Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Michael Rifkin

Design und Entwicklung eines Mikrocontroller-gesteuerten Bluetooth-basierten Zugangskontrollsystems mit einer auf Android aufbauenden Verwaltungssoftware

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Michael Rifkin

Design und Entwicklung eines Mikrocontroller-gesteuerten Bluetooth-basierten Zugangskontrollsystems mit einer auf Android aufbauenden Verwaltungssoftware

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung

im Studiengang Bachelor of Science Technische Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. rer. nat. Hans Heinrich Heitmann
Zweitgutachter: Prof. Dr.-Ing. Andreas Meisel

Eingereicht am: 7. Februar 2012

Michael Rifkin

Thema der Arbeit

Design und Entwicklung eines Mikrocontroller-gesteuerten Bluetooth-basierten Zugangskontrollsystems mit einer auf Android aufbauenden Verwaltungssoftware

Stichworte

Bluetooth, Server, Angriff, Sicherheit, secure simple pairing, Link Key, pairing, Android, App, Development-Board Alarm, Alarmsystem, Alarmsteuerung, Zugangskontrollsystem, Überwachung, Benutzererkennung, SSP, STM32F4, HCI, L2CAP, RFCOMM, SDP, Link Manager, Smartphone, man in the middle, MITM, Lauschangriff, PIN, Numeric Comparison

Kurzzusammenfassung

Dieses Dokument behandelt das Design und die Entwicklung eines Mikrocontroller-gesteuerten, Bluetooth-basierten Zugangskontrollsystems mit einer auf Android aufbauenden Verwaltungssoftware. Das Ziel ist es ein Gesamtsystem zu entwickeln, welches eine sichere Identifikation und Überwachung authentifizierter Benutzer, anhand der von ihnen eingesetzten Bluetooth-Hardware, erlaubt. Geringe Hardware-Kosten, leichte Installation, sowie eine zuverlässige Benutzererkennung sind Merkmale, welche das Gesamtsystem auszeichnen.

Michael Rifkin

Title of the paper

Design and development of a microcontroller driven and Bluetooth based access control system with an Android based user management software

Keywords

Bluetooth, server, attack, security, secure simple pairing, Link Key, pairing, Android, App, development, board, alarm, alarm system, alarm control, access control system, supervision, user detection, SSP, STM32F4, HCI, L2CAP, RFCOMM, SDP, link manager, smart phone, man in the middle, MITM, eavesdropping, PIN, numeric comparison

Abstract

This thesis discusses the design and development of a microcontroller driven and Bluetooth based access control system with Android based user management software. The goal is to develop a complete system which allows identification and supervision of authenticated users based on Bluetooth hardware. Low hardware costs, uncomplicated installation, and a reliable user identification are characteristics that distinguish the complete system.

Inhaltsverzeichnis

Abkürzungsverzeichnis

API	Application Programming Interface
BDADDR	Bluetooth Address
BT-NINO-MITM	Bluetooth No Input No Output - Man In The Middle
DCE	Data Communication Equipment
DH-Key	Diffie-Hellman Key
DTE	Data Terminal Equipment
FPGA	Field Programmable Gate Array
GPIO	General Purpose Input/Output
HCI	Host Controller Interface
K_{ab}	Link Key
K_{init}	Initialization Key
L2CAP	Logical Link Control and Adaptation Layer Protocol
LM	Link Manager
LMP	Link Manager Protocol
MITM	Man In The Middle
NFC	Near Field Communication
PIN	Personal Identification Number
PMP	Participant in Multiple Piconet

RAM	Random Access Memory
RFCOMM	Radio Frequency Communication
SDK	Software Development Kit
SDP	Service Discovery Protocol
SIG	Special Interest Group
SSP	Secure Simple Pairing
SWD	Serial Wire Debugging
USB	Universal Serial Bus
USB-OTG	USB On-the-go

1 Einleitung

Die Verbreitung von mobilen Geräten hat sich in letzten Jahren sehr stark gesteigert. Jeder Dritte in Deutschland besitzt heute ein Smartphone. Bei den unter 30-Jährigen ist jeder Zweite im Besitz eines solchen Gerätes. Rund 88 Prozent der Bundesbürger über 14 Jahre nutzen ein Mobilfunktelefon.[BITKOM (2012)] Durch die Einführung Googles Mobilbetriebssystems Android, sind die Preise für mobile Geräte wie zum Beispiel Smartphones oder Tablets stark gesunken. [Roger Cheng (2011)] Ein einzelnes Smratphone ersetzt heute nicht nur das Telefon, sondern auch ein Navigationsgerät, Diktiergerät, MP3-Player oder auch eine Spielkonsole. Die Geräte werden immer schneller und bieten inzwischen eine Vielzahl von Funktionen, welche den Benutzern das Leben erleichtern. Das große Touch-Display eines Smartphones hat es ermöglicht, bequem und unkompliziert mobile Dienste in Anspruch zu nehmen. Ob Fahrplanauskunft, Bezahlung, oder ein Preisvergleich, alle diese Aufgaben können heute blitzschnell erledigt werden.

1.1 Motivation

Nicht nur die bequeme Benutzung des mobilen Internets und die eingebaute Hardware machen Mobilfunktelefone heute so interessant. Da Mobilfunktelefone so weit verbreitet sind und meist von ihren Benutzern mitgeführt werden, liegt es nahe, sie auch als Steuergeräte einzusetzen. Zurzeit sind vergleichsweise wenige Lösungen zur Steuerung externer Hardware auf dem Markt. Fernsehgeräte, Beleuchtung, Steckdosen, oder Türen können heute bereits ganz bequem mit einem Smartphone gesteuert werden. Besonders im Sicherheitsbereich gibt es viele Anwendungsgebiete, für welche es noch keine technischen Lösungen mit Smartphone-Steuerung gibt. Da sehr viele Menschen bereits ein Mobilfunktelefon nutzen, ist es in meinen Augen sinnvoll, das Gerät zur Identifizierung in einem Sicherheitssystem zu nutzen. Es ist keine neue Idee und es gibt bereits Produkte auf dem Markt, welche diese Idee umsetzten. So gibt es zum Beispiel bereits eine Entwicklung der Firmen „Nissan“, „Sharp“ und „NTT Docomo“, welche es ermöglicht, den Autoschlüssel durch ein Handy zu ersetzen. Dabei wurde Nissans Intelligent-Key-Technik in einen Handy-Prototypen integriert. Diese Technik war zuvor bereits in fast einer Million Autos eingebaut worden. [PcWelt (2008)] Eine interessante Entwicklung ist auch das „Airkey-Schließzylinder“. Eine neue Zutrittslösung, welche von den Firmen „EVVA Sicherheitstechnologie

GmbH“ und „Rise F&E GmbH“ in Zusammenarbeit mit der Technischen Universität Wien entwickelt und getestet wurde. [sicherheit.info (2012)] Jedoch basieren diese Lösungen auf der Near Field Communication (NFC) Technologie. Leider sind jedoch NFC Chips nicht in allen Handys vertreten. Auch heute werden die preiswerten Modelle der großen Hersteller wie z.B. „HTC“ oder „Samsung“ nicht mit NFC-Chips bestückt. Dieser Umstand schränkt die Anzahl der möglichen Benutzer erheblich ein. Um ein Mobilfunktelefon zur Identifizierung zu nutzen, bietet es sich aus diesem Grund an, das Bluetooth-Protokoll einzusetzen.

1.2 Zielsetzung

Das Ziel dieser Arbeit ist es, ein Mikrocontroller-gesteuertes, Bluetooth-basiertes Zugangskontroll- und Verwaltungssystem zu entwickeln. Dieses System muss authentifizierte Benutzer anhand ihrer Bluetooth-Hardware sicher erkennen und eine Alarmanlage deaktivieren, sobald der Benutzer in der Bluetooth-Reichweite des Zugangskontrollsystems ist. Sobald kein authentifizierter Benutzer mehr in Reichweite ist, muss die Alarmanlage sofort wieder aktiviert werden. Ein Smartphone wird eingesetzt, um die Benutzer zu verwalten. Die Verwaltung der Benutzer sollte dabei durch einen Master-User erfolgen. Das Zugangskontrollsystem sollte für eine Installation an einem überwachten Objekt kostengünstig und möglichst klein sein. In meiner Arbeit möchte ich nachweisen, dass ein zuverlässiges Zugangskontrollsystem auf Basis des Bluetooth-Protokolls realisierbar ist. Als große Herausforderung sehe ich dabei, die Sicherheit des Zugangskontrollsystems sowie eine zuverlässige Benutzeridentifikation zu gewährleisten.

2 Technische Grundlagen des Bluetooth-Protokolls

2.1 Einführung in Bluetooth

Bluetooth ist ein Industriestandard für die Datenübertragung zwischen Geräten über kurze Distanz. Im Jahr 1999 wurde die erste Version der Bluetooth-Spezifikation durch die Bluetooth-Special Interest Group (SIG) veröffentlicht. [Holtkamp (2003)] Der Einsatz von Bluetooth ist heute nicht nur bei Verbrauchern, zum Beispiel bei Smartphones und Tablets, sondern auch in der Industrie weit verbreitet. Die Bezeichnung Bluetooth geht auf den dänischen König Harald I Blaatand (Blauzahn) zurück, welcher Dänemark christianisierte und Norwegen und Dänemark vereinigte. [Sikora (2008)]

2.2 Die Architektur von Bluetooth-Netzwerken

Die Grundeinheit eines Bluetooth-Netzwerkes bildet ein Piconet.

Definition 1 (*Piconet*)

> *„A collection of devices occupying a shared physical channel where one of the devices is the Piconet Master and the remaining devices are connected to it.“*
> *[BTSPEZ21 (2007)]*

In einem Piconet gibt es stets ein Bluetooth-Gerät, welches die Piconet-Master-Rolle übernimmt. Außer dem Master-Gerät können bis zu 7 Slave-Geräte in einem Piconet aktiv sein. Die Slave-Geräte können nicht direkt miteinander kommunizieren.

Damit weitere Bluetooth-Geräte an einem Piconet teilnehmen können, müssen diese geparkt beziehungsweise in den „PARKED“-Zustand versetzt werden. Geparkte Geräte sind auch ein Teil des Piconet, können aber nur auf Anfrage aktiv an der Kommunikation teilnehmen. [Holtkamp (2003)], [BTSPEZ21 (2007)]

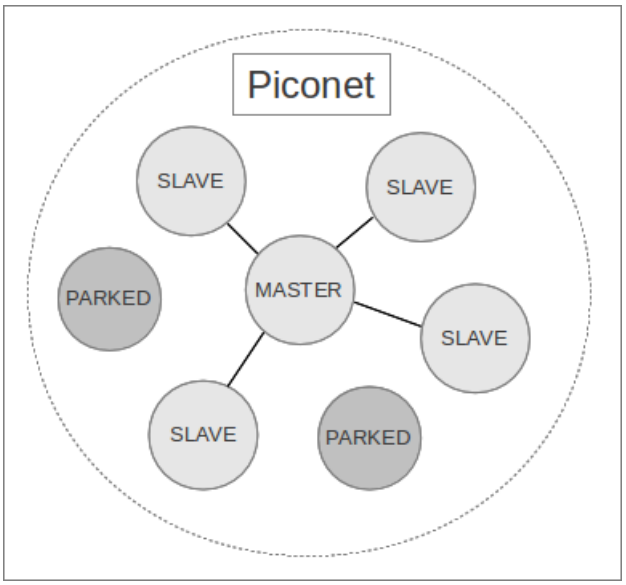

Abbildung 2.1: Die Abbildung zeigt ein Piconet mit aktiven Master- und Slave-Geräten, sowie inaktiven Geräten im „PARKED"-Zustand.

Die Verbindung von mehreren Piconets wird als Scatternet bezeichnet.

Definition 2 (*Scatternet*)

> *„Two or more piconets that include one or more devices acting as Participant in Multiple Piconet (PMP)s."*
> *[BTSPEZ21 (2007)]*

Definition 3 (*Participant in Multiple Piconet*)

> *„A device that is con- currently a member of more than one piconet, which it achieves using time division multiplexing (TDM) to interleave its activity on each piconet physical channel."*
> *[BTSPEZ21 (2007)]*

Ein Teilnehmer eines Piconet kann gleichzeitig ein Teilnehmer mehrerer weiterer Piconets sein. Ein Gerät kann dabei als Slave-Gerät in mehreren Piconets, aber als Master-Gerät nur eines Piconets fungieren. Sowohl ein Master-Gerät als auch ein Slave-Gerät können parallel Slave-Geräte in anderen Piconets sein. Jedes Piconet eines Scatternet hat eine eigene Frequenzsprungfolge. Bei der Kommunikation mit anderen Teilnehmern, muss immer die Frequenzsprungfolge des Kommunikationspartners beachtet werden. Es ist zu beachten, dass Scatternets nicht von allen Bluetooth-Geräten unterstützt werden. [Holtkamp (2003)], [BTSPEZ21 (2007)]

2.3 Das Frequenzsprungverfahren

Der Bluetooth-Standard arbeitet in einem lizenzfrei verfügbaren ISM-Band (Industrial, Scientific and Medical Band) zwischen 2.400-2.4835 GHz. [BTSPEZ21 (2007)] Somit können Bluetooth-Geräte weltweit zulassungsfrei betrieben werden. Da es potentiell zu einer sehr hohen Kanalauslastung kommen kann, muss entsprechend den Vorschriften der Regulierungsbehörden ein Frequenzspreizverfahren von Bluetooth-Geräten unterstützt werden. [Sikora (2008)]

Um den Regelungen zu entsprechen, um Robustheit gegenüber Störungen durch andere Geräte wie zum Beispiel WLAN-Router oder Mikrowellen zu erreichen und auch um mögliche Angriffe zu erschweren, wird das Frequenzsprungverfahren (Frequency Hopping) eingesetzt.

Dabei wird das gesamte Frequenzband in gleich große Sprungfrequenzen je 1 MHz eingeteilt. Da die Regelungen je nach Region unterschiedlich sind, unterstützt der Bluetooth-Standard zwei Betriebsmodi. Für Nordamerika und Europa steht ein Modus mit 79 Sprungfrequenzen zur Verfügung. Ein zweiter Modus mit 23 Sprungfrequenzen steht für Länder wie zum Beispiel Japan bereit, wo das verwendbare Frequenzband kleiner ist. Beim Frequenzsprungverfahren wird 1600 Mal pro Sekunde zwischen den einzelnen Sprungfrequenzen gewechselt. [Sikora (2008)], [Hildebrandt (2004)]
Die Frequenzsprungfolge wird vom Piconet-Master vorgegeben und folgt einer Pseudozufallsfolge. Die Pseudozufallsfolge wird nach bestimmten Regeln in Abhängigkeit von der 48-Bit breiten MAC-Adresse des Master-Gerätes (nur die unteren 28 Bit werden in die Berechnung einbezogen) und der 28 Bit breiten Master-Clock berechnet. Alle Geräte innerhalb eines Piconets folgen der Frequenzsprungfolge des Master-Gerätes. Diese Frequenzsprungfolge dient auch zur Identifizierung eines Piconets. [Sikora (2008)]

Vorteile des Frequenzsprungverfahren:

1. Die übertragenen Daten bei dem Frequenzsprungverfahren sind nur mit sehr aufwendiger und teurer Technik zu erfassen. Um die übertragenen Daten zu erfassen, müsste das ganze Frequenzband abgehört werden. Dies steigert erheblich die Sicherheit einer Bluetooth-Übertragung.

