoc-Netzwerk bestehend aus Bluetooth-Geräten wird Piconet genannt. Dabei existiert in einem Piconet immer ein Master-Gerät, das sich mit einem oder mehreren Slave-Geräten verbinden kann. Ein Gerät kann zur selben Zeit innerhalb mehrerer Piconets agieren, wobei die Rollen unabhängig sind. Z.B. könnte ein Gerät der Master eines Piconet A sein, während es ein Slave in einem Piconet B ist.

2.3 Verbindungsaufbau

Um mittels BLE ein Piconet zu bilden, benötigt es einen Advertiser. Auf drei vorgegebenen Frequenzen, den Advertising Channels (siehe X) , sendet dieser Daten, mit denen er

sich für andere Geräte bemerkbar macht (Advertisements). Dabei können Advertisements auch genutzt werden, um Nutzdaten zu senden. Jedes Advertisement-Paket beinhaltet eine Bluetooth-Adresse des Senders, welche 48 Bit lang ist.

Geräte, die Daten auf den Advertising Channels empfangen, werden Scanner bzw. Initiator genannt. Auf diesem Weg finden sich die Geräte (Discovering). Der Initiator unterscheidet sich vom Scanner, da er in der Lage ist, sich zu einem Advertiser zu verbinden, von dem er ein Advertisement erhielt, dass das Verbinden zu diesem ermöglicht. Sind zwei Geräte verbunden senden und empfangen sie ihre Pakete auf den Data Channels (siehe X). Verbinden sich zwei Geräte wird der Initiator zum Master und der Advertiser zum Slave.

Durch die Anwendung von Zeitmultiplexing senden die Geräte ihre Pakete immer zu festgelegten Zeitpunkten. Dabei ist ein Event ein zeitlicher Abschnitt, in dem zusammenhängende Daten in Form von Paketen gesendet bzw. empfangen werden. In Abbildung X ist ein Advertising Event dargestellt bei dem ein Advertiser auf allen drei Advertising Channels nacheinander Advertisement-Pakete sendet. Auf dem zweiten Kanal empfängt der Advertiser direkt gefolgt auf sein erstes Advertisement-Paket in diesem Kanal ein Paket eines Scanners, auf welches er mit einem weiteren Advertisement antwortet. In Abbildung X ist ein Advertising Event dargestellt, bei dem ein Initiator auf das Advertisement-Paket eines Advertiser antwortet, um eine Verbindung aufzubauen. Darauf folgt ein Connection Event bei dem Master (ursprünglich Initiator) und Slave (ursprünglich Advertiser) auf einem Data Channel sich gegenseitig Pakete senden. Danach folgt ein weiteres Connection Event auf einem anderen Data Channel.

2.4 Controller

Wie in Abbildung X zu sehen ist, umfasst der Controller die Schichten Physical Layer (PHY) und Link Layer (LL, auch Logical Layer genannt). Der Phyical Layer unterteilt sich weiter in die Physical Channels und die Phyical Links, während der Link Layer sich in Logical Transports und Logical Links aufteilt. In den Schichten bzw. ihren Untergliederungen wird definiert, wie Anwenderdaten, Advertisements und Kontrollsignale in Form von Unicasts bzw. Broadcasts übertragen werden.

2.4.1 Physical Layer

2.4.1.1 Physical Channel

Um miteinander zu kommunizieren, müssen zwei Bluetooth-Geräte (ein Sender und ein Empfänger) zur selben Zeit den selben Kanal nutzen, wobei sich der Empfänger in der Reichweite des Senders befinden muss. Da mehrere Piconets zu selben Zeit im selben lokalen Bereich agieren können, besteht die Wahrscheinlichkeit, dass zwei Sender zweier verschiedener Gerätepaare in Reichweite den selben Kanal zur selben Zeit nutzen, wodurch eine Kollision resultiert.

Mittles des Frequenzmultiplexverfahrens ist das ISM-Band eines LE-Systems von 2400 MHz bis 2483,5 MHz in 40 Funkkanäle aufgeteilt. Beginnend bei 2402 MHz nutzt jeder Kanal eine Frequenz, die 2 MHz über der Frequenz des Vorgängers liegt. Das Trägersignal wird mithilfe des Gaussian Frequency Shift Keying moduliert. [11]

Somit bildet der Physical Channel die niedrigste Ebene der Architektur. 37 der 40 Kanäle werden als LE Piconet Channel bezeichnet, die mit einem Piconet assoziiert werden und zur Kommunikation zwischen zwei bereits verbundenen Geräten dienen. Die verbleibenden drei Kanäle werden Advertisement Broadcast Channel genannt und befinden sich auf den Frequenzen 2402 MHz, 2426 MHz sowie 2480 MHz. [12]

Mittels Advertisements können Geräte in diesen drei Kanälen auf sich aufmerksam machen, damit andere Geräte diese entdecken können. Zudem werden diese genutzt, um Geräte miteinander zu verbinden oder Anwenderdaten an Scanner bzw. Initiatoren zu senden. Ein Gerät kann nur einen Kanal zur selben Zeit nutzen, weswegen das Zeitmultiplexverfahren verwendet wird, welches bereits verbundenen Geräten ermöglicht, zusätzlich das Advertisement zu betreiben.

Um Interferenzen innerhalb des genutzten Frequenzbands z.B. mit Wi-Fi zu vermeiden, wird das Adaptive Frequency Hopping [13] genutzt (eine Form des Frequency Hopping Spread Spectrum). Dabei wechseln Sender und Empfänger in kurzen Zeitabständen den Kanal und passen die Menge der zu nutzenden Kanäle (Channel Map) an, indem sie dynmaisch verfolgen in welchen Kanälen häufiger Kollisionen auftreten.

2.4.1.2 Physical Link

Ein Physical Link wird immer mit genau einem Physical Channel assoziiert. Dagegen kann ein Physical Channel mehrere Physical Links unterstützen. Bezüglich Bluetooth wird der Physical Link nicht in der Struktur eines Paketes repräsentiert, kann aber innerhalb eines LE-Paketes anhand der Access Address bestimmt werden.

Active Physical Links sind die Punkt-zu-Punkt-Verbindungen zwischen Master und Slave über einen Piconet Physical Channel und gelten nur als aktiv, wenn ein Asynchronous Connection (ACL) Logical Transport zwischen den Geräten existiert.

Advertising Physical Links dienen dazu, um zwischen einem Advertiser und einem Initiator einen Active Physical Link aufzubauen und existieren nur für einen kurzen Zeitraum. Zwischen Advertiser und Scanner existieren diese für längere Zeit und dienen dem Broadcast von Anwenderdaten.

2.4.2 Link Layer

2.4.2.1 Paketstruktur Der Link Layer nutzt ein gemeinsames Paketformat [14] für das Übertragen von Advertising-Paketen und Anwenderdaten-Pakten, dass in der Abbildung X dargestellt ist.

Die Preamble hat eine Größe von acht Bit und wird genutzt, um auf Empfängerseite die Frequenz zu synchronisieren, die Zeiteinteilung der Symbole zu schätzen und um die Automatic Gain Control zu trainieren. Die Preamble beträgt immer 0b01010101, falls das Bit mit dem niedrigsten Stellenwert (LSB für Least Significant Bit) der Access Address 1 ist. Anderenfalls beträgt die Preamble 0b10101010.

Die Access Address hat eine Größe von 32 Bit und identifziert eine Verbindung über den Link Layer bzw. dient dazu Pakete mittels des festgelegten Wertes 0x8E89BED6 als Advertisement-Pakete zu identifzieren. Bevor ein Initiator eine Verbindung zu einem Advertiser aufbaut, erstellt er eine zufällige Access Address, die neben weiteren Bedingungen nicht der des Advertisement-Pakets gleicht oder sich von dieser um ein Bit unterscheidet. Diese Access Address sendet er dann innerhalb der Verbindungsanfrage an den Advertiser.

Das letzte Feld des Link-Layer-Pakets ist der 24 Bit lange Cyclic Redundancy Check (CRC), der über das PDU-Feld berechnet wird. Im Fall, dass auf Ebene des Link Layer die PDU

verschlüsselt wird, wird der CRC erst nach der Verschlüsselung generiert. Unabhängig davon, ob verschlüsselt wird oder nicht, wird anschließend ein Whitening [15] durchgeführt, um Sequenzen vieler gleichbleibender Bits (bspw. 0b00000000) zu verhindern.

Die Protocol Data Unit (PDU) unterscheidet sich in die Advertising Channel PDU und Data Channel PDU.

Advertising Channel PDU

Wie in Abbildung X gezeigt wird, besteht die Advertising Channel PDU [16] aus einem 16 Bit langen Header und einem Payload variabler Länge. RFU steht dabei für Reserved for Future Use und wird nicht weiter behandelt.

Dabei beinhaltet der Header unter anderem ein 4 Bit langes Feld für den PDU Type (bspw. Connectable Undirected Advertising Event oder Scan Request) und zwei Flags TxAdd und RxAdd für zusätzliche Informationen bezüglich des PDU Type. Die Bedeutungen von TxAdd und RxAdd hängen vom PDU Type ab. Die Menge aller PDU Types lässt sich untergliedern in Advertising PDUs, Scanning PDUs und Initiating PDUs. Bei allen bilden die ersten 6 Bytes des Payload die Adresse des Senders (Advertiser, Scanner oder Initiator). Hier sagt TxAdd bei jedem PDU Type aus, ob die angegebene Adresse des Senders öffentlich (TxAdd = 0) oder zufällig (TxAdd = 1) ist. Öffentlich heißt, dass es die unverfäschte Adresse des Geräts ist, und zufällig demnach, dass eine zufällig generierte Adresse angegeben wird (siehe Sektion X). RxAdd dagegen ist nur bei PDU Types von Bedeutung, die in ihrem Payload eine zweite Adresse enthalten, nämlich die des Empfängers. Analog zu TxAdd sagt RxAdd aus, ob die Adresse des Empfängers öffentlich (RxAdd = 0) oder zufällig (RxAdd = 1) ist.

Ein weiteres Feld im Header der Advertising Channel PDU ist das 6 Bit lange Feld für die Länge des Payloads in Bytes, dessen Wert eine Spanne von 6 bis 37 Byte deckt.

Data Channel PDU

Die Data Channel PDU [17] nutzt entsprechend der Abbildung X einen 16 Bit langen Header, einen Payload variabler Länge und optional einen 32 Bit langen Message Integry Check (MIC), der die Integrität des Payload sicherstellt. Das MIC-Feld entfällt bei einer unverschlüsselten Link-Layer-Verbindung und bei einer Data Channel PDU, deren Payload die Länge null Beträgt.

Das erste Feld des Header ist der 2 Bit lange Link Layer Identifier (LLID), der mit 0b01 sowie 0b10 aussagt, dass es sich um eine LL Data PDU handelt und mit 0b11, dass es sich um eine LL Control PDU handelt. Der Wert 0b00 ist reserviert. Die LL Control PDU dient dazu, um die LL-Verbindung zu steuern. Dazu gehören unter anderem Anfragen zum Ändern der Verbindungsparameter (z.B. Window Size oder Wert bis zur Zeitüberschreitung), zum Ändern der Channel Map oder zum Verschlüsseln.

Auf die LLID folgt das Feld der Next Expected Sequence Number (NESN) und das Feld der Sequence Number (SN) mit jeweils einem Bit Länge, die innerhalb Sektion X näher erläutert werden.

Unter anderem beinhaltet der Header ein 5 Bit langes Feld für die Länge des Payload in Byte und ggf. einschließlich der Länge des MIC. Der maximale Wert der Länge beträgt 31

Byte, wobei sich der Payload in jedem Fall auf eine maximale Länge von 27 Byte bemisst.

2.4.2.2 Logical Transport

Über dem Physical Layer baut sich der Link Layer auf, beginnend mit dem Logical Transport, der sich in die zwei Arten LE Asynchronous Connection (LE ACL) und LE Advertising Broadcast (ADVB) unterteilt.

Die LE ACL transportiert Kontrollsignale des über ihr befindlichen Logical Link und Logical Link Control and Adaption Protocol (L2CAP). Außerdem überträgt die LE ACL asynchrone Anwenderdaten nach dem Best-Effort-Prinzip.

Mithilfe der Next Expected Sequence Number bzw. Sequence Number (NESN/SN), die jeweils nur die Größe eines Bits besitzen, wird eine einfache Zuverlässigkeit gewährleistet. Empfängt ein Gerät ein Paket B, vergleicht es dessen NESN mit der SN, die es innerhalb des vorherigen Pakets A abgesendet hat. Wenn diese unterschiedlich sind, wurde das vorherige Paket A vom Gegenüber vollständig und korrekt empfangen (ACK für Acknowledgement). Anderenfalls sind die Nummern gleich, was bedeutet, dass das vorherige Paket A nicht vollständig bzw. korrekt empfangen wurde (NACK/NAK für Negative Acknowledgement) und nun erneut an den Gegenüber gesendet werden muss. Zudem prüft das Gerät die SN des empfangenen Pakets B mit der NESN seines vorher gesendeten Pakets A. Sind diese Nummern gleich, wurde das empfangene Paket B vom Gerät erwartet. Anderenfalls sind die Nummern verschieden und somit wurde das Paket B nicht erwartet, weswegen es ignoriert wird. Somit wird auch die Flusskontrolle ermöglicht, da ein Empfänger bei nicht ausreichend freiem Speicher im Buffer ein NACK mithilfe der Sequenznummern zurücksenden kann. [18]

Wenn ein Gerät einem Piconet beitritt, wird zwischem dem Master und dem Slave eine Default LE ACL über einen Active Physical Link gebildet. Die Default LE ACL ist einer Access Address zugeordnet. Wird die Default LE ACL getrennt, werden alle Logical Transports zwischen Master und Slave getrennt. Bei einem unerwarteten Synchronisationsverlust zum LE Piconet Physical Channel werden der LE Physical Link und alle LE Logical Transports und LE Logical Links entfernt.

Der ADVB transportiert ohne Verwendung von Acknowledgements Kontrollsignale und Anwenderdaten bezüglich des Broadcasts über den darunter gelegenen LE Advertising Broadcast Link. Der Datenverkehr ist überwiegend unidirektional, ausgehend vom Advertiser zu allen in Reichweite befindlichen Scannern. Scanner können eine Anfrage an den Advertiser senden, um weitere Anwenderdaten über den Broadcast zu empfangen oder um eine LE ACL zu bilden. Aufgrund des Verzichts auf Acknowledgements ist der ADVB unzuverlässig, weswegen Pakete redundant übertragen werden. Sobald ein Gerät mit dem Advertising beginnt, wird ein ADVB erzeugt, der anhand der Adresse des Gerätes identifiziert wird. [19]

2.4.2.3 Logical Link

Ein Logical Link unterscheidet sich je nachdem, ob er auf einem LE ACL Logical Transport oder einem ADVB Logical Transport aufbaut und ob er zur Übertragung von Kontrollsignalen oder Nutzdaten genutzt wird. Anhand des Logical Link Identifier (LLID) im Header des Basisbandpakets wird unterschieden, ob es sich bei der zu übertragenden PDU um Nutzdaten oder Kontrollsignale handelt.

Der Control Logical Link (LE-C) nutzt den darunter liegenden LE ACL Logical Transport, um Kontrollsignale zwischen Geräten im Piconet zu übertragen.

Der User Asynchronous Logical Link (LE-U) nutzt den darunter liegenden LE ACL Logical Transport, um alle asynchronen Anwenderdaten zu übertragen. Über dem Link Layer

agiert das Protokoll L2CAP, dessen Frame für den Link Layer fragmentiert werden müssen. Mithilfe des LLID-Wertes 0b10 wird der Beginn eines L2CAP-Frame (das erste Fragment eines L2CAP-Frame) und der Wert 0b01 die Fortsetzung eines L2CAP-Frame (die folgenden Fragmente des L2CAP-Frame) gekennzeichnet. Somit wird der Header des Protokolls L2CAP einfach gehalten und eine korrekte Synchronisation bei der Zusammensetzung der Fragmente zu einem L2CAP-Frame garantiert. Jedoch muss folglich ein L2CAP-Frame vollständig übertragen werden, bevor ein neues übertragen wird.

Der Advertising Broadcast Control Logical Link (ADVB-C) nutzt den darunter liegenden Default ADVB, um Kontrollsignale für Verbindungsanfragen oder Anfragen für weitere Broadcast-Anwenderdaten zu übertragen.

Der Advertising Broadcast User Data Logical Link (ADVB-U) nutzt den darunter liegenden Default ADVB, um verbindungslos und ohne den Gebrauch von LE-U Anwenderdaten als Broadcast zu senden. [20]

2.5 Host

Im Wesentlichen umfasst der Host das Logical Link Control and Adaption Protocol (L2CAP), das Generic Access Profile (GAP) und Generic Attribute Profile (GATT) sowie das Security Manager Protocol (SMP).

Damit Host und Controller Daten austauschen können, dient das Host Controller Interface (HCI) als Schnittstelle zwischen diesen. Jedoch soll auf dieses nicht weiter eingegangen werden, da es die Übertragung der Daten nicht nennenswert beeinflusst.