2. Sollte eine bestimmte Frequenz durch andere Geräte gestört sein, so wirkt sich die Störung nicht die gesamte Zeit auf die Datenübertragung über Bluetooth aus, da alle 625 Mikrosekunden zwischen bis zu 79 Sprungfrequenzen automatisch gewechselt wird.

Weitere Details zum Frequenzsprungverfahren können der Bluetooth-Spezifikation [BTSPEZ21 (2007)] entnommen werden.

2.4 Der Bluetooth-Protokollstapel

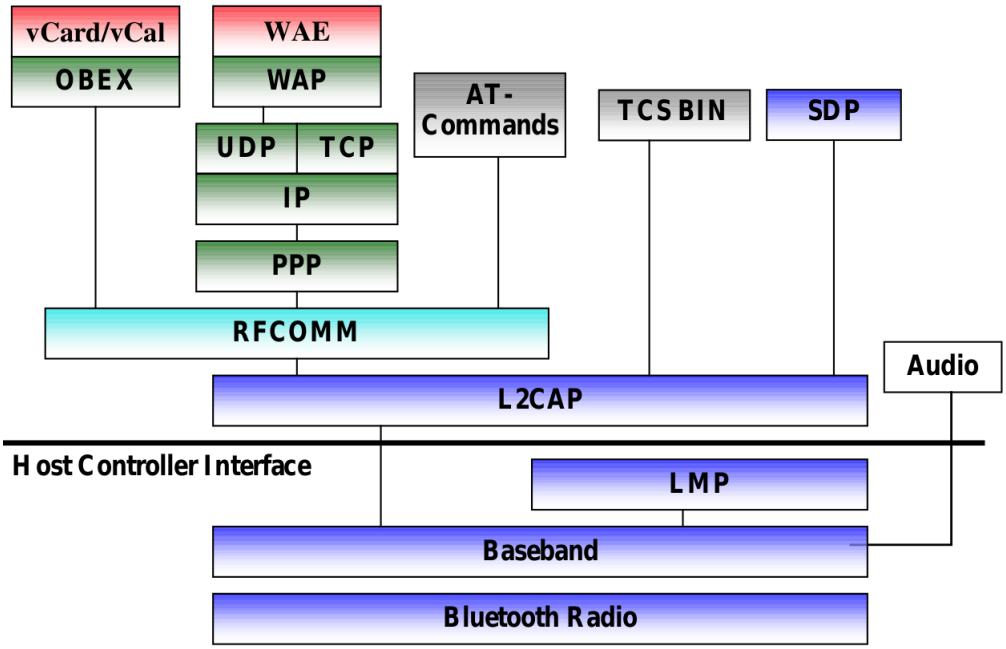

Abbildung 2.2: Der Bluetooth-Protokollstapel

Quelle: [Mettala (1999)], Figure 1 Bluetooth Protocol Stack

Definition 4 (*Bluetooth Radio*)

> *Die physikalische Realisierung des Bluetooth-Standards.*
> *[BTSPEZ21 (2007)]*

Definition 5 (*Bluetooth Baseband*)

> *„The part of the Bluetooth system that specifies or implements the medium access and physical layer pro- cedures to support the exchange of real-time voice, data information streams, and ad hoc networking between Bluetooth Devices.“*
> *[BTSPEZ21 (2007)]*

Die Bluetooth-Protokollarchitektur kann nach nach ihren Schichten in vier Gruppen eingeteilt werden.

Zusätzlich zu den oben genannten Protokollen wird in der Bluetooth-Spezifikation das Host Controller Interface (HCI) definiert.

Protocol layer	Protocols in the stack
Bluetooth Core Protocols	Baseband, LMP, L2CAP, SDP
Cable Replacement Protocol	RFCOMM
Telephony Control Protocols	TCS Binary, AT-commands
Adopted Protocols	PPP, UDP/TCP/IP, OBEX, WAP,1vCard, vCal, IrMC, WAE

Tabelle 2.1: Die Tabelle verdeutlicht die vier Protokollgruppen des Bluetooth-Protokollstapels.
Quelle: [Mettala (1999)], Table 1: The protocols and layers in the Bluetooth protocol stack

Definition 6 (*Host Controller Interface*)

„The Bluetooth Host Controller Interface (HCI) provides a command interface to the baseband controller and link manager and access to hardware status and con- trol registers. This interface provides a uniform method of accessing the Bluetooth baseband capabilities.“
[BTSPEZ21 (2007)]

Definition 7 (*Link Manager (LM)*)

„The link manager is responsible for the creation, modification and release of logical links (and, if required, their associated logical transports), as well as the update of parameters related to physical links between devices. The link manager achieves this by communicating with the link manager in remote Bluetooth devices using the Link Management Protocol (LMP).“
[BTSPEZ21 (2007)]

Definition 8 (*Link Manager Protocol (LMP)*)

„The Link Manager Protocol (LMP) is used to control and negotiate all aspects of the operation of the Bluetooth connection between two devices. This includes the set-up and control of logical transports and logical links, and for control of physical links.

The Link Manager Protocol is used to communicate between the Link Managers (LM) on the two devices which are connected by the ACL logical transport. All LMP messages shall apply solely to the physical link and associated logical links and logical transports between the sending and receiving devices.“
[BTSPEZ21 (2007)]

Die Bluetooth-Core- und Radio Protokolle werden von den meisten Bluetooth-Geräten unterstützt. Für alle anderen muss entschieden werden, ob diese für die jeweilige Aufgabe benötigt werden und somit implementiert werden müssen.

Wie man in den oberen Schichten des Bluetooth-Protokollstapels erkennen kann, greift der Bluetooth-Standard auf bereits bestehende und verbreitete Protokolle zurück. Protokolle wie zum Beispiel UDP, TCP, oder OBEX sind auch in anderen Standards vertreten. Durch ihren Einsatz wird das Portieren von bereits bestehenden Anwendungen auf den Bluetooth-Standard erheblich vereinfacht. Weitere Protokolle wie zum Beispiel HTTP, SMTP, DNS oder FTP können ebenfalls auf die bereits bestehenden Protokolle des Bluetooth-Protokollstapels aufgesetzt werden. [Mettala (1999)]

2.5 HCI - Host Controller Interface

Das Host Controller Interface bietet eine einheitliche Schnittstelle für den Zugriff auf den Bluetooth-Controller.

Definition 9 (*Bluetooth-Controller*)

> *„A sub-system containing the Bluetooth RF, baseband, resource controller, link manager, device manager and a Bluetooth HCI.“*
> *[BTSPEZ21 (2007)]*

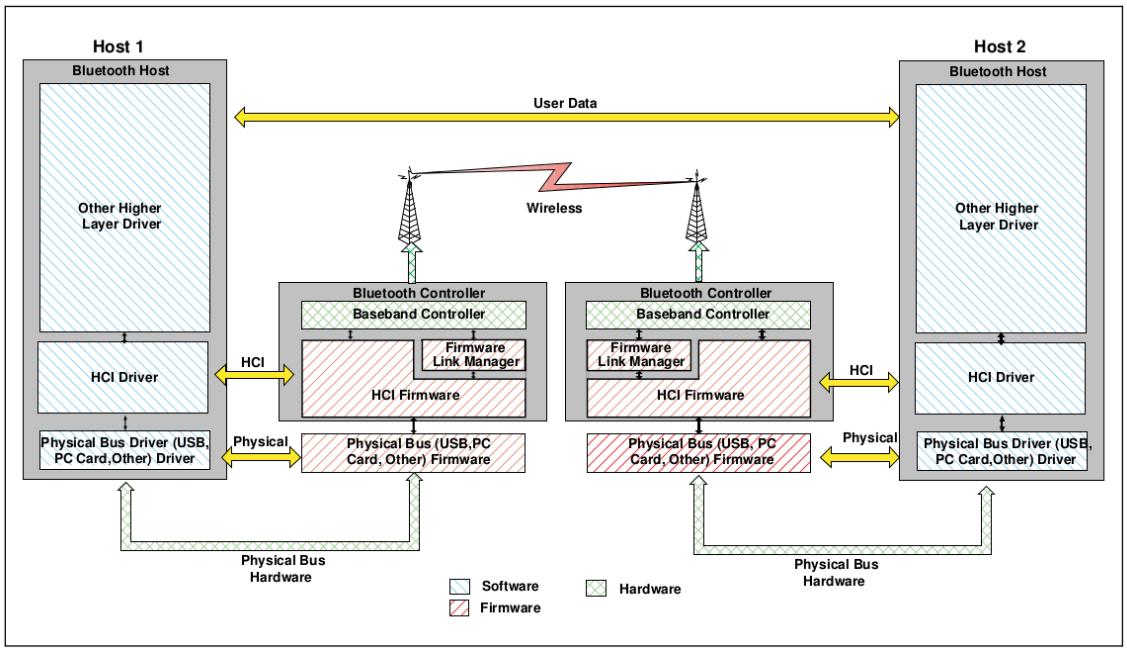

Abbildung 2.3: Punkt zu Punkt Datenübertragung der unteren Softwareschichten

Quelle: [BTSPEZ21 (2007)], Figure 1.2: End to End Overview of Lower Software Layers to Transfer Data

Das Bild verdeutlicht den Einsatz von Host Controller Interface (HCI) bei der Kommunikation zwischen zwei Bluetooth-Geräten. Der HCI-Treiber stellt eine Software dar, welche Daten mit der HCI-Firmware austauscht. Die HCI-Firmware läuft dabei auf der Hardware des Bluetooth-Gerätes. Alle anderen Software-Schichten bauen auf dem HCI-Treiber auf. Die eigentliche Datenübertragung läuft auf der Host-Control-Transport Schicht ab. Dabei handelt es sich um einen Hardware-Datenbus wie zum Beispiel Universal Serial Bus (USB). Dieser Zusammenhang wird in der Abbildung 2.3 verdeutlicht. [BTSPEZ21 (2007)]

Definition 10 (*HCI-Event*)

> *Ein HCI-Event ist eine asynchrone Nachricht vom Bluetooth-Controller zum Bluetooth-Host. Diese Nachrichten informieren den Host, über den aktuellen Status des Bluetooth-Controllers. [BTSPEZ21 (2007)]*

Definition 11 (*HCI-Event-Packet*)

> *„The HCI Event Packet is used by the Controller to notify the Host when events occur.“ [BTSPEZ21 (2007)]*

Vom Bluetooth-Controller zum Bluetooth-Host werden HCI-Packete versendet. Ein HCI-Event-Packet enthält die Information, welcher HCI-Event erfolgt ist. [BTSPEZ21 (2007)]

Definition 12 (*HCI-Befehl*)

> *Ein HCI-Befehl ist eine Nachricht, welche vom Bluetooth-Host an den Bluetooth-Controller versendet werden kann, um diesen zu steuern. [BTSPEZ21 (2007)]*

Sowohl die HCI-Befehle als auch die HCI-Events werden nach ihrer Funktionalität in Gruppen eingeteilt. Die Tablelle 2.2 stellt diese Einteilung dar.

Generic Events	The generic events can occur due to multiple commands, or events that can occur at any time.
Device Setup	The device setup commands are used to place the Con- troller into a known state.
Controller Flow Control	The controller flow control commands and events are used to control data flow from the Host to the controller.
Controller Information	The controller information commands allow the Host to discover local information about the device.
Controller Configuration	The controller configuration commands and events allow the global configuration parameters to be configured.
Device Discovery	The device discovery commands and events allow a device to discover other devices in the surrounding area.
Connection Setup	The connection setup commands and events allow a device to make a connection to another device.
Remote Information	The remote information commands and events allow information about a remote device's configuration to be discovered.
Synchronous Connections	The synchronous connection commands and events allow synchronous connections to be created
Connection State	The connection state commands and events allow the configuration of a link, especially for low power operation.
Piconet Structure	The piconet structure commands and events allow the discovery and reconfiguration of piconet.
Quality of Service	The quality of service commands and events allow qual- ity of service parameters to be specified.
Physical Links	The physical link commands and events allow the con- figuration of a physical link.
Host Flow Control	The Host flow control commands and events allow flow control to be used towards the Host.
Link Information	The link information commands and events allow infor- mation about a link to be read.
Authentication and Encryption	The authentication and encryption commands and events allow authentication of a remote device and then encryption of the link.
Testing	The testing commands and events allow a device to be placed into test mode.

Tabelle 2.2: Übersicht der HCI-Events und HCI-Befehle nach Funktion.
Quelle: [BTSPEZ21 (2007)], Table 3.1: Overview of commands and events

2.6 L2CAP - Logical Link Control and Adaptation Protocol

Logical Link Control and Adaptation Layer Protocol (L2CAP) ermöglicht es, den über ihm liegenden Protokollen verbindungsorientierte und verbindungslose Dienste anzubieten. [BTSPEZ21 (2007)]

1. **Multiplexing**

   Das Baseband-Protokoll kann nicht unterscheiden, an welchen der Protokolle aus den oberen Protokoll-Schichten eine Nachricht gerichtet ist. Diese Aufgabe übernimmt das L2CAP-Protokoll. Das L2CAP-Protokoll ist für die Weiterleitung der Verbindungsanfrage bis zu dem gewünschten Protokoll in einer der oberen Protokoll-Schichten verantwortlich. Durch den Einsatz von „L2CAP-Channels“ wird es möglich, auch mehrere Verbindungen parallel zu unterhalten. [BTSPEZ21 (2007)]

   Definition 13 (*L2CAP-Channel*)

   > *„L2CAP provides a multiplexing role allowing many different applications to share the resources of an ACL-U logical link between two devices. Applications and service protocols interface with L2CAP using a channel-oriented interface to create connections to equivalent entities on other devices. L2CAP channel endpoints are identified to their clients by a Channel Identifier (CID). This is assigned by L2CAP, and each L2CAP channel endpoint on any device has a different CID.“*
   > *[BTSPEZ21 (2007)]*

2. **Segmentierung**

   L2CAP übernimmt die Segmentierung der Pakete. Dadurch können größere Pakete versendet werden, als es die Protokolle aus den darunter liegenden Schichten erlauben. [BTSPEZ21 (2007)]

3. **Flusskontrolle**

   L2CAP bietet eine Flusskontrolle für die einzelnen L2CAP-Channels. Die Flusskontrolle ist optional und kann daher abgeschaltet werden. [BTSPEZ21 (2007)]

4. **Fehlerbehandlung**

   L2CAP bietet eine eigene Fehlerbehandlung. Die Fehler in den PDUs werden erkannt, und es wird eine neue Übertragung des defekten Paketes angestoßen. Die Benutzung der L2CAP-Fehlerbehandlung ist optional. [BTSPEZ21 (2007)]

5. **Quality of Service**

   L2CAP bietet eine „Quality of Service“ Unterstützung. Beim Verbindungsaufbau erlaubt L2CAP einen Austausch von „Quality of Service“ Informationen. [BTSPEZ21 (2007)]

6. **Fragmentierung**

   Protokolle aus den Schichten über dem L2CAP-Protokoll könnten Pakete in Größen benötigen, welche sich von der durch die Segmentierung bestimmten Paketgröße abweichen. Durch die Fragmentierung ist die Anpassung der Paketgröße für die Protokolle aus den oberen Schichten möglich. [BTSPEZ21 (2007)]

Die Details zum L2CAP-Protokoll können der Bluetooth-Spezifikation [BTSPEZ21 (2007)] entnommen werden.