2.5.1 Logical Link Control and Adaption Protocol

-Grob "was ist l2cap"mit einordnung host -funktionen

- QoS mit Konfiguration möglich, mapped channel ACL-U logical Link oder LE-U - unterstützt verbindunsorientierte Channel - neben Aufbau, Konfiugration und Abbau von L2CAP-Channels, ist L2CAP für das Multiplexing der Service Data Units (SDU) auf ACL-U, LE-U zuständig - pro channel flusskontrolle (muss bei channel aufbau eingestellt werden) - enhanced error detection and retransmission - bei HCI muss L2CAP die SDUs in Fragmente segmentieren, die in Basebandbuffers passen, und eine Token-basierende Flusskontrolle über das HCI ausführen, um Fragmente nur an das Baseband zu übertragen, wenn es erlaubt ist

Das Logical Link Control and Adaption Protocol (L2CAP) bildet die unterste Schicht im Host (siehe Abb. X) und dient je nach Konfiguration dazu, um den Datenverkehr zu steuern und um zwischen höheren und niedrigeren Schichten zu vermitteln. Es verfügt über fünf Modi:

- Basic L2CAP Mode
- Flow Control Mode
- Retransmission Mode
- Enhanced Retransmission Mode
- Streaming Mode
- LE Credit Based Flow Control Mode (seit Bluetooth 4.2)

Für Bluetooth allgemein (BR/EDR und LE) wird immer der Basic L2CAP Mode genutzt, wenn kein anderer festgelegt wird. Der LE Credit Based Flow Control Mode soll [21] zufolge als einziger Modus für verbindungsorientierte LE-Kanäle genutzt werden ("This is the only

mode that shall be used for LE L2CAP connection oriented channels"[21]). Da diese Aussage nicht ausschließt, dass eine verbindungsorientierte LE-Verbindung über den standardmäßig festgelegten Basic L2CAP Mode erfolgen kann, und der LE Credit Based Flow Control Mode noch nicht in der Bluetooth-Version 4.0 vertreten war [22], ist anzunehmen, dass eine verbindungsorientierte LE-Verbindung auch mit dem Basic L2CAP Mode möglich ist.

Logische Kanäle, genannt L2CAP Channels, dienen innerhalb eines Geräts als Endpunkt für höher gelegene Protokolle oder direkt für die Anwendung und sind für jedes Gerät individuell an dem Channel Identifier (CID) unterscheidbar. D.h. der L2CAP Channel einer Verbindung zwischen zwei Geräten muss von diesen nicht zwingend mit der gleichen CID gekennzeichnet sein. Einige CIDs sind bestimmten Zwecken zugeteilt (siehe Anhang X).

Der L2CAP Layer wird von zwei Modulen gesteuert: dem Resource Manager und dem Channel Manager (siehe Abb. X).

Der Channel Manager ist in Bezug auf L2CAP zuständig für die Signalübertragung intern, Peer-to-Peer, und zu höheren und niedrigeren Schichten. Die Signale, die zwischen zwei L2CAP-Entitäten zweier verbundenen Geräte übertragen werden, sind Kommandos wie bspw. die LE Credit Based Connection Request und die entsprechende Response (siehe Anhang Tabelle X). Dafür wird ein separater L2CAP Channel mit der CID 0x0005 genutzt. Zudem betreibt der Channel Manager den L2CAP-Zustandsautomaten, auf den hier nicht näher eingegangen werden soll.

Da der L2CAP Layer nach unten durch das HCI mit dem Link Layer verknüpft ist, müssen die L2CAP PDUs dem Paketformat des HCI und dieses dem Paketformat des Link Layers gerecht werden. Dementsprechend werden die L2CAP PDUs fragmentiert bzw. wieder zusammengesetzt.

Der Resource Manager übernimmt mehrere Aufgaben. Dazu zählt die Kapselung der Service Data Units (SDU)

Der Resource Manager ist zuständig für den Frame Relay Service, die Segmentierung bzw. Zusammensetzung der Service Data Units (SDU), die erneute Übertragung und Flusskontrolle, sowie die Kapselung bzw. Entkapselung.

Eine Besonderheit des L2CAP ist das Protokoll- und das Kanalmultiplexverfahren. Bei Ersterem wird während des Erstellens eines L2CAP Channel die Verbindung zum zugehörigen höhergelegenen Protokoll geleitet. Zweiteres wird genutzt, um bei der Datenübertragung zwischen mehreren höhergelegenen Entitäten zu differenzieren, da mehrere dieser Entitäten das gleiche Protokoll nutzen könnten.

An diesen L2CAP Channels lässt sich das Multiplexverfahren über einen oder mehrere Logical Links anwenden. Dabei ist die Art des verwendeten Controllers nicht zwingend ein LE Controller, da L2CAP auch kompatibel mit dem BR/EDR Controller oder einer Kombination aus BR/EDR und LE ist.

2.5.2 Generic Attribute Profile

Das Generic Attribute Profile (GATT) dient zur Kommunikation zwischen Client und Server. Es basiert auf dem Attribute Protocol (ATT), welches ausgehend vom Server Attribute bereitstellt, die von einem oder mehreren Client entdeckt werden können. Ein Attribut be-

steht aus einem Typ, der anhand des Universal Unique Identifier (UUID) identifiziert wird, einem Attribute Handle, um auf das Attribut zuzugreifen, und einem Wert, der von Server und Client ausgelesen bzw. überschrieben werden kann. Zudem ist es möglich für Attribute Berechtigungen festzulegen (bspw. nur Lesen, nicht Schreiben). [23]

Entsprechend der Abbildung X ordnet sich ATT über L2CAP in den Protokollstapel ein.

Möchte ein Client einen Attributwert lesen bzw. schreiben, dann sendet er eine Read Request bzw. eine Write Request an den Server. Dieser reagiert, indem er bei einer Read Request das Attribut an den Client sendet oder bei einer Write Request den Attributwert entsprechend ändert und dem Client eine Bestätigung sendet. Im Fall, dass ausgehend vom Server ein Attribut geändert werden soll, sendet dieser eine Notification oder eine Indication an den Client. Im Gegensatz zur Notification wird eine Indication vom Client bestätigt, falls diese empfangen wurde. [24] [25]

Die Daten werden in Form von Attribute Protocol PDUs (siehe Abb. X) übertragen. Der ein Byte lange Opcode sagt aus, ob die PDU entweder eine Request, Response, Notification, Indication oder Bestätigung ist, und enthält eine Flag für die Authentifizierung. Die Attributparameter unterteilen sich in zwei Byte für den Attribute Handle, 2 oder 16 Byte für den Attributyp also die UUID, den Attributwert variabler Länge und die Attributberechtigungen, deren Länge von der Implementierung abhängig ist. Auf die Attributparameter folgt die 12 Byte lange Signatur für die Authentifizierung, falls diese gefordert wird. [26]

GATT bildet eine Hierachie (siehe Abb. X) bestehend aus den grundlegenden Elementen

hbt!

Abb. 3: [27]

Profile, Service und Characteristic, die alle als Attribute definiert werden. An oberster Stelle befindet sich das Profile. Dieses enthält einen oder mehrere Services. Ein Service kann wiederum eine oder mehrere Characteristics enthalten, die sich aus Properties, einem Wert und Descriptors zusammensetzen.

2.5.3 Generic Access Profile

Das Generic Access Profile (GAP) definiert verschiedene Rollen und Modi bzw. Prozeduren für Broadcasts und den Aufbau von Verbingungen.

Für LE existieren die vier Rollen: Broadcaster, Observer, Peripheral und Central. Ein Broadcaster ist aus Sicht des Link Layers (LL) ein Advertiser, da er verbindungslos Daten in Form von Advertising Events sendet. Der zugehörige Modus ist der Broadcast Mode. Diese Advertising Events können von Observern, die die Observation Procedure ausführen, empfangen werden, weswegen diese aus Sicht des LL als Scanner bezeichnet werden. Die Rolle Peripheral wird einem Gerät zugewiesen, wenn es den Aufbau eines LE Physical Link akzeptiert. Dabei nimmt es in Bezug auf den Link Layer die Rolle des Slave ein. Die Rolle Central wird einem Gerät zugewiesen, wenn dieses den Aufbau einer physischen Verbindung einleitet. Dabei nimmt es in Bezug auf den Link Layer die Rolle des Master ein. Ein Gerät kann mehrere Rollen zur selben Zeit einnehmen. [28] [29]

Jedes Gerät befindet sich entweder im Non-discoverable Mode, in dem es nicht von anderen Geräten entdeckt werden kann, oder im General Discoverable Mode bzw. im Limited Discoverable Mode, in denen es entdeckbar ist. Im Letzteren ist ein Gerät nur für eine bestimmte Dauer entdeckbar. Geräte, die andere Geräte entdecken sollen, müssen die General Discovery Procedure bzw. Limited Discovery Procedure ausführen. [30]

Um Verbindungen und deren Aufbau zu steuern, gibt es mehrere Modi und Prozeduren, von denen einige in der Tabelle X zusammengefasst werden.

Modus/Prozedur	**Beschreibung**
Non-connectable Mode	keine Verbindungen akzeptieren
Directed Connectable Mode	nur Verbindungen von bekannten Peer-Geräten (Bluetooth-Adresse bekannt) akzeptieren, die die Auto oder General Connection Establishment Procedure ausführen
Undirected Connectable Mode	nur Verbindungen von Geräten akzeptieren, die die Auto oder General Connection Establishment Procedure ausführen
Auto Connection Establishment Procedure	Aufbau von Verbindungen zu Geräten, die in einem Connectable Mode sind und deren Adresse auf der Whitelist eingetragen ist
General Connection Establishment Procedure	Aufbau von Verbindungen zu bekannten Peer-Geräten, die in einem Connectable Mode sind
Connection Parameter Update Procedure	Peripheral oder Central kann Link-Layer-Parameter einer Verbindung ändern

Tab. 2: Modi und Prozeduren für Verbindungen [31]

Bonding ist das Speichern von Sicherheits- und Identitätsinformationen wie dem Identity Resolving Key (IRK) oder dem Long Term Key (LTK). Der IRK generiert zufällige Adressen oder löst diese auf. Der LTK wird beim Pairing generiert und dient zur Verschlüsselung einer Verbindung (siehe Sektion X).

Ein Gerät im Non-Bondable Mode erlaubt es nicht einen sogenannten Bond mit einem Peer-Gerät zu erstellen. Dagegen erlaubt dies der Bondable Mode. Haben ein Central und ein Periphal zusammen einen Bond erstellt, können sie beim erneuten Verbinden mithilfe der gespeicherten Sicherheits- und Identitätsinformationen das Pairing überspringen. Broadcaster und Obersever sollten das Bonding nicht unterstützen. [32]

2.5.3.1 Sicherheitsaspekte

Zusätzlich verfügt GAP über Sicherheitsaspekte, die unteranderem das Privacy Feature und die Random Device Address enthalten. Das Privacy Feature erschwert das Verfolgen eines Gerätes durch das Ändern der Adresse des Gerätes und wird nur in den Modi und Prozeduren für Verbindungen genutzt, falls es angewandt werden soll.

Entsprechend Abb. X existieren verschiedene Arten von Bluetooth-Adressen.

Die öffentliche Bluetooth-Adresse ist die vom Hersteller bzw. IEEE festgelegte Adresse mit einer Länge von 48 Bit. Die 24 Bit mit dem höchsten Stellenwert bilden die von der IEEE vergebene Hersteller-ID und die 24 Bit mit dem niedrigsten Stellenwert bilden die vom Hersteller festgelegte Geräts-ID. [33]

Eine zufällig generierte Adresse kann entweder statisch oder privat sein. Eine zufällige statische Adresse kann bei jedem Neustart des Gerätes geändert werden und soll nicht während der Laufzeit geändert werden. Die zwei Bits mit dem höchsten Stellenwert sind jeweils auf 1 gesetzt, während die restlichen 46 Bit zufällig generiert werden, ohne dass alle nur auf 0 oder nur auf 1 gesetzt sind.

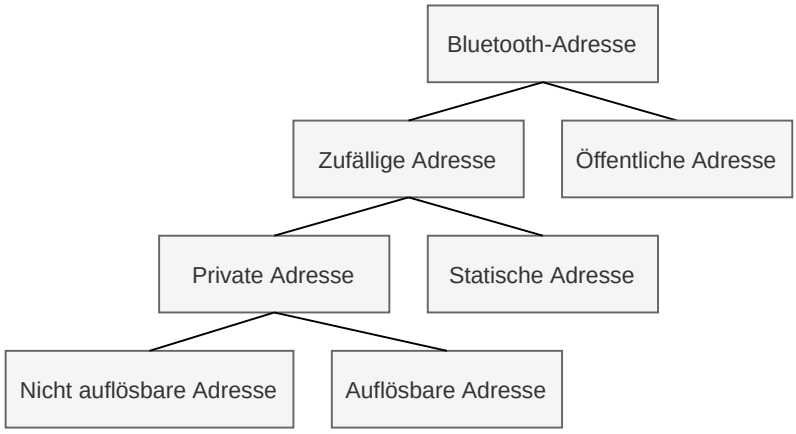

Abb. 4: Arten der Bluetooth-Adressen

Zufällig generierte private Adressen werden in auflösbare und nicht auflösbare Adressen unterschieden. Eine Nicht auflösebare Adresse ist an den zwei Bit mit höchstem Stellenwert jeweils auf 0 gesetzt und die restlichen 46 Bit werden zufällig generiert, ohne dass alle nur auf 0 oder nur auf 1 gesetzt sind. Zudem darf diese Adresse nicht der öffentlichen Adresse gleichen.

Mittels eines IRK (der lokale oder der eines Peer-Geräts) und einer zufällig generierten Zahl (24 Bit Länge) kann ein Gerät eine auflösbare Adresse generieren. Die zwei Bit der zufällig generierten Zahl mit dem höchsten Stellenwert sind auf 0 (höchster Stellenwert) und 1 (zweithöchster Stellenwert) gesetzt, während die restlichen 22 Bit zufällig generiert werden, ohne dass alle nur auf 0 oder nur auf 1 gesetzt sind. Diese zufällig generierte Zahl nimmt die 24 Bit mit den höchsten Stellenwerten ein. Die restlichen 24 Bit mit den niedrigsten Stellenwerten der Adresse repräsentieren einen Hashwert, der mit der Funktion ah [34] und den Eingabewerten IRK und der zufällig generierten Zahl ermittelt wird. [35]

Nutzt ein Broadcaster oder Observer das Privacy Feature, dann verwendet er eine auflösbare oder nicht auflösbare Adresse, die er zeitlich periodisch ändert, solang er das Advertising bzw. Scanning betreibt. Jedoch wird die Adresse nur dann geändert, wenn der Controller die Auflösung von Adressen nicht unterstützt. Beim Advertising sollte der Broadcaster (genauso Peripherals) nicht seinen Gerätenamen oder Daten preisgeben, die zu dessen Identifikation verhelfen. [36]

Ein Peripheral mit aktiviertem Privacy Feature nutzt im Connectable Mode eine auflösbare Adresse und im Non-Connectable Mode eine auflösbare oder nicht auflösbare Adresse. Empfängt das Peripheral eine auflösbare Adresse und verfügt über Bonding-Information, löst es mit diesen die empfangene Adresse auf. Konnte die empfangene Adresse erfolgreich aufgelöst werden, kann das Peripheral die Verbindung mit dieser akzeptieren. Anderenfalls wird die Verbindung abgelehnt oder das Pairing ausgeführt. Basiert das Privacy Feature des Peripheral auf dem Host, dann wird die auflösbare bzw. nicht auflösbare Adresse nur während des Advertising zeitlich periodisch geändert. [37]

Während des Scanning nutzt ein Central eine auflösbare oder nicht auflösebare Adresse, als Initiator dagegen nur eine auflösbare Adresse. Empfängt das Central eine auflösbare Adresse und verfügt über Bonding-Information, löst es mit diesen die empfangene Adresse auf. Konn-

te die empfangene Adresse erfolgreich aufgelöst werden, kann das Central die Verbindung mit dieser akzeptieren. Anderenfalls wird die enpfangene Adresse nicht beachtet. Basiert das Privacy Feature des Central auf dem Host, dann wird die auflösbare bzw. nicht auflösbare Adresse nur während des Scanning zeitlich periodisch geändert. [38]

2.5.4 Security Manager

Der Security Manager (SM) bzw. das Security Manager Protocol (SMP) ist dafür zuständig eine sichere Verbindung zwischen zwei Geräten (Master und Slave) aufzubauen. Dies umfasst im Wesentlichen das Pairing, welches zur Authentifizierung und Generierung eines Schlüssels dient, und die Verteilung von Schlüsseln. Entsprechend der Abb. 5 lässt sich der Security Manager bzw. das SMP in die BLE-Architektur einordnen.