2.7 RFCOMM - Radio Frequency Communication

Das Radio Frequency Communication (RFCOMM)-Protokoll baut auf dem L2CAP-Protokoll auf. Die Aufgabe dieses Protokolls ist es, eine serielle (RS-232) Ports zu emulieren. Das RFCOMM-Protokoll basiert auf dem TS 07.10. Standard. Dieser Standard beschreibt die kabellose Verbindung über die serielle Schnittestelle. [Sikora (2008)]

Dieses Protokoll erlaubt eine einfache Portierung von Anwendungen, welche die weit verbreitete serielle Schnittstelle RS-232 nutzen, auf den Bluetooth-Standard. Dieses Protokoll erlaubt die Verwendung von mehreren parallel geöffneten Ports. Jedes Bluetooth-Gerät kann dadurch gleichzeitig bis zu 60 serielle Schnittstellen emulieren. [RFCOMM (2003)] [RFCOMM (2003)]

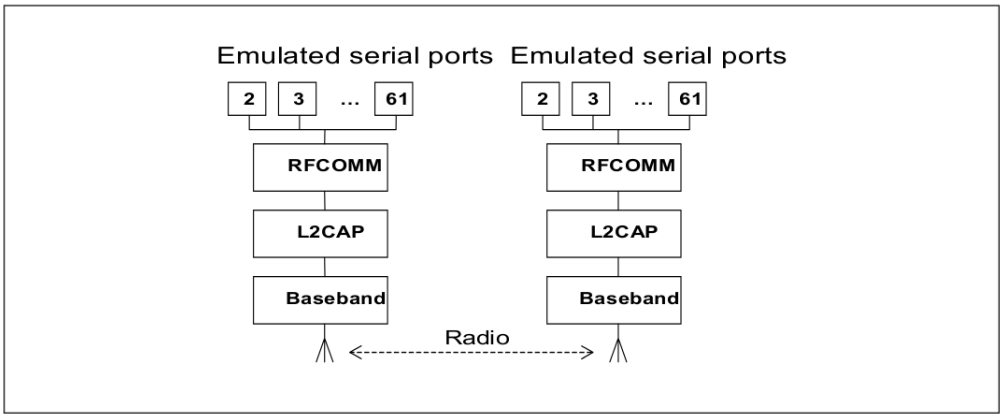

Abbildung 2.4: Diese Abbildung veranschaulicht die gleichzeitige Emulation von bis zu 60 seriellen Schnittstellen.

Quelle: [RFCOMM (2003)], Figure 2.2: Multiple Emulated Serial Ports.

RFCOMM unterscheidet nicht zwischen Data Terminal Equipment (DTE)- und Data Communication Equipment (DCE)-Geräten. (Ein DTE-Gerät kann zum Beispiel ein Rechner sein, und ein DCE-Gerät ein Modem). Die Art auf die der TS 07.10 Standard die RS-232 Kontrollsignale versendet, ist die gleiche wie der Versand der Kontrollsignale zweier Geräte über eine Null-Modem-Verbindung. [RFCOMM (2003)]

Weiterhin erlaubt der RFCOMM-Protokoll eine parallele Kommunikation mit mehreren Bluetooth-Geräten. Diese verlangt allerdings die Unterhaltung mehrerer „Multiplexer Sitzungen“ (nach dem TS 07.10 Standard). [RFCOMM (2003)]

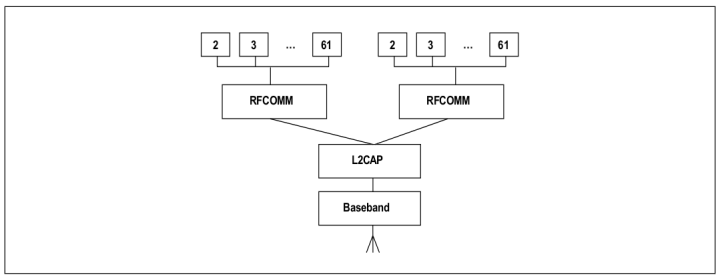

Abbildung 2.5: Parallele Kommunikation mit mehreren Bluetooth-Geräten.

Quelle: [RFCOMM (2003)], Figure 2.3: Emulating serial ports coming from two Bluetooth devices.

2.8 SDP - Service Discovery Protocol

Das Service Discovery Protocol (SDP) beitet den Anwendungen die Möglichkeit, nach Diensten in Reichweite zu suchen und die Eigenschaften der angebotenen Dienste zu erfragen. Die gefundenen Dienste können dann von der Anwendung in Anspruch genommen werden. Er erlaubt es natürlich auch, eigene Dienste anderen Bluetooth-Anwendungen in Reichweite zur Verfügung zu stellen.

Die Bluetooth-Verbindung wird erst nach der Wahl des gewünschten Dienstes aufgebaut. Es ist zu beachten, dass nach der Wahl eines Dienstes der Client stets eine separate Verbindung zum Dienst aufbauen muss. Das SDP-Protokoll ist bei der eigentlichen Benutzung des Dienstes nicht von Bedeutung. [Hildebrandt (2004)] [BTSPEZ21 (2007)]

Die angebotenen Dienste können sowohl als Software, Hardware oder auch eine Verbindung von Software und Hardware realisiert sein. Die Kommunikation läuft dabei zwischen einem Client, welcher einen Dienst in Anspruch nehmen möchte und einem Server, welcher Dienste anbietet, ab. Der Server verwaltet dabei eine Liste von sogenannten „Service-Records“. Jeder „Service-Record“ ist ein Eintrag, welcher Informationen zu einem einzelnen Dienst speichert. Um die Liste der „Service-Records“ vom Server zu erhalten, sendet der Client eine „SDP-Request-Nachricht“ an den Server. [BTSPEZ21 (2007)]

Bei Benutzung eines Bluetooth-Gerätes kann nur ein SDP-Server zur Zeit aktiv sein. Dabei ist zu beachten, dass ein einzelnes Bluetooth-Gerät parallel sowohl als SDP-Server, als auch als SDP-Client fungieren kann. Sollte der SDP-Server ausfallen, so bekommt der Client keine Benachrichtigung. [BTSPEZ21 (2007)]

Die in diesem Kapitel vorgestellten Protokolle sind sehr verbreitet und in den meisten Smartphones realisiert. Um eine Vielzahl von Smartphones zu unterstützen, bietet es sich an, auf diese Protokolle zurückzugreifen. In den folgenden Kapiteln werden Soft- und Hardwarekomponenten vorgestellt, welche diese Protokolle implementieren, beziehungsweise auf ihnen aufbauen.

3 Analyse der Anforderungen an das Gesamtsystem

3.1 Anwendungsszenarien

Um die Funktionalität des Gesamtsystems zu beschreiben und Anforderungen zu definieren, müssen zunächst Anwendugsszenarien definiert werden:

Definition 14 (*Zugangskontrollsystem*)

> *Das Zugangskontrollsystem ist für die Erkennung und Überwachung von authentifizierten Benutzern zuständig. Weiterhin ist es für die Aktivierung einer Alarmanlage verantwortlich, um den Zugang zu einem geschützten Objekt (zum Beispiel einer Garage) für nicht authentifizierte Benutzer zu erschweren. Das Zugangskontrollsystem besteht aus einer Menge von Soft- und Hardware-Komponenten.*

Definition 15 (*Benutzerverwaltungssystem*)

> *Das Benutzerverwaltungssystem (verkürzt auch Verwaltungssystem) ist für die Verwaltung und Speicherung von authentifizierten Benutzern verantwortlich. Das Benutzerverwaltungssystem besteht aus einer Menge von Soft- und Hardware-Komponenten.*

Definition 16 (*Gesamtsystem*)

> *Das Gesamtsystem besteht aus dem Zugangskontrollsystem und dem Benutzerverwaltungssystem. Das Gesamtsystem beschreibt die Menge aller in dieser Arbeit vorgestellten Soft- und Hardware-Komponenten.*

3.1.1 Ein Master wird festgelegt

Nach dem Start des Zugangskontrollsystems ist zunächst kein einziger User angemeldet. Auch der Master der Anwendung ist noch nicht festgelegt.

Definition 17 (*User*)

Ein User bzw. Benutzer ist ein Bluetooth-Gerät welches, im Zugangskontrollsystem gespeichert ist. Sobald das Benutzergerät sich im Bluetooth-Bereich des Zugangskontrollsystems befindet, wird es vom Zugangskontrollsystem als solches erkannt und angemeldet.

Definition 18 (*Master*)

Ein Master ist ein User und wird auch wie ein User vom Zugangskontrollsystem behandelt. Weiterhin übernimmt der Master die Verwaltung des Zugangskontrollsystems. Der Master legt fest, welche Geräte als User im Zugangskontrollsystem gespeichert sind. Ohne die Zustimmung des Masters, kann kein neuer User im Zugangskontrollsystem aufgenommen werden. Der Master kann bereits bestehende User vom Zugangskontrollsystem löschen. Um Verwaltungsaufgaben übernehmen zu können, muss der Master über eine Verwaltungssoftware verfügen und als Master am Zugangskontrollsystem angemeldet sein.

Nach dem erstmaligen Hochfahren des Gesamtsystems startet der Master die Suche nach vorhandenen Bluetooth-Geräten und baut eine sichere Verbindung zum Zugangskontrollsystem auf. Das erste Gerät, welches eine Verbindung zum Zugangskontrollsystem aufgebaut hat, wird als Master festgelegt. In einem Zugangskontrollsystem kann nur ein Master-Gerät gespeichert sein. Ein anderer Master kann nur nach einem Reset des Gesamtsystems bestimmt werden.

3.1.2 Ein User wird hinzugefügt

Sobald die Initialisierung erfolgreich verlaufen ist und ein Master festgelegt wurde, kann ein neuer User aufgenommen werden. Der Master muss sich über das Verwaltungssystem mit dem Zugangskontrollsystem verbinden. Danach muss der User eine Verbindung zum Zugangskontrollsystem aufbauen. Nun hat der Master über das Verwaltungssystem den neuen User zu akzeptieren oder abzulehnen. Sollte der Master den User nicht akzeptieren, wird er als solcher nicht im Zugangskontrollsystem abgespeichert. Sollte der Master seine Zustimmung erteilen, muss ein eindeutiger Name für diesen User festgelegt werden.

3.1.3 Die Erkennung und Anmeldung der User

Das Zugangskontrollsystem hat die Aufgabe, ständig nach den gespeicherten Benutzern (den Usern) im Bluetooth-Bereich zu suchen. Sobald ein User als solcher erkannt wurde, wird er am System angemeldet.

3.1.4 Überwachung der angemeldeten User

Das Zugangskontrollsystem hat die Aufgabe, die angemeldeten User (soweit vorhanden) zu überwachen. Sollte sich ein User aus dem Bluetoth-Bereich entfernen, muss das Zugangskontrollsystem dies so schnell wie möglich erkennen. Sollte das passieren, wird sofort die Suche nach anderen Usern gestartet.

3.1.5 Aktivierung der Alarmvorrichtung

Sollte kein einziger User oder Master angemeldet sein, so muss das Zugangskontrollsystem eine Alarmvorrichtung aktivieren. Dabei schaltet diese einen Bewegungssensor ein. Wenn dieser eine Bewegung erkannt hat, wird sofort ein Alarm ausgelöst.

3.1.6 Deaktivierung der Alarmvorrichtung

Sobald ein Nutzer angemeldet ist, wird sofort die Alarmvorrichtung deaktiviert.

3.1.7 User aus dem Zugangskontrollsystem entfernen

Um einen User zu entfernen, muss sich der Master über das Verwaltungssystem mit dem Zugangskontrollsystem verbinden. Danach hat der die Möglichkeit, den Benutzer nach dem festgelegtem Namen zu finden und auch zu löschen.

3.1.8 Die Liste der gespeicherten User auslesen

Um die Liste der gespeicherten Nutzer auslesen, muss sich der Master über das Verwaltungssystem sich mit dem Zugangskontrollsystem verbinden. Danach hat der die Möglichkeit, die Liste der gespeicherten User abzurufen.

3.2 Anforderungen an das Zugangskontrollsystem

3.2.1 Der Bluetooth Server

Definition 19 (*Bluetooth-Server*)

Der Bluetooth-Server ist sowohl eine Komponente des Zugangskontrollsystems, als auch eine Komponente des Benutzerverwaltungssystems. Das ganze Zugangskontrollsystem und ein Teil des Benutzerverwaltungssystems laufen auf einem Bluetooth-Server. Der Bluetooth-Server kontrolliert, ob authentifizierte Nutzer in der Nähe sind, und aktiviert falls notwendig die

Alarmanlage. Alle Software-Komponenten des Gesamtsystems mit Ausnahme der Benutzerverwaltungssoftware werden auf dem Bluetooth-Server ausgeführt.

Definition 20 (*Benutzerverwaltungssoftware*)

Die Benutzerverwaltungssoftware ist eine der Komponenten des Benutzerverwaltungssystems. Diese Software läuft jedoch nicht auf dem Bluetooth-Server wie alle anderen Software-Komponenten des Gesamtsystems, sondern auf dem Smartphone des Masters. Dieses Software erlaubt eine grafische Steuerung der Benutzerverwaltung durch den Master.

Das Zugangskontrollsystem muss den Benutzern als ein Bluetooth Server zur Verfügung stehen. Jedes Bluetooth Gerät sollte bei einem Suchlauf den Bluetooth-Server finden können, und somit die Möglichkeit haben, sich mit diesem zu verbinden.

3.2.2 Festlegung des Masters

Der Master muss im System abgespeichert sein und sicher bei einer Reauthentifizierung erkannt werden. Kein Dritter soll die Möglichkeit erhalten, den Master aus dem Zugangskontrollsystem zu löschen oder gar zu verändern. Ein neuer Master kann nur nach einem Reset der Applikation festgelegt werden.

3.2.3 Sichere Authentifizierung der User

Die Benutzerauthentifizierung muss garantieren, dass nur die vom Master akzeptierten User als solche erkannt werden. So darf der Nutzer zum Beispiel nicht an der MAC-Adresse seines Bluetooth-Gerätes erkannt werden, da diese gefälscht werden kann. Die Authentifizierung verlangt an dieser Stelle, den Einsatz sicherer Verfahren.