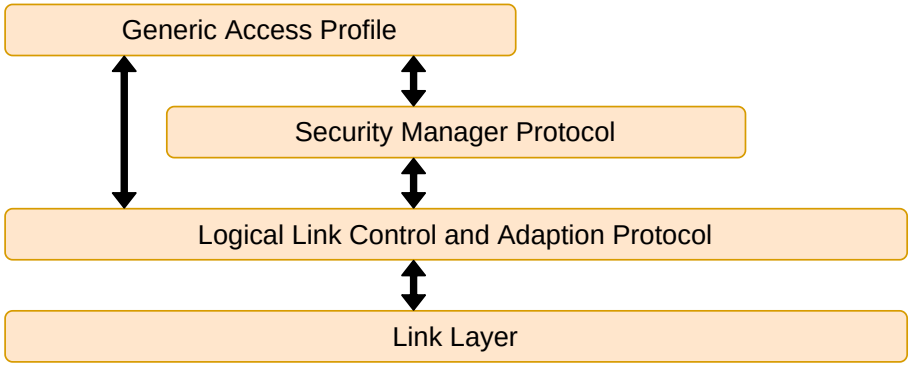

Abb. 5: Beziehungen des Security Managers zu anderen Komponenten [39]

Um mit dem Link Layer zu interagieren, nutzt der SM einen L2CAP Channel mit einer festgelegten CID. Zudem kann er mit GAP direkt kommunizieren.

Das Pairing lässt sich in drei Phasen aufteilen. Phase 1 und 2 dienen der Generierung eines Schlüssels, den beide Geräte zur Verschlüsselung und Authentifizierung im Link Layer nutzen. Dafür wird der Cipher Block Chaining - Message Authentication Code (CCM) Mode in Verbindung mit dem Advanced Encryption Standard (AES), genauer dem AES-128 Block Cipher, genutzt [40].

2.5.4.1 Pairing: Phase 1

Die erste Phase ist der Pairing Feature Exchange, bei dem die beiden Geräte ihre für den Nutzer zugänglichen Ein- und Ausgabemöglichkeiten (IO Capabilities) austauschen. Die Tabellen 3 und 4 listen die entsprechenden Bezeichnungen dieser Möglichkeiten auf.

Eingabemöglichkeit	Beschreibung
Keine Eingabe	Gerät kann keine Eingaben entgegennehmen
Ja / Nein	Gerät kann zwei verschiedene Eingaben verarbeiten, die der Bedeutung Ja bzw. Nein zugewiesen werden können (bspw. zwei Tasten)
Tastatur	Eingabe der Ziffern 0 bis 9 möglich und die Möglichkeit Ja und Nein einzugeben

Tab. 3: Eingabemöglichkeiten eines Gerätes [41]

Ausgabemöglichkeiten	Beschreibung
Keine Ausgabe	Gerät kann keine 6-stellige Dezimalzahl dem Nutzer anzeigen bzw. kommunizieren
Numerische Ausgabe	Gerät kann eine 6-stellige Dezimalzahl dem Nutzer anzeigen bzw. kommunizieren

Tab. 4: Ausgabemöglichkeiten eines Gerätes [42]

Desweiteren tauschen die Geräte in der ersten Phase Informationen darüber aus, ob eine Authentifizierung zum Schutz vor einem Man-In-The-Middle-Angriff nötig ist (MITM Flag), und ob Daten für die Authentifizierung über die Pairing-Methode Out Of Band (OOB), d.h. mittels einer anderen Technologie (z.B. Near Field Communication), übertragen werden können (OOB Flag). Außerdem werden die minimalen und maximalen Größen für die Schlüssel ausgetauscht, die sich in 8 Bit großen Schritten zwischen 56 Bit (7 Byte) und 128 Bit (16 Byte) befinden. Dabei wird der kleinere Wert beider maximaler Größen übernommen. Falls die beiden Spannen sich nicht schneiden, wird das Pairing abgebrochen.

2.5.4.2 Pairing: Phase 2

Anhand der ausgetauschten Informationen aus der ersten Phase wird in der zweiten Phase entschieden, welche der folgenden Methoden zur Generierung des Short Term Key (STK) bzw. Long Term Key (LTK) zu verwenden ist:

- Numeric Comparison (für LE erst ab Bluetooth-Version 4.2)
- Just Works
- Out of Band (OOB)
- Passkey Entry

Da das Pairing (speziell Phase 2) für LE in der Bluetooth-Version 4.0 funktionale Unterschiede zum Pairing für LE ab Version 4.2 aufweist, wird das Pairing für LE in Version 4.0 als **LE Legacy Pairing** und das Pairing für LE ab Version 4.2 als **LE Secure Connections Pairing** bezeichnet. Aus Sicht des Nutzers sind diese jedoch gleich und jede Bluetooth-Version, die LE unterstützt, unterstützt das LE Legacy Pairing. Im Gegensatz zum LE Secure Connections Pairing bietet LE Legacy Pairing für die Methoden Just Works und Passkey Entry keinen Schutz vor passivem Abhören während des Pairings, da LE Secure Connections Elliptic Curve Diffie-Hellman (ECDH) für den Schlüsselaustausch nutzt und LE Legacy Pairing nicht. [43]

Haben beide Geräte OOB-Authentifizierungsdaten für das LE Legacy Pairing, wird unabhängig von der jeweiligen MITM-Flag die Methode OOB gewählt. In LE Secure Connections Pairing wird ebenso verfahren, nur dass hier nicht die OOB-Flag beider Geräte gesetzt sein müssen (aber können), da eine gesetzte OOB-Flag genügt. Ist die MITM-Flag beider Geräte nicht gesetzt, wird die Methode Just Works ausgeführt. Anderenfalls werden die Ein- und Ausgabemöglichkeiten der Geräte für die Wahl der Methode einbezogen (siehe Anhang Tabelle X).

Pairing Methoden

Bei der **Numeric Comparison** wird dem Nutzer auf beiden Geräten jeweils eine zufällig generierte sechsstellige Dezimalzahl angezeigt. Diese muss der Nutzer vergleichen und im Falle der Übereinstimmung auf beiden Geräten bestätigen oder anderenfalls ablehnen. Somit kann der Nutzer unabhängig von der Namensgegebung der Geräte sicherstellen, die richtigen Geräte ausgewählt zu haben. Zudem bietet diese Methode Schutz vor MITM-Angriffen. Außenstehende, die Kenntnis über diese Zahl gewonnen haben, können laut [44] damit keinen Vorteil zur Entschlüsselung der zwischen den beiden Geräten ausgetauschten Daten erlangen, da die Zahl nicht als Eingabe zur Generierung eines Schlüssels verwendet wird.

Just Works basiert auf der Funktionsweise von Numeric Comparison mit dem Unterschied, dass hier dem Nutzer keine sechsstellige Dezimalzahl ausgegeben wird und er die Verbindung nur bestätigen muss. Dadurch bietet Just Works keinen Schutz vor MITM-Angriffen, aber einen Schutz gegen passives Abhören (außer für LE Legacy Pairing). [45]

Out of Band ist die Nutzung einer anderen Technologie (z.B. Near Field Communication), um Geräte zu entdecken oder um kryptographische Informationen für den Pairing-Prozess auszutauschen. Dabei sollte die Technologie Schutz vor MITM-Angriffen bieten. [46]

Die Methode **Passkey Entry** definiert, dass ein Gerät die zufällig generierte sechstellige Dezimalzahl ausgibt und der Nutzer diese auf dem anderen Gerät eingeben muss. Ein Schutz gegen MITM-Angriffe existiert, da diese nur mit einer Wahrscheinlichtkeit von 0,000001 für jede Durchführung der Methode möglich sind. Schutz gegen passives Abhören bietet Passkey Entry nur in LE Secure Connections Pairing und nicht in LE Legacy Pairing. [47] [48]

LE Legacy Pairing: Schlüssel und deren Generierung

Beim LE Legacy Pairing zweier Geräte generieren beide einen 128 Bit langen Temporary Key (TK), der bei der Authentifizierung genutzt wird, um den STK zu generieren und die Verbindung zu verschlüsseln. In Just Works wird der TK auf null gesetzt. Bei der Methode Passkey Entry ist der TK die besagte zufällig generierte sechstellige Dezimalzahl, die bereits mit 20 Bit dargestellt werden kann, weswegen die restlichen Bit des TK auf null gesetzt werden müssen. Dagegen kann bei OOB auf diese Einschränkung verzichtet werden, wodurch der TK wahrhaftig eine Länge von 128 Bit besitzt.

Das Gerät, welches das Pairing einleitet (Master), generiert eine zufällige 128 Bit große Nummer *Mrand* und ermittelt den 128 Bit großen Bestätigungswert *Mconfirm* mit der Confirm Value Generation Function c1 [49]. Zur Berechnung von *Mconfirm* erhält die Funktion c1

folgende Eingabewerte entsprechend Gl. 1 [50].

$$\begin{aligned} \text{Mconfirm} = \text{c1}(&\text{TK, Mrand,} \\ &\text{Pairing Request Command, Pairing Response Command,} \\ &\text{Adresstyp des Masters, Adresse des Masters,} \\ &\text{Adresstyp des Slaves, Adresse des Slaves)} \end{aligned} \tag{1}$$

Ebenso führt das antwortende Gerät (Slave) diese Schritte durch, wobei *Mrand* als *Srand* bezeichnet wird und *Mconfirm* als *Sconfirm*. Die Eingabewerte der Funktion c1 zur Berechnung von *Sconfirm*sind demnach analog zu denen von *Mconfirm*. Danach findet entsprechend Abb. 6 folgender Austausch statt.