3.2.4 Speicherung der User

Die akzeptierten User müssen sicher gespeichert werden und dürfen nicht ohne Zustimmung des Masters gelöscht oder ihre Daten verändert werden. Eine Festlegung der maximalen Nutzeranzahl ist notwendig, um eventuelle Softwarefehler zu vermeiden.

3.2.5 Überwachung angemeldeter User

Die angemeldeten Benutzer müssen überwacht werden. Falls der Nutzer den Bluetooth-Bereich verlassen hat, oder er aus einem anderen Grund nicht mehr erreichbar ist, muss immer gewährleistet werden, dass das Zugangskontrollsystem dies bemerkt.

3.2.6 Suche nach Usern

Die Software muss gewährleisten, dass stets eine Suche nach Nutzern im Bluetooth-Bereich erfolgt, falls kein User angemeldet ist. Die ständige Suche darf auf keinen Fall unterbrochen werden. Auch die Wiederaufnahme der Suche nach dem ein User die Verbindung unterbrochen hat, muss immer gewährleistet werden.

3.2.7 Steuerung der Alarmvorrichtung

Die Verbindung zwischen dem Zugangskontrollsystem und der Alarmvorrichtung darf dem Angreifer nicht zugänglich sein. Auch die Stromversorgung muss vor unbefugtem Zugriff geschützt sein.

3.2.8 Kommunikation mit dem Verwaltungssystem

Das Verwaltungssystem muss auf dem Smartphone des Masters laufen. Sobald der Master eine Verbindung zum Bluetooth-Server aufgebaut hat, müssen Daten zur Verwaltung ausgetauscht werden.
Für diesen Austausch muss ein Protokoll festgelegt werden. Die Verbindung zwischen dem Zugangskontrollsystem und dem Verwaltungssystem muss verschlüsselt sein.

3.3 Anforderungen an das Verwaltungssystem

Das Verwaltungssystem muss auf dem Smartphone des Masters laufen. Je nach verwendetem Betriebssystem muss eine entsprechende App realisiert werden.

3.3.1 GUI

Die Verwaltungssoftware muss dem Master eine grafische Oberfläche zur Verwaltung der Benutzer bieten.

3.3.2 Kommunikation mit Zugangskontrollsystem

Der Austausch von Nachrichten zwischen dem Verwaltungssystem und dem Zugangskontrollsystem muss auf Basis vom Bluetooth-Protokoll ablaufen. Die Kommunikation muss verschlüsselt erfolgen.

3.3.3 Anfrage zur Authentifizierung

Das Verwaltungssystem muss den Master informieren, sobald ein potenzieller User erstmalig eine Authentifizierungsanfrage an das Zugangskontrollsystem sendet. Der Master muss die Möglichkeit erhalten, diese Anfrage per Eingabe zu bestätigen oder abzulehnen. Dies ist notwendig um das „Numeric-Comparison-Verfahren“ auszuführen. Details dazu folgen im Kapitel 3.4.6 „Assoziationsmodelle“.

3.3.4 Anzeige der Benutzer-Liste

Zu jedem Zeitpunkt muss es möglich sein, die aktuelle Benutzerliste abzurufen.

3.3.5 Löschen der Benutzer

Das Löschen eines Users aus dem Zugangskontrollsystem muss unterstützt werden.

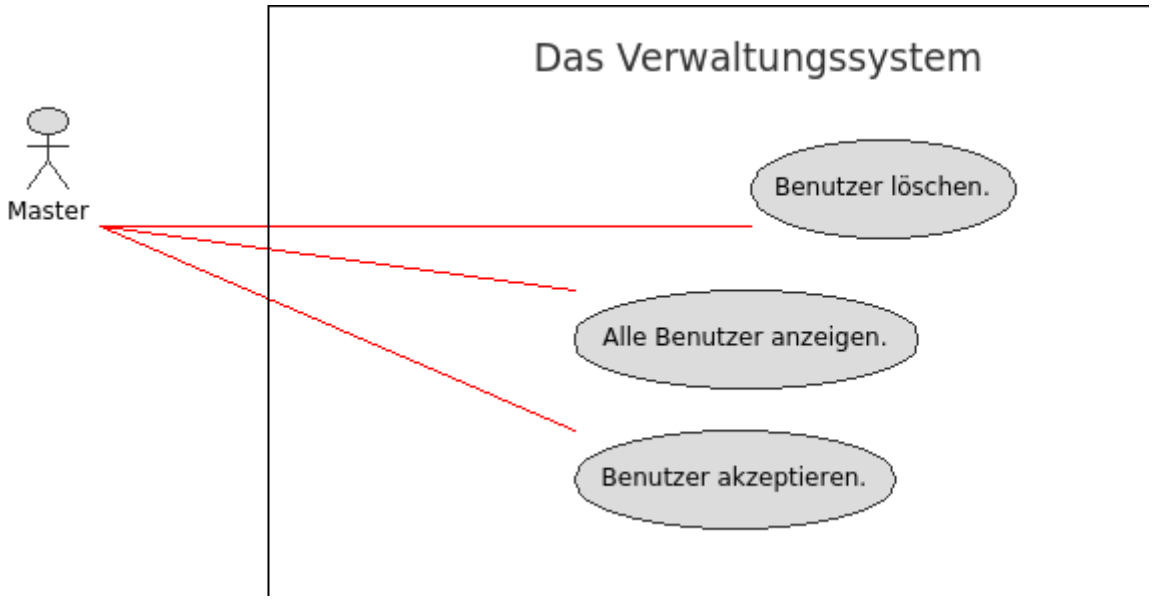

Abbildung 3.1: Beabsichtigte Master Operationen bezogen auf das Verwaltungssystem.

3.4 Bluetooth Sicherheit

3.4.1 Abhör- und Eindringsicherheit Allgemein

Grundsätzlich bietet eine Bluetooth Verbindung an sich keine Abhör- bzw. Eindringsicherheit. Bis zur Version 2.1 konnte die Sicherheit nur durch eine Schlüsselvergabe erreicht werden. Erst nach dem Austausch eines Schlüssels (einer Personal Identification Number (PIN)) kann dabei eine sichere Verbindung aufgebaut werden. Maßgebend dabei ist die Länge des gewählten Schlüssels. Wird der PIN-Code zu kurz gewählt, so gilt die Verbindung nicht mehr als sicher. Ab der Bluetooth Spezifikation 2.1 steht das Secure Simple Pairing (SSP) zur Verfügung. Mit dieser Erweiterung kann ein hohes Maß an Sicherheit, auch ohne eine Eingabe einer PIN erreicht werden.

3.4.2 Das Pairing Allgemein

Definition 21 (*Link Key*)

> *Ein Link Key ist ein Schlüssel, welcher als Basis für die Unterhaltung einer sicheren Verbindung dienen kann. [BTSPEZ21 (2007)]*

Das Ziel eines Pairing-Prozesses ist die Generierung eines Link Keys. Dieser Schlüssel wird zur Authentifizierung verwendet. Weiterhin wird vom Link Key und mehreren Zufallszahlen ein Encryption Key abgeleitet, welcher wiederum für die Verschlüsselung der Verbindung eingesetzt wird. Die Bluetooth Spezifikation definiert heute zwei Verfahren:

1. LMP-Pairing.
2. Secure Simple Pairing

Die Bluetooth-Spezifikation erlaubt es auch, eigene Pairing-Verfahren zu realisieren. Sobald am Ende eines Pairing-Prozesses beide Partner einen Link Key erhalten haben, wird dieser gespeichert, um beim nächsten Verbindungsversuch eine Reauthentifizierung zu ermöglichen. Es muss dabei lediglich überprüft werden, ob beide Parteien den gleichen Link Key besitzen.

3.4.3 Der LMP-Pairing-Prozess

Definition 22 (*LMP-Pairing*)

> *LMP-Pairing bezeichnet ein Verfahren zur Authentifizierung zweier Geräte. Dieses Verfahren basiert auf der Eingabe einer Personal Identification Number (PIN). Mit Hilfe dieser PIN wird ein Link Key generiert. [BTSPEZ21 (2007)]*

Die israelischen Forscher Yaniv Shaked und Avishai Wool beschrieben im Jahr 2005 ein Verfahren, mit dem eine bereits bestehende, vermeintlich abhörsichere Verbindung unterbrochen, und beim erneutem Verbindungsversuch in die Kommunikation eingerbochen werden kann. [Shaked und Wool (2005a)]

Da die PIN nur während des Pairingprozesses geknackt werden kann, wird genau dieser im Folgenden näher erläutert. Der Bluetooth Pairing- und Authentifizierungsprozess besteht nach Shaked und Wool aus 3 Schritten.

1. Erstellung eines Initialization Key (K_{init}).
2. Erstellung eines Link Key (K_{ab}).
3. Authentifizierung.

Nach der Ausführung dieser 3 Schritte können die Partner einen Encryption-Key berechnen und zur Verschlüsselung nutzen. Auf diese Weise wird die darauf folgende Kommunikation abhörsicher durchgeführt. Bevor das Pairing beginnen kann, müssen beide Partner getrennt voneinander eine PIN eingeben. Dieser Schlüssel wird benötigt, um den K_{init} zu berechnen.

Im ersten Schritt wird das K_{init} durch den Algorithmus E_{22} berechnet. Alle hier aufgeführten Algorithmen (E22, E21,und E1) beruhen auf SAFER+, einer Familie von Verschlüsselungsverfahren. Der K_{init} wird lediglich zur Berechnung des K_{ab} benötigt und wird nach dessen Generierung sofort verworfen. Zur seiner Berechnung sind drei Eingaben erforderlich:

1. Die Bluetooth Address (BDADDR).
2. Die eingegebene PIN.
3. 128 Bit lange Zufallszahl IN_RAND.

Der Berechnete Schlüssel K_{init} ist 128 Bit lang.

Sowohl die PIN als auch die Zufallszahl IN_RAND werden an die Gegenstelle übertragen. Welche BDADDR oder auch welche PIN zur Berechnung benutzt wird, wird von weiteren Faktoren bestimmt. Die Details können der Bluetooth Spezifikation [BTSPEZ21 (2007)] entnommen werden.

Im Zweiten Schritt wird K_{ab} generiert. Partner A und B berechnen jeweils eine neue 128 Bit lange Zufallszahl. Im Weiteren werden diese Zufallszahlen LK_RAND_a und LK_RAND_b bezeichnet. Die ($LK\_RAND_a \oplus K_{init}$) und ($LK\_RAND_b \oplus K_{init}$) werden zwischen den Parteien ausgetauscht. Nach dem Austausch der beiden Zufallszahlen, wird nun mit Hilfe des Algorithmus E_{21} der Schlüssel K_{ab} berechnet. Der Algorithmus E_{21} hat dabei folgende Eingabeparameter:

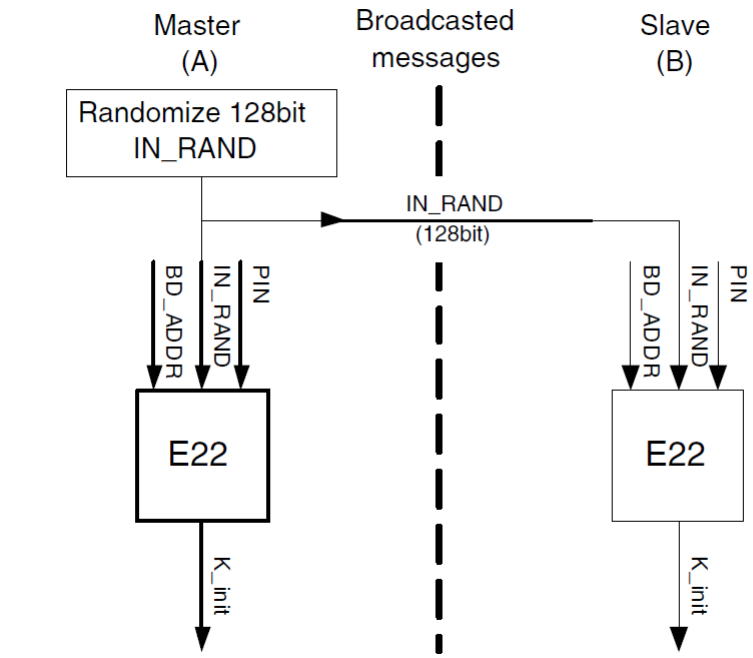

Abbildung 3.2: Berechnung des K_{init} Schlüssels durch den E_{22} Algorithmus.

Quelle: [Shaked und Wool (2005a)]

1. BDADDR.
2. LK_RAND.

$$K_{\text{ab}} = (E_{21}(BDADDR_{\text{a}} \oplus LK\_RAND_{\text{a}})) \oplus (E_{21}(BDADDR_{\text{b}} \oplus LK\_RAND_{\text{b}}))$$

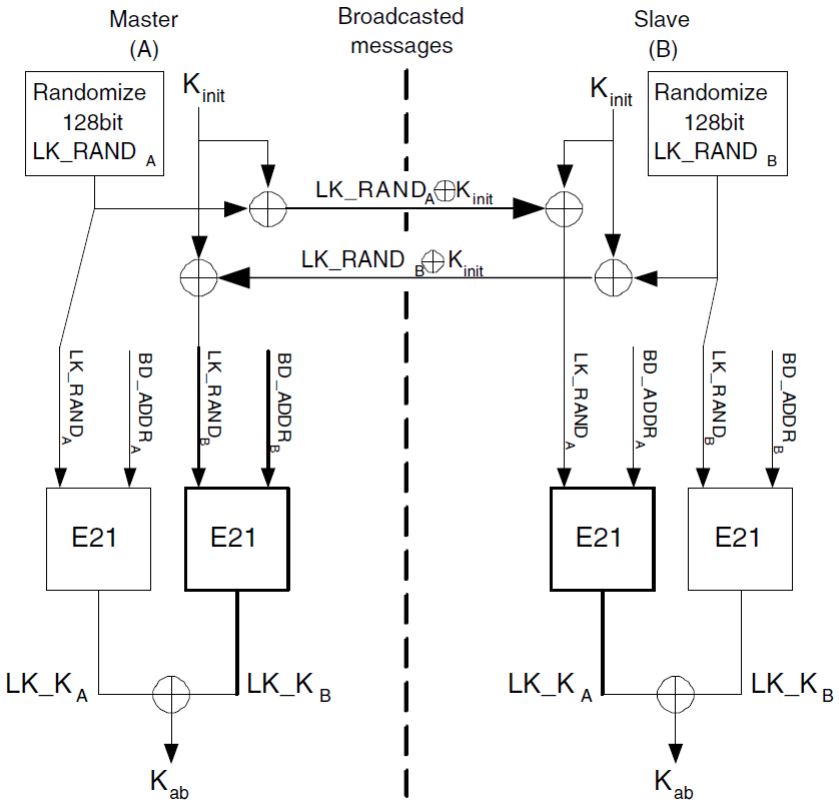

Abbildung 3.3: Berechnung des K_{ab} Schlüssels durch den E_{21} Algorithmus.