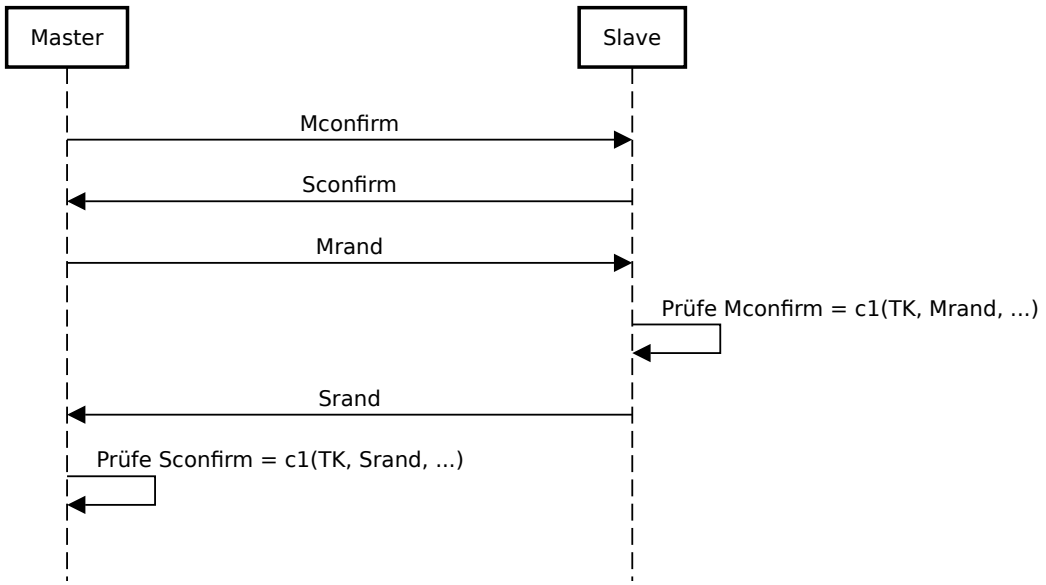

Abb. 6: Austausch von *Mconfirm*, *Sconfirm*, *Mrand* und *Srand* zwischen Master und Slave [50]

Anschließend wird der STK mit der Funktion s1 [51] entsprechend Gl 2 [50] berechnet.

$$\text{STK} = \text{s1}(\text{TK, Srand, Mrand}) \tag{2}$$

Demnach kann bei der Methode Passkey Entry kein ausreichender Schutz gegen passives Abhören geboten werden, da der TK nur wenig mögliche Werte annehmen kann. Ist die vereinbarte Schlüsselgröße kleiner als 128 Bit, werden die überschüssigen Bit beginnend bei dem Bit mit dem höchsten Stellenwert auf null gesetzt. Der STK wird nun zur Verschlüsselung der Verbindung genutzt. [50]

LE Secure Connections Pairing: Schlüssel und deren Generierung

Beim LE Secure Connections Pairing wird ein Long Term Key (LTK) erstellt. Zuvor erstellen beide Geräte jeweils ein ECDH-Schlüsselpaar (PK - Public Key, SK - Private Key) und tauschen ihre Public Keys aus. Danach berechnet jedes Gerät den Diffie-Hellman-Schlüssel aus seinem Private Key und dem Public Key des Anderen. Durch den Diffie-Hellman-Schlüssel kennen beide Parteien ein gemeinsames Geheimnis, mit dem sie den weiteren Datenaustausch zur Authentifizierung verschlüsseln können. Die Authentifizierung ist notwendig, da

der ECDH-Schlüsselaustausch zwar resistent gegen passives Abhören ist, jedoch nicht gegen MITM-Angriffe. [52]

Diese Authentifizierung wird mit den Pairing Methoden Numeric Comparison, Just Works, OOB und Passkey Entry ermöglicht. Jedoch unterscheiden diese sich aus funktionaler Sicht (nicht aus Nutzersicht) zum LE Legacy Pairing durch komplexere Verfahren. Letztendlich lässt sich für die vier Pairing-Methoden Folgendes zusammenfassen. Numeric Comparison signalisiert dem Nutzer mit einer Wahrscheinlichkeit von 0,999999 einen stattfindenden MITM-Angriff [53]. Just Works bietet keinen Schutz vor einem MITM-Angriff [45]. Ein MITM-Angriff während des Passkey Entry gelingt nur mit einer Wahrscheinlichkeit von 0,000001 [54]. Wie anfällig OOB für Angriffe ist hängt von der verwendeten OOB-Technologie ab [55].

Nach der Ausführung einer Pairing-Methode wird der LTK als Teilergebnis der Funktion f5 [56] mit den Eingabewerten Diffie-Hellman-Key, ein Nonce des Masters, ein Nonce des Slaves und der Adresse des Masters und Slaves ermittelt [57].

2.5.4.3 Pairing: Phase 3

Wurde der STK bzw. LTK generiert, wird dieser genutzt, um die Verbindung zu verschlüsseln. Nun können in der dritten Phase transportspezifische Schlüssel ausgetauscht werden. Z.B. wird der Identity Resolving Key (IRK) zur Generierung und Auflösung von zufälligen Adressen verwendet und der Connection Signature Resolving Key (CSRK) zur Signatur von Daten und Überprüfung von Signaturen.

2.6 Sicherheit

Obwohl Bluetooth LE mit dem Privacy Feature und dem Security Manager (vor allem ab Bluetooth 4.2) einige Optionen für eine sichere Infrastruktur bietet, bleiben Probleme offen, die betrachtet werden müssen.

Ein erfolgreicher Angriff auf zwei Geräte, die mittels BLE verschlüsselt kommunizieren, in Form von passivem Abhören oder eines MITM-Angriffs kann beim Aufbau einer Verbindung, also dem Pairing (siehe Sektion 2.5.4), auftreten, da hier die Schlüssel ausgetauscht werden. In Tabelle 5 wird dargestellt welchen Schutz die Pairing-Methoden des LE Legacy Pairing und des LE Secure Connections Pairing gegen passives Abhören und MITM-Angriffe bieten.

	Schutz gegen Passives Abhören		**Schutz gegen MITM-Angriff**	
	LE Legacy	**LE Secure Connections**	**LE Legacy**	**LE Secure Connections**
Numeric Comparison	-	Ja	-	Ja [53]
Just Works	Nein [58]	Ja [45]	Nein [58]	Nein [45]
Out of Band	abhängig von OOB-Technologie [59] [55]			
Passkey Entry	Nein [48]	Ja	Ja [48]	Ja [54]

Tab. 5: Schutz durch Pairing-Methoden vor passivem Abhören und MITM-Angriffen; Der Bindestrich symbolisiert, dass diese Methode nicht verfügbar ist. LE Secure Connections ist aufgrund des ECDH-Schlüsselaustausches [52] generell vor passivem Abhören geschützt.

Dabei ist zu beachten, dass die Methode Numeric Comparison dem Nutzer mit einer Wahrscheinlichkeit von 0,999999 einen stattfindenden MITM-Angriff signalisiert [53], bevor dieser

das Pairing fortsetzt, und dass die Methode Passkey Entry nur mit einer Wahrscheinlichkeit von 0,000001 anfällig für einen MITM-Angriff ist [48] [54]. Die Methode OOB des LE Legacy Pairing ist bei einer sicheren OOB-Technologie ebenfalls mit einer Wahrscheinlichkeit von 0,000001 oder kleiner (je nach Schlüsselgröße) anfällig gegen MITM-Angriffe [59].

Obwohl die Methoden Numeric Comparison und Passkey Entry des LE Secure Connections Pairing eine beachtliche Sicherheit bieten, ist laut den Bluetooth-Spezifikationen 4.0 bis 5.2 jede dieser Bluetooth-Versionen in der Lage das LE Legacy Pairing auszuführen [60] [61], welches wiederum anfälliger ist (OOB ausgenommen). Vermutlich kann dies durch die letztendlichen Entwickler einer Bluetooth-Software bzw. -Hardware oder durch den Anwender selbst eingeschränkt werden. Falls dies nicht umgesetzt wird, stellt die Möglichkeit einer Rückstufung auf LE Legacy Pairing ein Sicherheitsrisiko dar.

Dabei sei zu erwähnen, dass jede Methode bestimmte Möglichkeiten zur Ein- und Ausgabe von den zu verwendenten Geräten voraussetzt. Ist es für den Anwender nicht möglich diese Voraussetzungen in den Geräten zu implementieren, kann BLE selbst keine Schutz bieten.

Ein weiteres Problem ist die Sicherheit auf der Anwendungsebene. Unterstützt ein System weitere Anwendungen, auf die der Anwender keinen Zugriff hat, könnte es möglich sein, dass eine solche fremde Anwendung auf Daten zugreifen kann, die nur für die Anwendung des Anwenders bestimmt sind. Die Möglichkeit dazu besteht, da BLE zu übertragende Daten innerhalb des Controllers auf der Ebene des Link Layer verschlüsselt [62] [40].

Ein Beispiel für ein solches System ist das Betriebssystem Android von der Open Handset Alliance. Auf Androids Webpräsenz für BLE wird darauf hingewiesen, dass BLE keine Sicherheit auf Anwendungsebene liefert: "When a user pairs their device with another device using BLE, the data that's communicated between the two devices is accessible to all apps on the user's device"[63].

Den Beweis dafür, dass Daten, die durch BLE über den Link Layer verschlüsselt oder unverschlüsselt übertragen werden, von anderen Apps ausgelesen werden können, liefert eine Studie der Royal Holloway University of London [64]. Diese zeigt, wie innerhalb des Android Betriebssystems eine fremde Anwendung Daten über das Protokoll ATT bzw. GATT empfangen kann, die theoretisch für eine andere Anwendung bestimmt sind.

3 Grundlagen zu Transport Layer Security

3.1 Zertifikate

3.2 Algorithmen

3.3 Protokoll

3.4 Sicherheit

4 Infrastruktur

Die Infrastruktur beschreibt eine allgemeine Lösung, um eine sichere Kommunikation mittels Bluetooth Low Energy zu gewährleisten. Bis auf den Fakt, dass ein Mikrocontroller und ein Smartphone miteinander kommunizieren, sind die Systeme der beiden Kommunikationsparteien nicht von Relevanz. Dabei ist die Lösung auf weitere Konstellationen der Systeme anwendbar, solang jedes System über ein Bluetooth-Modul (mind. der Bluetooth-Version 4.0) verfügt und eine Software-Bibliothek für Transport Layer Security (TLS) der Version 1.2 oder höher unterstützt. Somit ist die Infrastruktur unabhängig von dem in der Sektion X beschriebenen Projekt SteigtUM.

Obwohl die Sicherheit in der Datenübertragung keine allumfassende Definition besitzt, lassen sich für diese trotzdem essenzielle Aspekte formulieren. Nach X sind diese Vertraulichkeit, Datenintegrität und Authentzität.

Vertraulichkeit bedeutet, dass "Übertragene Daten [...] nur berechtigten Instanzen zugänglich sein [sollen], d.h. keine unbefugte dritte Partei soll an den Inhalt von übertragenen Nachrichten gelangen können".

Die Datenintegrität trägt folgende Bedeutung. "Für den Empfänger muss eindeutig erkennbar sein, ob Daten während ihrer Übertragung unbefugt geändert wurden".

Authentizität teilt sich in zwei Punkte auf. Zum einen soll Ëine Instanz [...] einer an deren ihre Identität zweifelsfrei nachweisen können (Identitätsnachweis bzw. Authentifizierung der Instanz)". Zum anderen ßoll überprüft werden können, ob eine Nachricht von einer bestimmten Instanz stammt (Authentizität der Daten)".

4.1 Topologie

Die Topologie der Infrastruktur beschränkt sich auf das Minimum an Kommunikationsparteien und ist unabhängig von der Anwendung.

Dem Thema zufolge sollen ein Mikrocontroller und ein Smartphone sicher Daten austauschen. Um dies zu bewerkstelligen, wird über den Transport der Daten durch Bluetooth Low Energy (BLE) das Verschlüsselungsprotokoll TLS verwendet (siehe Sektion X). Dementsprechend ist, wie in Abbildung X zu sehen , neben dem Mikrocontroller und dem Smartphone eine Zertifizierungsstelle notwendig. In welcher Form diese auftritt ist von der Anwendung abhängig.

Für einen seriösen Anwendungsfall sollte die Zertifizierungsstelle in Form eines Servers existieren, der dem Mikrocontroller und Smartphone in regelmäßigen Abständen (z.B. jährlich) jeweils ein Zertifikat ausstellt. Mit diesen Zertifikaten und der Kenntnis über das Root-Zertifikat können Mikrocontroller und Smartphone sich gegenseitig authentifizieren und somit die Grundlage für eine sichere Kommunikation bilden.

Beispielsweise könnte es für einen privaten Anwendungsfall ausreichend sein, die Zertifizierungsstelle nicht als dauerhaft betriebenen Server darzustellen, sondern lediglich ein Root-Zertifikat zu erstellen und mit diesem dem Mikrocontroller und Smartphone jeweils ein digitales Zertifikat auszustellen.

Die Rolle der Partei, die die Zertifikate für Mikrocontroller und Smartphone austellt muss nicht zwingend eine Zertifizierungsstelle sein, sondern könnte auch eine Entität sein, der von einer Zertifizierungsstelle ein Zwischenzertifikat ausgestellt wurde.

4.2 Transport

Eine der grundlegensten Bedingungen dieser Arbeit ist, dass BLE als Technologie zum übertragen der Daten zwischen Smartphone und Mikrocontroller genutzt werden soll. Demnach stellt sich die Frage nach einem geeigneten Transport der Daten innerhalb des BLE-Protokollstapels.

Es existieren mehrere Referenzmodelle für Kommunikation innerhalb von Rechnernetzen. Geläufig sind das TCP/IP-Referenzmodell (Transport Control Protocol / Internet Protocol), das OSI-Referenzmodell (Open Systems Interconnection) und ein hybrides Referenzmodell aus diesen beiden. Alle drei setzen unterschiedliche Anzahlen von Schichten voraus, von denen sich einige gleichen oder ähneln und andere nicht. Die Eigenschaften der Transportschicht sind bei den drei Referenzmodellen identisch.

Der Protokollstapel des BLE-Controller besteht aus dem Physical Layer, der die Übertragung der Daten auf physischer Ebene definiert, und dem Link Layer, der Pakete

segmente, ports, verbindungslos/verbindungsorientiert, Flusskontrolle, verlustfreie übertragung/sicherstellung, Reihenfolge

4.3 Sicherheit

4.4 Verbindungsaufbau

5 Implementierung

5.1 Ziel der Implementierung

5.2 Topologie

5.3 Hardware und Software

5.4 Transport und Sicherheit

5.5 Ausleihprozess

5.5.1 Verbindungsaufbau

5.5.2 Beenden des Ausleihprozesses

6 Ausblick

7 Zusammenfassung

Literatur

[1] Institute of Electrical und Electronics Engineers. *IEEE 802.15.1-2002 - IEEE Standard for Telecommunications and Information Exchange Between Systems - LAN/MAN - Specific Requirements - Part 15: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Wireless Personal Area Networks (WPANs)*. URL: https://standards.ieee.org/standard/802_15_1-2002.html (besucht am 28.06.2021).

[2] Bluetooth Special Interest Group. *The Bluetooth Range Estimator*. URL: https://www.bluetooth.com/learn-about-bluetooth/key-attributes/range/ (besucht am 28.06.2021).

[3] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 1. 30. Juni 2010. Kap. Part A, 1.1 Overview of BR/EDR Operation, S. 18 (PDF S. 124). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[4] Bluetooth Special Interest Group. „Bluetooth Core Specification Version 5.2“. In: Bd. Vol 1. 31. Dez. 2019. Kap. Part A, 1.1 Overview of BR/EDR Operation, S. 188 (PDF S. 188). URL: https://www.bluetooth.com/specifications/specs/core-specification/.

[5] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 1. 30. Juni 2010. Kap. Part A, 1 General Description, S. 17 (PDF S. 123). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[6] Bluetooth Special Interest Group. „Bluetooth Core Specification Version 5.2“. In: Bd. Vol 1. 31. Dez. 2019. Kap. Part A, 1 General Description, S. 187 (PDF S. 187). URL: https://www.bluetooth.com/specifications/specs/core-specification/.

[7] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 1. 30. Juni 2010. Kap. Part A, 1.2 Overview of Bluetooth Low Energy Operation, S. 20 (PDF S. 126). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[8] Bluetooth Special Interest Group. „Bluetooth Core Specification Version 5.2“. In: Bd. Vol 1. 31. Dez. 2019. Kap. Part A, 1.2 Overview of Bluetooth Low Energy Operation, S. 190 (PDF S. 190). URL: https://www.bluetooth.com/specifications/specs/core-specification/.