Quelle: [Shaked und Wool (2005a)]

Nach dem K_{ab} berechnet wurde, wird im drittem Schritt die Authentifizierung ausgeführt. Die Grundlage für die Authentifizierung bildet das Challenge-Response-Verfahren. Der Ablauf wird dabei ein drei Schritte eingeteilt:

1. Der Partner A wählt eine 128 Bit große Zufallszahl AU_RAND_a und versendet diese an Partner B.

2. Beide Parteien berechnen nun die 32 Bit große Zahl SRES mit Zuhilfenahme des Algorithmus E_1. Die Parameter der Funktion E_1 sind:
   a) Zufallszahl AU_RAND_a
   b) K_{ab}
   c) $BDADDR_b$
3. Im dritten Schritt werden die beiden Wörter verglichen. Die Authentifizierung war erfolgreich, falls die Werte SRES übereinstimmen.

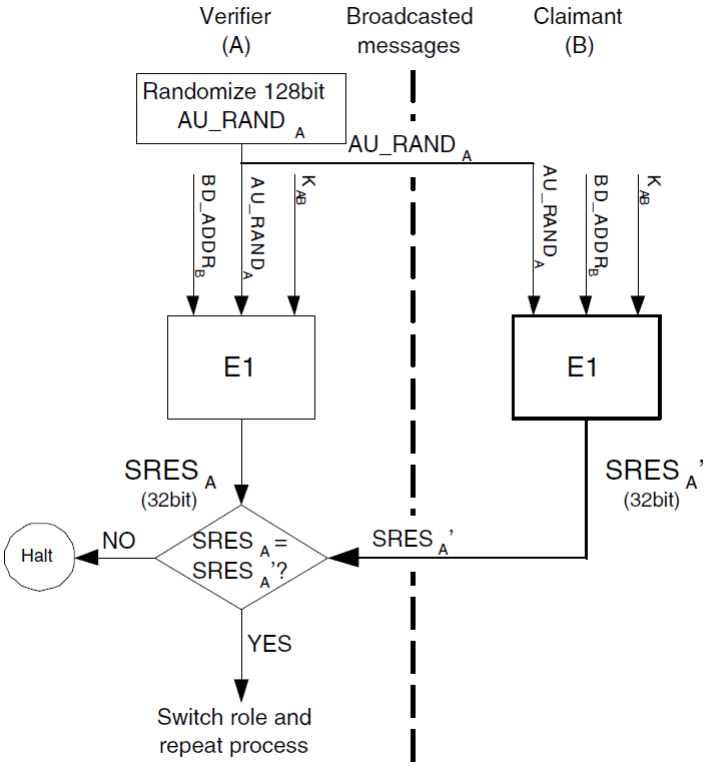

Abbildung 3.4: Ablauf der Authentifizierung. Die Authentifizierung war erfolgreich, falls die Werte SRES übereinstimmen.

Quelle: [Shaked und Wool (2005a)]

3.4.4 Die Sicherheitslücke

Um einen möglichen Angriff zu beschreiben, zeigen Yaniv Shaked und Avishai Wool die übertragenen Daten in einer Tabelle auf.

	Quelle	Ziel	Daten	Datengöße in Bit
1	A	B	IN_RAND	128
2	A	B	LK_RAND_a	128
3	B	A	LK_RAND_b	128
4	A	B	AU_RAND_a	128
5	B	A	$SRES_a$	32
6	B	A	AU_RAND_b	128
7	A	B	$SRES_b$	32

Tabelle 3.1: Nachrichten die von einem Angreifer zu Beginn des Pairing-Prozesses mitgehört werden können.

Unter der Annahme, dass der Angreifer diese Nachrichten zu Beginn der Übertragung aufgezeichnet hat, hat er nun die Möglichkeit die PIN zu bestimmen. Dazu muss der Angreifende einen Brute-Force-Algorithmus anwenden. Da die IN_RAND und die BDADDR bekannt sind, wird durch das Anwenden der E_{22} Funktion ein potenziell richtiges K_{init} bestimmt. Mit diesem Schlüssel kann nun versucht werden, die Nachrichten 2 und 3 zu entschlüsseln. Nachdem $LKRAND_a$ und LK_RAND_b bestimmt wurden, kann nun der potenzielle K_{ab} berechnet werden. Der errechnete Schlüssel K_{ab} und die Zahl AU_RAND_a aus der Nachricht 4 können nun zur Berechnung des SRES Wortes eingesetzt werden. Nun muss das Ergebnis mit dem SRES nur noch verglichen werden. Bei Notwendigkeit kann dass errechnete SRES mit dem Wort aus der Nachricht 7 nochmals verglichen werden, um sicherzustellen, dass die PIN auch wirklich die richtige ist.

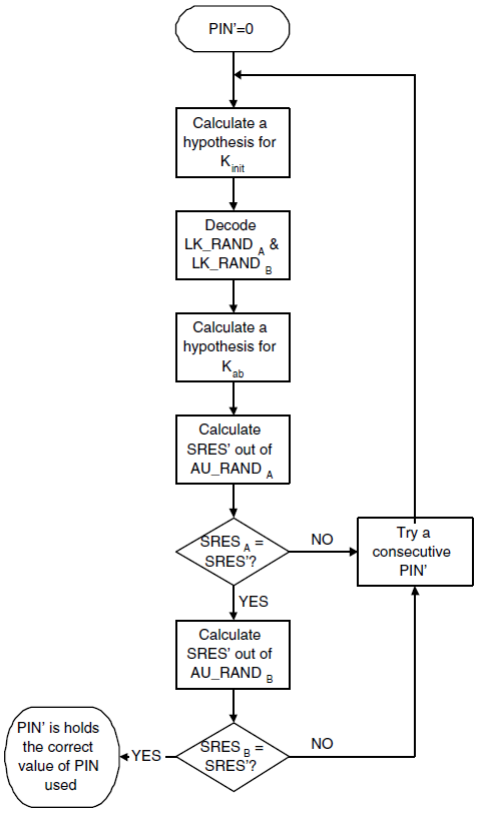

Abbildung 3.5: Ablauf des von Shaked und Wool beschriebenen Angriffes zur Bestimmung der PIN durch die Anwendung eines Brute-Force-Algorithmus.

Quelle: [Shaked und Wool (2005a)]

Da die PIN nur während des Pairing-Prozesses geknackt werden kann, ist die Vermutung nahe, dass Geräte, welche die Link Keys bereits ausgetauscht haben angriffssicher sind. Dies ist jedoch leider nicht der Fall. Es gibt mehrere Möglichkeiten die Beiden Parteien zum erneuten Starten des Pairing-Prozesses zu bewegen. Die Bluetooth Spezifikation erlaubt es den Teilnehmern, den Link Key zu verwerfen. In diesem Fall wird wird sofort eine Reauthentifizierung gestartet. Es gibt mehrere Möglichkeiten, das Pairing zwischen Teilnehmern unbefugt erneut anzustoßen:

1. In der Authentifizierungsphase, nach dem der Master AU_RAND versendet hat, kann der Angreifer eine „LMP_not_accepted“-Nachricht an den Master zurücksenden, welche den Master dazu zwingt, das Pairing erneut zu starten.

2. In der Authentifizierungsphase, nach dem der Master AU_RAND versendet hat, kann der Angreifer eine falsche SRES an den Master zurücksenden, welche den Master dazu zwingt das Pairing erneut zu starten.

3. In der Authentifizierungsphase, versendet der Angreifer als erster die IN_RAND Nachricht an den Slave, und lässt ihn somit denken, dass der Master den Link Key vergessen hat.

Durch mehrere Optimierungen haben Shaked und Wool es geschafft, einen Algorithmus zu entwickeln, welcher auf einem Pentium IV (3Ghz) in 63 Millisekunden eine 4-Stellige PIN knacken kann. Unter den selben Bedingungen dauerte es rund 76,127 Sekunden für das Herausfinden einer 7-Stelligen PIN. Natürlich kann dieses Verfahren durch den Einsatz von zum Beispiel Field Programmable Gate Array (FPGA)-Boards weiter beschleunigt werden. Ein solcher Angriff kann jedoch nicht mit üblichen Bluetooth-Geräten ausgeführt werden. Um das gesamte Bluetooth-Frequenzband zu belauschen, braucht der Angreifer eine spezielle Hardware. Eine solche Hardware kostet jedoch mehrere tausend Euro.

Dabei muss man beachten, dass viele Hersteller von Bluetooth-Geräten lediglich eine 4-Stellige PIN unterstützen, wobei die Bluetooth Spezifikation 128 Bit anbietet. Oft wird auch zum Beispiel bei Head-Sets standardmäßig eine feste PIN wie z.B. „0000“ vergeben. Diese kann leicht erraten und so das Gespräch mitgehört werden. Um die die Sicherheit zu Steigern bietet es sich an, eine längere PIN zu wählen. Leider gibt es keine Erkenntnisse über den Zusammenhang zwischen der Dauer der Berechnung der PIN und der gewählten PIN Länge mit der heute zur Verfügung stehenden Hardware. Die Ermittlung dieser Werte würde den Rahmen dieser Arbeit sprengen.

Da in dieser Arbeit die User sich mit einem Smartphone am Zugangskontrollsystem anmelden und die meisten Smartphones meist nur eine Eingabe von 4-Stelligen PIN aus Ziffern erlauben, ist eine ausreichende Sicherheit somit nicht gegeben. Eine andere Möglichkeit, eine sichere Verbindung zu betreiben ist es, den Link Key in den jeweiligen Authentifzierungslisten dauerhaft zu speichern und die Reauthentifizierung in der Software zu deaktivieren. Nur so kann sichergestellt werden, dass kein Angriff möglich ist. Dies ist jedoch nicht möglich, da bei der Initialisierung des Systems die Nutzer noch nicht bekannt sind.

3.4.5 Secure Simple Pairing

Das Secure Simple Pairing (SSP) ist eine Erweiterung, welche seit der Version 2.1 der Bluetooth Spezifikation zur Verfügung steht. SSP wurde entwickelt, um die Handhabung des Pairing-Prozesses zu vereinfachen und dabei gleichzeitig die Sicherheit zu steigern. Einfachheit der Benutzung und ein gleichzeitig hoher Sicherheitsstandard werden in dieser Technologie vereinigt. Die Sicherheit wird vor allem dadurch gesteigert, dass nun eine 16 Stellige PIN eingesetzt wird, welche aus Buchstaben und Zahlen besteht.

Bluetooth-Geräte, welche Bluetooth Spezifikationen bis zur Version 2.0 unterstützen, setzten zum Pairing eine 4-Stellige PIN aus Zahlen ein. Somit stellen sie ein Risiko für ihre Benutzer dar. Das Elliptic Curve Diffie Hellman-Verfahren wird hier eingesetzt, um möglichen Angriffen entgegenzuwirken. Dadurch ist die Eingabe einer Personal Identification Number (PIN) nicht mehr erforderlich. (Unter Angriffen sind an dieser Stelle Man In The Middle (MITM)- und Lauschangriffe gemeint, welche das SSP unterbinden soll).

Beim SSP hat jeder Partner einen Diffie-Hellman- Public Key und einen Diffie-Hellman- Private Key. Ein Public Key ist ein Schlüssel, welcher für jeden im Bluetooth-Bereich sichtbar ist. Ein Private Key ist wiederum ein Schlüssel, welcher streng geheim ist und niemals nach außen sichtbar wird. Beide Werte sind 192-Bit groß. Um den Link Key nun durch einen Angriff in Erfahrung zu bringen, müsste der Angreifer das Diffie-Hellman-Problem lösen. Von diesem wird aber angenommen, dass es praktisch nicht lösbar ist. Der Public Key kann dabei leicht auf der Grundlage des Private Key berechnet werden. Angenommen zwei Partner A und B führen das Pairing durch. Dann gibt es eine wohldefinierte Funktion für die gilt:

$$\boxed{F(PublicKey_{\mathrm{b}}, PrivateKey_{\mathrm{a}}) = F(PublicKey_{\mathrm{a}}, PrivateKey_{\mathrm{b}}))}$$

Das Ergebnis der Funktion F ist der Diffie-Hellman Key (DH-Key). Dieser Schlüssel ist 192 Bit groß und bildet die Grundlage für die Berechnung des Link Key. Der Angreifer kann aus den abgefangenen Nachrichten nur mit enormen Rechenaufwand den Link Key bestimmen. Dazu müssten diskrete Logarithmen berechnet werden. Diese Berechnung würde mit dem heutigen Stand der Technik einen so großen Zeitaufwand erfordern, dass die Angriff damit nicht lohnenswert wäre. [ELYSISBT (2011)]

Der entscheidender Vorteil des SSP im Bezug auf Sicherheit liegt darin, dass bei diesem Verfahren, im Gegensatz zum LMP-Pairing, keine sensiblen Daten übertragen werden. Beim LMP-Pairing war die Abhörsicherheit bei Geräten wie zum Beispiel Headsset erheblich eingeschränkt. Da keine Hardware zur Eingabe einer PIN vorhanden war, wurde diese meist fest vergeben.

3.4.6 Assoziationsmodelle

Um auch bei Geräten mit eingeschränkter Ein- oder Ausgabefähigkeit ausreichende Sicherheit zu gewährleisten, bietet das SSP vier Assoziationsmodelle an:

1. **Numeric Comparison**

   Dieses Modell sollte eingesetzt werden, falls beide Parteien sowohl über Eingabe- als auch über Ausgabeschnittstellen verfügen. Beim Pairing zwischen zwei Smartphones zum Beispiel, verfügen beide Parteien über einen Bildschirm und eine Tastatur. Beim Pairing-Prozess wird auf beiden Bildschirmen zunächst eine 6-Stellige Nummer angezeigt. Um einen MITM-Angriff zu vermeiden, sollten die Nummern verglichen werden. Bei Übereinstimmung müssen beide Teilnehmer mit der Eingabe „YES“ den Prozess abschließen. Die 6-Stellige Nummer dient hier nur dazu sicherzustellen, dass die Verbindung mit dem gewünschten Partner aufgebaut wird. Auch wenn der Angreifer an diese gelangt, bleibt die Verbindung abhörsicher.

2. **Just Works**

   Just Works wurde für Szenarien entwickelt, bei denen einer der Partner keine Eingabe- bzw. Ausgabeschnittstellen zur Verfügung hat. Hier wird ebenfalls das Numeric Comparison Protocol verwendet, jedoch wird die 6-Stellige Nummer nicht angezeigt. Dieses Model bietet damit keine Sicherheit gegen MITM-Angriffe.

3. **Out-Of-Band**

   Bei diesem Verfahren wird ein „Out-Of-Band“-Kanal (z.B. Near Field Communication (NFC)) benutzt, um das Pairing zu ermöglichen. Dabei muss sichergestellt werden, dass dieser Kanal die notwendigen Anforderungen erfüllt, um zum Beispiel MITM-Angriffe zu unterbinden. Die auf diesem Weg übertragenen Daten müssen Informationen zur Lokalisierung und Authentifizierung beinhalten.

4. **Passkey Entry**

   Dieser Modus sollte in dem Fall eingesetzt werde, wenn der Partner A nur über eine Eingabeschnittstelle verfügt und der Partner B nur über eine Ausgabeschnittstelle. Dem Partner B wird dabei eine 6-Stellige Nummer angezeigt, welche der Partner A eingeben muss, um den Pairing-Prozess durchzuführen. Dabei dient diese Nummer ebenfalls nur der Vermeidung vom MITM-Angriffen. Sie kann nicht eingesetzt werde, um die übertragenen Daten zu entschlüsseln.

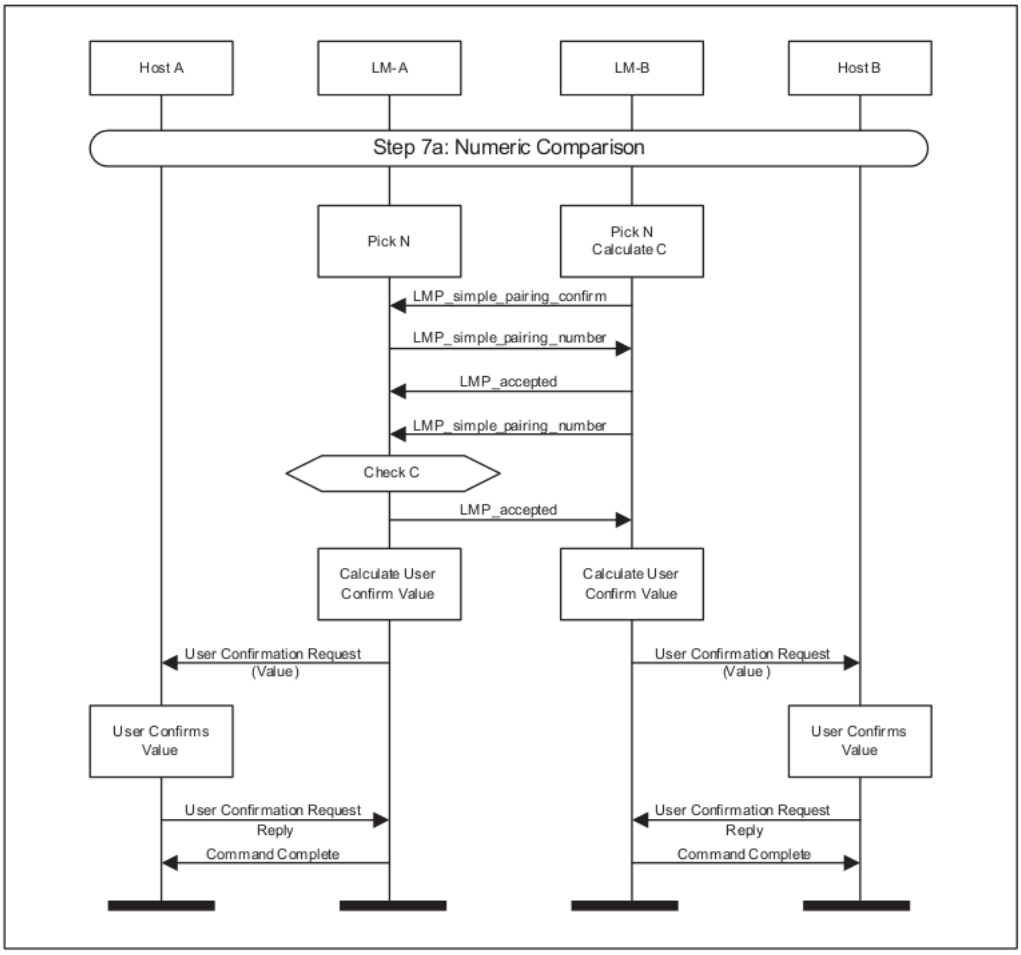

Abbildung 3.6: Ablauf der Authentifizierung mit Hilfe des Numeric-Comparison-Verfahrens.
Quelle: [BTSPEZ21 (2007)], Figure 4.10: Numeric Comparison Authentication

3.4.7 Das Man-In-The-Middle Problem beim SSP

Der Diffie-Hellman-Schlüsselaustausch allein, welcher beim SSP-Verfahren verwendet wird, bietet keine Sicherheit gegen Man In The Middle (MITM)-Angriffe. Um diesen Angriff auszuführen, muss der Angreifer die ausgetauschten Nachrichten abfangen. Danach sendet er an die beiden Partner seinen eigenen $\text{PublicKey}_\text{m}$. Beide Partner glauben nun, miteinander zu kommunizieren, wobei alle Pakete an sie vom Angreifer weitergeleitet werden. So kann nun ein unbefugter die gesamte Kommunikation abhören.

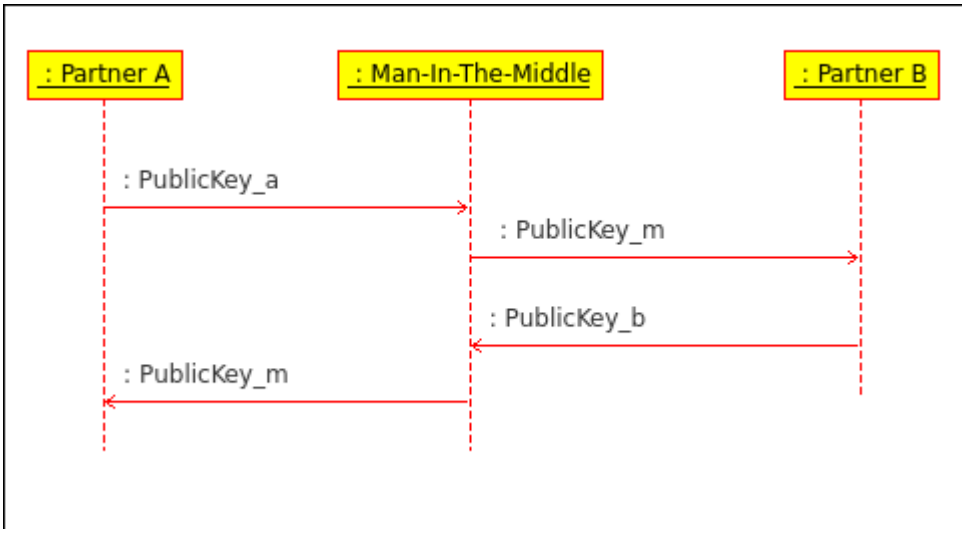

Abbildung 3.7: Dieses Diagramm veranschaulicht einen möglichen MITM-Angriff.

Der Diffie-Hellman-Schlüsselaustausch wurde hier im Prinzip zwei mal ausgeführt. Einmal zwischen dem Angreifer und dem Partner_a, und ein zweites Mal zwischen dem Angreifer und dem Partner_b.

Um diesem Angriff entgegenzuwirken, bietet das Secure Simple Pairing (SSP) zwei Assoziationsmodelle zur Auswahl. Durch die Benutzung des Numeric-Comparison-Verfahrens oder des Passkey-Entry-Verfahrens kann man sichergehen, dass die Verbindung zum richtigen Partner aufgebaut wird. Das SSP-Verfahren arbeitet mit einer aus Zahlen bestehender 6-stelligen Nummer. Diese wird auch als Comparison-Number bezeichnet. Somit steht die Chance für den Angreifer die Comparison-Number zu erraten und einen MITM-Angriff zu starten 1 zu 1.000.000. Die Länge der Comparison-Number ist auf 6 Stellen begrenzt, um einen Ausgleich zwischen der Sicherheit und der Benutzbarkeit zu schaffen. Selbstverständlich würde eine Nummer mit mehr Stellen die Sicherheit gegen MITM-Angriffe weiter erhöhen.

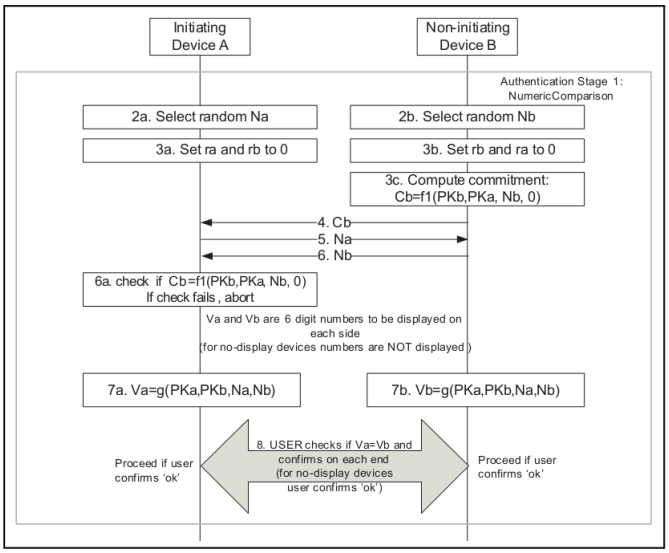

Abbildung 3.8: ,
Figure 4.10: Numeric Comparison Authentication, Figure 7.4: Authentication Stage 1: Numeric Comparison Protocol Details
Abgerufen am: 25.12.2012]Der Numeric-Comparison Protokoll im Detail.

Quelle: [BTSPEZ21 (2007)], Figure 4.10: Numeric Comparison Authentication, Figure 7.4: Authentication Stage 1: Numeric Comparison Protocol Details

Nach dem ersten Schritt der Authentifizierung, in welchem die Public Keys ausgetauscht wurden, wird im zweitem Schritt der Austausch nach dem Numeric-Comparison Protokoll ausgeführt. In den Schritten 2_a und 2_b berechnet jeder Partner eine 128 Bit große Zufallszahl N. Wie man im Schritt 3 sieht, wird eine Variable C mit Hilfe der Funktion f1 aus den beiden Public Keys der Kommunikationspartner und der Zufallszahl N auf beiden Seiten berechnet. Dieser Wert wird berechnet, um dem Angreifer später nicht die Möglichkeit zu geben, die ausgetauschten Werte zu modifizieren. Der Trick bei der Vorbeugung der MITM-Angriffe ist die Abfolge der Nachrichten 4, 5 und 6. Um sich zischen den beiden Parteien in die Kommunikation einzuschmuggeln, muss der Angreifer zunächst einen Austausch der o.g. Werte mit einem der Partner durchführen.
Es gibt dabei nur zwei Möglichkeiten:

1. **Fall 1:** Der Angreifer tauscht seine Zufallszahl N zuerst mit dem Partner B aus. Sollte der Man-In-The-Middle den Austausch zuerst mit dem Partner B durchführen, so müsste er im Schritt 5 die Zufallszahl Na an den Partner B senden. Da aber Na zu diesem Zeitpunkt noch nicht bekannt ist, kann diese auch nicht versendet werden.

2. **Fall 2:** Der Angreifer tauscht seine Zufallszahl N zuerst mit dem Partner A aus. Sollte der Angreifer den Austausch zuerst mit dem Partner A durchführen, müsste er im Schritt 4 die Zufallszahl Cb an den Partner B senden, diese kann aber zu diesem Zeitpunkt noch nicht berechnet werden, da Nb noch nicht bekannt ist. [BTSPEZ21 (2007)]

Der Angreifer kann nur seine eigenen Zufallszahlen N_{mitm} einschleusen, dies würde jedoch dazu führen, dass die Variablen Va und Vb im Schritt 7 nicht den gleichen Wert erhalten würden. In diesem Fall würden die Benutzer den Angriff sofort bemerken und das Pairing abbrechen.

3.4.8 Der Bluetooth NINO - MITM - Angriff

In der Arbeit „Secure Public Key Exchange Against Man-in-the-Middle. Attacks During Secure Simple Pairing (SSP) in Bluetooth.“ [Iman ALMomani und AL-Akhras (2011)] beschreiben die Autoren ein Verfahren, mit dem es möglich ist, einen MITM-Angriff bei einem Verbindungsaufbau durchzuführen. Dieser Vorgang wird auch als Bluetooth No Input No Output - Man In The Middle (BT-NINO-MITM) - Attacke bezeichnet. Das Problem besteht darin, dass das verwendete Assoziationsmodell durch den Austausch von Nachrichten bestimmt wird. Bei der Authentifizierung durch SSP werden sogenannte IO-Capabilities-Nachrichten ausgetauscht. Dabei informiert jede Partei seinen Partner über die ihm zur Verfügung stehenden Ein- und Ausgabeschnittstellen. Dabei kann jedes Gerät eines der folgenden Parameter versenden:

1. Display Only

2. DisplayYesNo (Ein Display und die Möglichkeit per Eingabe zu bestätigen oder abzulehnen ist vorhanden.)
3. KeyboardOnly (Nur eine Eingabemöglichkeit ist vorhanden, keine Ausgabe möglich.)
4. NoInputNoOutput

Nach dem Austausch der besagten IO-Capability-Nachrichten wird je nach Konfiguration ein entsprechendes Assoziationsmodell ausgewählt. Der Angriff nutzt die Tatsache aus, dass viele Geräte autoamtisch das „Just Works“-Modell auswählen, wenn Sie merken, dass der Partner weder über Ein- noch Ausgabemöglichkeiten verfügt.
Der Mögliche Ablauf:

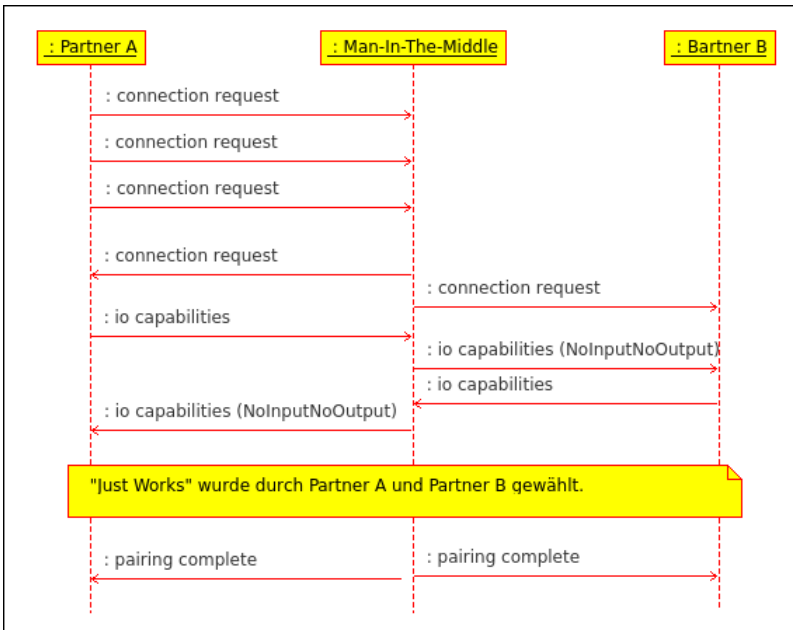

Abbildung 3.9: Der mögliche Ablauf eines MITM -Angriffes.

Um den Angriff durchzuführen, fängt der Angreifer die Verbindungsanfragen des Gerätes A ab. Dies kann zum Beispiel durch das Fälschen der Bluetooth-Adresse des Gerätes B erreicht werden. Im zweitem Schritt wird eine Verbindung zu beiden Partnern aufgebaut. Beiden Parteien wird dabei vorgemacht, dass sie miteinander kommunizieren. Das Bluetooth-Protokoll sieht vor, dass ein Link Key zu jedem Zeitpunkt verworfen werden kann. In diesem Fall ist eine Reauthentifizierung vorgesehen. Beide Parteien werden zur erneuten Authentifizierung gezwungen, wobei die IO-Capability-Packete so gefälscht werden, dass das „JustWorks“-Assoziationsmodell auf beiden

Seiten gewählt wird. Dieses Verfahren bietet jedoch keinen Schutz gegen MITM- Angriffe. Die Verbindung kann nun abgehört und die Pakete gefälscht werden.