[9] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 1. 30. Juni 2010. Kap. Part A, 1 General Description, Figure 1.1, Figure 1.2, S. 18 (PDF S. 124). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[10] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 1. 30. Juni 2010. Kap. Part A, 2 Core System Architecture, Figure 2.1, S. 31 (PDF S. 137). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[11] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 6. 30. Juni 2010. Kap. Part A, 2 Frequency Bands and Channel Arrangement, S. 16 - 17 (PDF S. 2180 - 2181). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[12] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 6. 30. Juni 2010. Kap. Part B, 1.4.1 Advertising and Data Channel Indexes, S. 35 (PDF S. 2199). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[13] Bluetooth Special Interest Group. *How Bluetooth Technology Uses Adaptive Frequency Hopping to Overcome Packet Interference*. URL: how-bluetooth-technology-uses-adaptive-frequency-hopping-to-overcome-packet-interference/ (besucht am 28. 06. 2021).

[14] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 6. 30. Juni 2010. Kap. Part B, 2.1 Packet Format, S. 36-37 (PDF S. 2200-2201). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[15] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 6. 30. Juni 2010. Kap. Part B, 3.2 Data Whitening, S. 53-54 (PDF S. 2217-2218). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[16] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 6. 30. Juni 2010. Kap. Part B, 2.3 Advertising Channel PDU, S. 37-44 (PDF S. 2201-2208). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[17] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 6. 30. Juni 2010. Kap. Part B, 2.4 Data Channel PDU, S. 44-45 (PDF S. 2208-2209). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[18] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 6. 30. Juni 2010. Kap. Part B, 4.5.9 Acknowledgement and Flow Control, S. 75-77 (PDF S. 2239-2241). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[19] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 1. 30. Juni 2010. Kap. Part A, 3.5.4.6 LE Asynchronous Connection (LE ACL), 3.5.4.7 LE Advertising Broadcast (ADVB), S. 68-69 (PDF S. 174-175). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[20] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 1. 30. Juni 2010. Kap. Part A, 3.5.5.2 LE Logical Links, S. 70-71 (PDF S. 176-177). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[21] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part A, 2.4 Modes of Operation, S. 43 (PDF S. 1735). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[22] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 3. 30. Juni 2010. Kap. Part A, 2.4 Modes of Operation, S. 41 (PDF S. 1401). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[23] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 3. 30. Juni 2010. Kap. Part F, 3.1 Introduction, S. 475 (PDF S. 1835). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[24] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 3. 30. Juni 2010. Kap. Part F, 3.4.4.3 Read Request, 3.4.4.4 Read Response, S. 494 - 495 (PDF S. 1854 - 1855). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[25] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 3. 30. Juni 2010. Kap. Part F, 3.4.5.1 Write Request, 3.4.5.2 Write Response, S. 501 - 503 (PDF S. 1861 - 1863). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[26] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 3. 30. Juni 2010. Kap. Part G, 2.5 Attribute Protocol, S. 528 - 529 (PDF S. 1888 - 1889). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[27] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 3. 30. Juni 2010. Kap. Part G, 2.6.1 Overview, S. 532 (PDF S. 1892). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[28] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 3. 30. Juni 2010. Kap. Part C, 2.2.2 Roles when Operating over an LE Physical Channel, S. 278 - 279 (PDF S. 1638 - 1639). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[29] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 3. 30. Juni 2010. Kap. Part C, 9.1 Broadcast Mode and Observation Procedure, S. 335 - 337 (PDF S. 1695 - 1697). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[30] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 3. 30. Juni 2010. Kap. Part C, 9.2 Discovery Modes and Procedures, S. 337 (PDF S. 1697). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[31] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 3. 30. Juni 2010. Kap. Part C, 9.3 Connection Modes and Procedures, S. 344 - 359 (PDF S. 1704 - 1718). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[32] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part C, 9.4 Bonding Modes and Procedures, S. 368 - 370 (PDF S. 2060 - 2062). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[33] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 6. 2. Dez. 2014. Kap. Part B, 1.3.1 Public Device Address, S. 33 (PDF S. 2576). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[34] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part H, 2.2.2 Random Address Hash function ah, S. 595 (PDF S. 2287). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[35] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 6. 2. Dez. 2014. Kap. Part B, 1.3.2 Random Device Address, S. 34 - 36 (PDF S. 2577 - 2579). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[36] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part C, 10.7.3 Privacy Feature in a Broadcaster, 10.7.4 Privacy Feature in an Observer, S. 386 - 387 (PDF S. 2078 - 2079). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[37] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part C, 10.7.1 Privacy Feature in a Peripheral, S. 385 (PDF S. 2077). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[38] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part C, 10.7.2 Privacy Feature in a Central, S. 386 (PDF S. 2078). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[39] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 3. 30. Juni 2010. Kap. Part H, 1.1 Scope, Figure 1.1, S. 598 (PDF S. 1958). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[40] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 6. 30. Juni 2010. Kap. Part E, 1 Encryption and Authentication Overview, S. 121 (PDF S. 2285). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[41] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 3. 30. Juni 2010. Kap. Part H, 2.3.2 IO Capabilities, Table 2.1, S. 605 (PDF S. 1965). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[42] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 3. 30. Juni 2010. Kap. Part H, 2.3.2 IO Capabilities, Table 2.2, S. 605 (PDF S. 1965). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[43] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 1. 2. Dez. 2014. Kap. Part A, 5.4.1 Association Models, S. 93 (PDF S. 248). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[44] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 1. 2. Dez. 2014. Kap. Part A, 5.2.4.1 Numeric Comparison, S. 89 - 90 (PDF S. 244 - 245). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[45] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 1. 2. Dez. 2014. Kap. Part A, 5.2.4.2 Just Works, S. 90 (PDF S. 245). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[46] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 1. 2. Dez. 2014. Kap. Part A, 5.2.4.3 Out of Band, S. 91 (PDF S. 246). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[47] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 1. 2. Dez. 2014. Kap. Part A, 5.2.4.4 Passkey Entry, S. 91 - 92 (PDF S. 246 - 247). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[48] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part H, 2.3.5.3 LE Legacy Pairing - Passkey Entry, S. 612 (PDF S. 2304). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[49] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part H, 2.2.3 Confirm value generation function c1 for LE Legacy Pairing, S. 596 (PDF S. 2288). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[50] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part H, 2.3.5.5 LE Legacy Pairing Phase 2, S. 613 - 614 (PDF S. 2305 - 2306). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[51] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part H, 2.2.4 Key generation function s1 for LE Legacy Pairing, S. 598 (PDF S. 2290). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[52] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part H, 2.3.5.6.1 Public Key Exchange, S. 615 (PDF S. 2307). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[53] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part H, 2.3.5.6.2 Authentication Stage 1 – Just Works or Numeric Comparison, S. 617 (PDF S. 2309). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[54] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part H, 2.3.5.6.3 Authentication Stage 1 – Passkey Entry, S. 619 (PDF S. 2311). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[55] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part H, 2.3.5.6.4 Authentication Stage 1 – Out of Band, S. 620 - 621 (PDF S. 2312 - 2313). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[56] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part H, 2.2.7 LE Secure Connections Key Generation Function f5, S. 600 - 601 (PDF S. 2292-2293). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[57] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part H, 2.3.5.6.5 Authentication Stage 2 and Long Term Key Calculation, S. 622 (PDF S. 2314). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[58] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part H, 2.3.5.2 LE Legacy Pairing - Just Works, S. 612 (PDF S. 2304). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[59] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 3. 2. Dez. 2014. Kap. Part H, 2.3.5.4 Out of Band, S. 613 (PDF S. 2305). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[60] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.2“. In: Bd. Vol 1. 2. Dez. 2014. Kap. Part A, 5.4 LE SECURITY, S. 613 (PDF S. 2305). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.

[61] Bluetooth Special Interest Group. „Bluetooth Core Specification Version 5.2“. In: Bd. Vol 1. 31. Dez. 2019. Kap. Part A, 5.4 LE Security, S. 277 (PDF S. 277). URL: https://www.bluetooth.com/specifications/specs/core-specification/.

[62] Bluetooth Special Interest Group. „Bluetooth Specification Version 4.0“. In: Bd. Vol 1. 30. Juni 2010. Kap. Part A, 5.2.3 Encryption, S. 90 (PDF S. 196). URL: https://www.bluetooth.com/specifications/specs/core-specification-4-0/.

[63] Bluetooth Special Interest Group. *Bluetooth low energy*. URL: https://developer.android.com/guide/topics/connectivity/bluetooth-le (besucht am 28. 06. 2021).

[64] Pallavi Sivakumaran und Jorge Blasco. „A Study of the Feasibility of Co-located App Attacks against BLE and a Large-Scale Analysis of the Current Application-Layer Security Landscape“. In: *28th USENIX Security Symposium (USENIX Security 19)*. Santa Clara, CA: USENIX Association, Aug. 2019, S. 3–5. ISBN: 978-1-939133-06-9. URL: https://www.usenix.org/conference/usenixsecurity19/presentation/sivakumaran (besucht am 28. 06. 2021).