3.4.9 Die Passkey Entry - Attacke

In seiner Arbeit „Attacks on the Pairing Protocol of Bluetooth v2.1“ beschreibt Andrew Y. Lindell einen weiteren möglichen Angriff. [Lindell (2008)] Dieser Angriff kann jedoch nur erfolgreich ausgeführt werden, wenn das „Passkey Entry“-Assoziationsmodell eingesetzt wird und das Opfer ein einmalig gewähltes Password dauerhaft gespeichert.

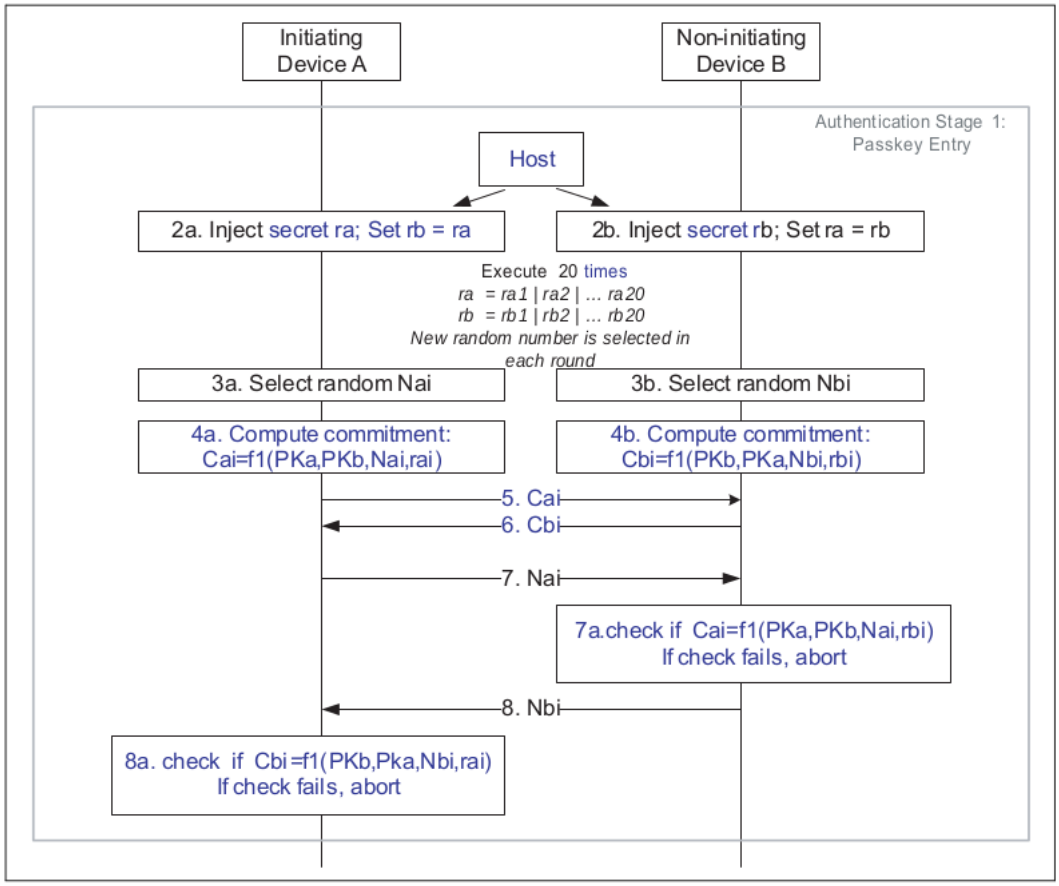

Abbildung 3.10: ,
Figure 4.10: Numeric Comparison Authentication, Figure 7.6: Authentication Stage 1: Passkey Entry Details
Abgerufen am: 25.12.2012]Das Passkey-Entry-Protokoll im Detail.

Quelle: [BTSPEZ21 (2007)], Figure 4.10: Numeric Comparison Authentication, Figure 7.6: Authentication Stage 1: Passkey Entry Details

1. In den Schritten 2_a bwz. 2_b erfolgt die Eingabe eines 6-Stelligen, aus Ziffern bestehenden, Schlüssels r_a bzw. r_b. Dieeser Schlüssel wird als PassKey bezeichnet. Die Eingabe erfolgt auf beiden Seiten, wobei die beiden Schlüssel identisch sein sollten.

2. Die Schritte 3 bis 8 werden K Mal wiederholt. K ist dabei die Anzahl der Bits, welche für die Darstellung des PassKey benötigt werden.

3. Beide Parteien generieren eine 128 Bit große Zufallszahl N.

4. Zu jedem einzelnen Bit des PassKey wird nun der Wert C mit Hilfe der Funktion f1 berechnet.

5. In den Schritten 5 - 8 werden die Zufallszahlen N ausgetauscht und die Werte C verglichen. Der bitweise Abgleich soll dabei verhindern, dass ein Erraten des PassKey möglich ist. Sollte bei einem der Bits der Abgleich fehlschlagen, so wird der Prozess unterbrochen. Der Angreifer kann also maximal zwei Bits des PassKey in Erfahrung bringen (ein Bit pro Partner).

[BTSPEZ21 (2007)]

Aus dem oben gezeigten Verfahren wird deutlich, dass ein MITM-Angriff möglich ist. Sollte der PassKey nicht bei jedes Mal neu gewählt werden, sondern fest vergeben sein, kann er durch einen Brute-Force-Angriff bitweise erraten werden. Der Angreifer versendet dabei ein zufälliges Bit B. Sollte er nach dem Versand eine Fehlernachricht erhalten, so muss das Bit B nur negiert werden. Bei jedem Versuch wird also mindestens ein Bit des PassKey erraten. Allein die Benutzung des Passkey-Entry-Verfahrens an sich bietet demnach keinen ausreichenden Schutz.

3.4.10 Vor- und Nachteile der Pairing-Verfahren im Überblick

LMP-Pairing

1. Vorteile
   a) Fast alle über einen Bluetoth-Chip verfügenden Geräte unterstützen auch LMP-Pairing.
   b) Bei der Wahl einer ausreichend großen PIN und weiteren Sicherheitsvorkehrungen ist dieses Verfahren sicher gegen Angriffe von außen.

2. Nachteile

a) Die meisten Geräte, unterstützen eine 4-Stellige PIN, welche mit relativ wenig Aufwand gecknackt werden kann.

Secure Simple Pairing

1. Vorteile
   a) Bei Vorhandensein Ein- bzw. Ausgabeschnittstellen bei beiden Parntern und der Auswahl richtiger Assoziationsmodelle ist dieses Verfahren gegen Angriffe sicher.
2. Nachteile
   a) Sollte die Software es erlauben, das Just-Works Assoziationsmodell zu benutzen, kann eventuell ein MITM Angriff erfolgreich durchgeführt werden.
   b) Ein PassKey darf nicht fest vergeben werden.
   c) Wird nur von den neueren Geräten unterstützt.
   d) Beide Partner müssen SSP unterstützen.

3.4.11 Analyse und Fazit

Natürlich gibt es weitere bekannte Arten von Angriffen auf Bluetooth-Geräte wie zum Beispiel „BlueBug“ oder „Blueprinting“. Eine Übersicht der möglichen Angriffe ist von der Vereinigung „trifinite.group“ unter dem Link „`http://trifinite.org/trifinite_stuff.html`“ zusammengefasst. In dieser Arbeit werden diese aber nicht näher dargestellt, da sie nicht die Schwächen der Bluetooth-Spezifikation ausnutzen, sondern die der jeweiligen Implementierungen. Die Sicherheit des Gesamtsystems ist eines der Hauptziele dieser Arbeit. Die Realisierung des Mikrocontroller-gesteuerten Bluetooth-basierten Zugangskontrollsystems sollte vor allem sicher vor MITM- und Lauschangriffen sein. Es muss sichergestellt werden, dass die Alarmanlage nur dann abgestellt wird, wenn sich ein authentifiziertes Gerät im Bluetooth-Bereich befindet. Zum Pairing mit dem Zugangskontrollsystems werden Smartphones eingesetzt. Ob es sich jeweils um den User oder dem Master handelt, einer der Partner bei der Authentifizierung wird immer ein Mobilfunktelefon sein. Aus diesem Grund kann das LMP-Pairing für die Realisierung nicht eingesetzt werden. Das SSP steht erst ab der Bluetooth-Version 2.1 zu verfügung. Um die Sicherheit der Authentifizierung zu gewährleisten, muss die Tatsache in Kauf genommen werden, dass ältere Geräte (Bluetooth-Version 2.0 und frühere) nicht unterstützt werden können.

3.5 Anforderungen an die Hardware

3.5.1 Anforderungen an die Hardware des Zugangskontrollsystems

Um das Zugangskontrollsystem zu realisieren, ist ein Bluetooth-Server notwendig. Damit das Gesamtsystem auch im häuslichem Bereich verwendet werden kann, muss es klein, leise und kostengünstig im Kauf und Unterhalt sein. Weiterhin sollte diese Hardware wenig Strom verbrauchen und leicht installierbar sein. Natürlich muss der Bluetooth-Server über eine Bluetooth Schnittstelle verfügen. Eine ausreichende Prozessorleistung ist von Nöten, damit die Suche nach den Benutzern und ihre Erkennung schnell genug abläuft, um die Alarmanlage rechtzeitig zu aktivieren. Jeder Nutzer muss über ein Bluetooth-Gerät verfügen, welches dass SSP-Verfahren unterstützt.

Weiterhin muss eine Alarmanlage mit einem Bewegungssensor vorliegen und vom Bluetooth-Server aus steuerbar sein. Mindestens das Ein- und Ausschalten der Anlage muss möglich sein. Ebenso wie der Server, muss diese klein und kostengünstig im Unterhalt sein.

3.5.2 Anforderungen an die Hardware des Verwaltungssystems

Das Benutzerverwaltungssystem besteht aus zwei Teilen. Ein Teil läuft auf dem Bluetooth-Server und der zweite Teil auf dem Smartphone des Masters. Die Kommunikation erfolgt zwischen den Teilrealisierungen auf Basis von Bluetooth. Aus diesem Grund müssen Sowohl das Smartphone als auch der Bluetooth-Server über eine Bluetooth-Schnittstelle verfügen und SSP unterstützen. Ein Server muss über genügend persistenen Speicher verfügen, um die Informationen zu einer gewünschten Anzahl von Benutzern abspeichern zu können.

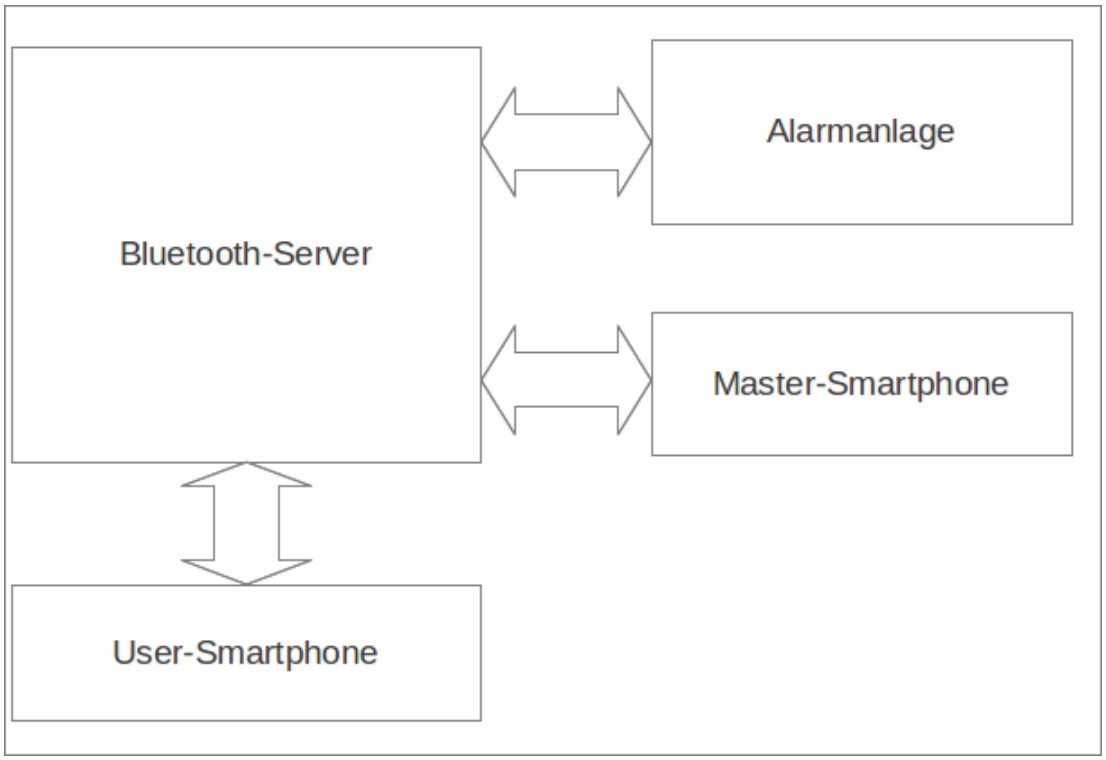

Abbildung 3.11: Das mögliche Zusammenspiel der Hardware-Komponenten im Gesamtsystem.

3.5.3 Fazit

Um dem Ziel dieser Arbeit gerecht zu werden, muss die Hardware sehr günstig sein. Da die Hardware die Alarmvorrichtung steuern soll und somit am zu schützenden Objekt installiert wird, bietet es sich an, ein Development-Board als Hardware für den Bluetooth-Server einzusetzen. Das Bord sollte über eine Schnittstelle verfügen, über die ein Bluetooth-Adapter angeschlossen werden kann. Das ganze System darf einem potenziellen Angreifer nicht zugänglich sein. Genauso darf die Stromversorgung von Unbefugten nicht unterbrochen werden können.

4 Das Design des Gesamtsystems aus der Hardwaresicht

4.1 Das Design des Zugangskontrollsystems aus der Hardware-Sicht

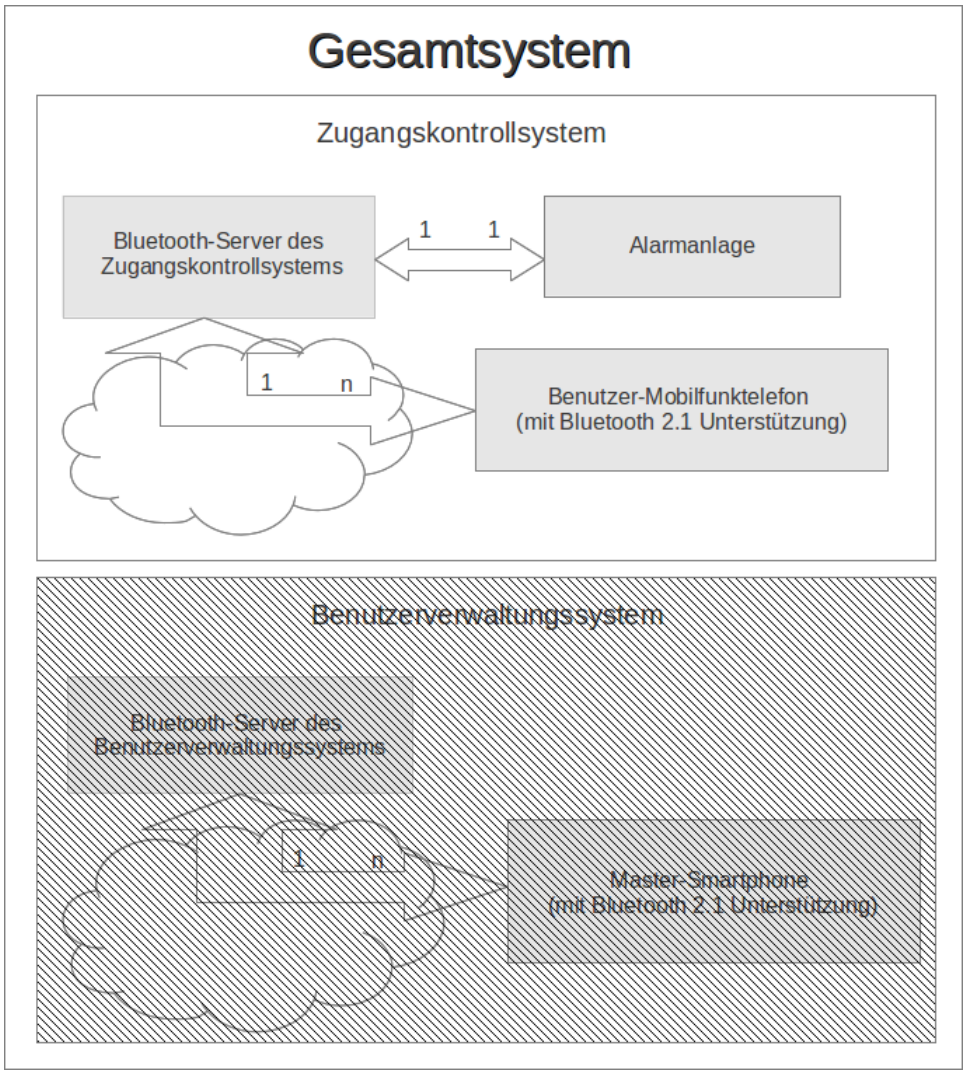

Abbildung 4.1: Das Design des Zugangskontrollsystems aus der Hardware-Sicht. Die Hardware-Komponenten sind zur Verdeutlichung durch einen grauen Hintergrund hervorgehoben.

Das Zugangskontrollsystem besteht im Wesentlichen aus drei Hardware-Komponenten. Der Bluetooth-Server ist hauptsächlich für die Überwachung der Benutzer und die Aktivierung der Alarmanlage zuständig. Es kann sich dabei um eine beliebige Alarmanlage handeln. Wichtig ist nur, dass der Bluetooth-Server über die Möglichkeit verfügt, diese ein- und auszuschalten. Die Überwachung der Benutzer muss zu jeder Zeit durchgeführt werden, da sowohl das Betreten als auch das Verlassen des Bluetooth-Bereiches durch den Benutzer sofort vom Bluetooth-Server erkannt werden muss.

4.2 Das Design des Benutzerverwaltungssystems aus der Hardware-Sicht

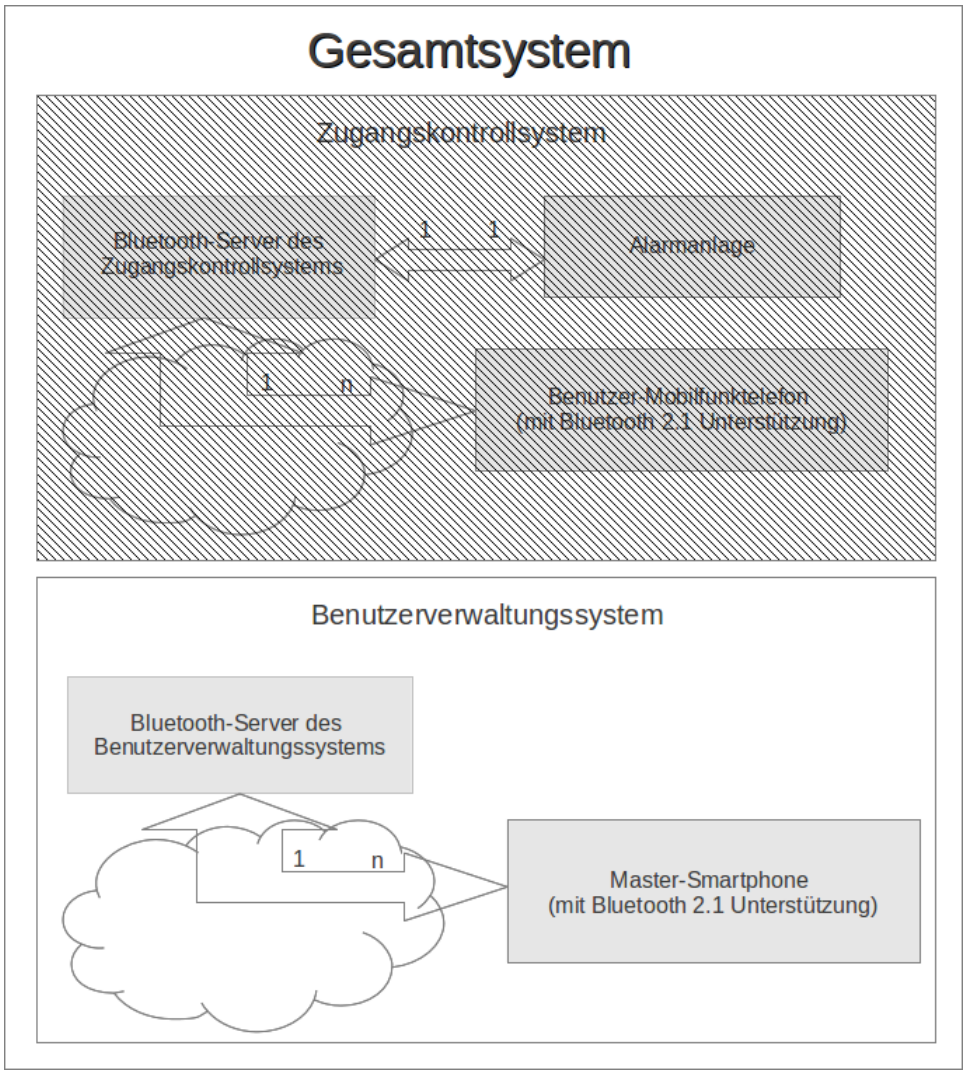

Abbildung 4.2: Das Design des Benutzerverwaltungssystems aus der Hardware-Sicht.

Der Bluetooth-Server übernimmt die Speicherung und Verwaltung der Benutzer. Das Smartphone des Masters muss die grafische Oberfläche zur entfernten Steuerung der Benutzerverwaltung zur Verfügung stellen. Zusätzlich muss jede Seite über eine Schnittstelle zur Bluetooth-Kommunikation mit dem Partner verfügen. An dieser Stelle sei angemerkt, dass das Mobilfunktelefon des Masters den Anforderungen der mobilen Benutzerverwaltungssoftware gerecht werden muss. Da ein einfacher Benutzer diese Software nicht installieren muss, kann sein Mobilfunktelefon wesentlich einfacher und kostengünstiger sein.

5 Das Design des Gesamtsystems aus der Softwaresicht

5.1 Das Design des Zugangskontrollsystems aus der Software-Sicht

Da die Bluetooth-Protokolle in Mobilfunktelefonen der Benutzer bereits implementiert sind, muss lediglich die Software des Bluetooth-Servers implementiert werden.

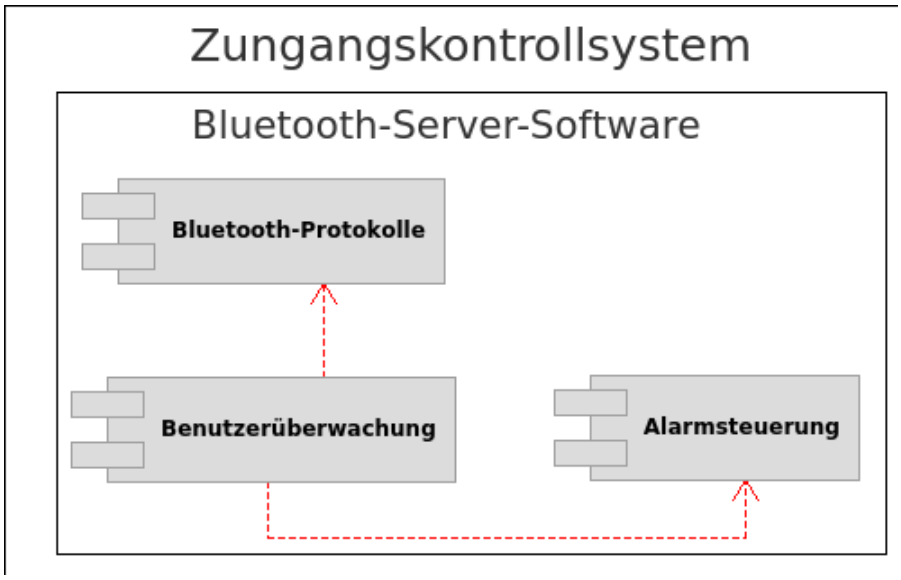

Abbildung 5.1: Das Komponentendiagramm stellt die Software-Komponenten des Zugangskontrollsystems und ihre Abhängigkeiten dar.

Das Zugangskontrollsystem besteht aus drei Software-Komponenten.

1. **Die Implementierung der Bluetooth-Protokolle**

   Zunächst müssen die notwendigen Protokolle des Bluetooth-Protokollstapels implementiert werden. Erst nach dem diese Implementierung zu Verfügung steht, kann eine Kommunika-

tion über Bluetooth stattfinden. Die Protokolle Host Controller Interface (HCI), Logical Link Control and Adaptation Layer Protocol (L2CAP), Radio Frequency Communication (RFCOMM) und Service Discovery Protocol (SDP) müssen realisiert sein, um mit einer auf Android basierten Benutzerverwaltungssoftware zu kommunizieren.

2. **Die Benutzerüberwachung**

   Wie im Kapitel 3.1 „Anwendungsszenarien“ beschrieben, übernimmt diese Komponente die Überwachung und die Erkennung der Benutzer in der Bluetooth-Reichweite.

3. **Die Alarmsteuerung**

   Die Komponente „Alarmsteuerung“ kann unterschiedlich realisiert werden. Im einfachsten Fall kann die Alarmanlage bei Notwendigkeit ein- oder ausgeschaltet werden. Jedoch kann die Art der Ansteuerung je nach dem verwendeten Alarmsystem und Sicherheitsanforderungen variieren.

5.2 Das Design des Benutzerverwaltungssystems aus der Software-Sicht

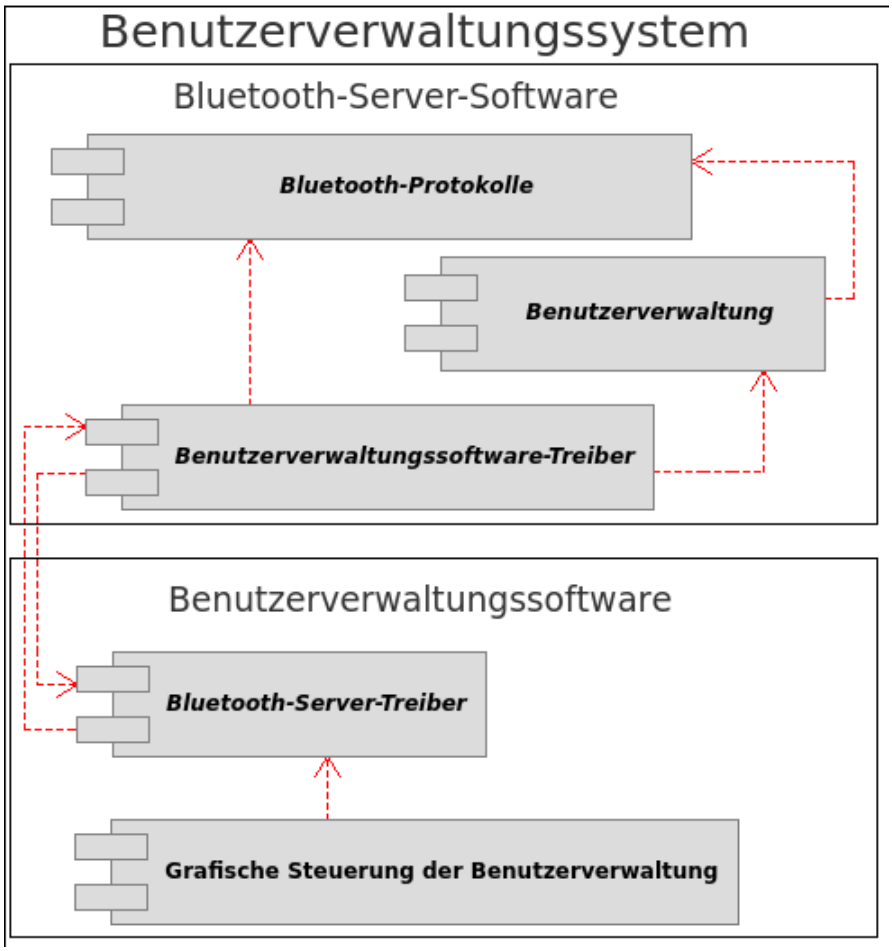

Abbildung 5.2: Das Komponentendiagramm stellt die Software-Komponenten des Zugangskontrollsystems und ihre Abhängigkeiten dar.

Das Benutzerverwaltungssystem besteht aus zwei Software-Komponenten, einem Bluetooth-Server, welcher über eine Benutzerverwaltungs-Komponente verfügt, und einer Benutzerverwaltungssoftware, welche auf einem Smartphone ausgeführt wird.

5.2.1 Die Bluetooth-Server-Software

1. Die Benutzerverwaltungskomponente

   Die Bluetooth-Server-Software muss eine Benutzerverwaltung bieten. Die Benutzer müssen mindestens hinzugefügt, gespeichert und wieder gelöscht werden können. Außerdem sollte jeder Benutzer eindeutig identifiziert werden können. Wie die Benutzerverwaltungskomponente implementiert wird, ist nicht festgelegt.

2. Der Treiber für die Benutzerverwaltungssoftware

   Wichtig ist, dass ebenfalls ein Treiber zur entfernten Steuerung durch die Benutzerverwaltungssoftware realisiert ist. Das Vorhandensein dieses Treibers ist in erster Linie dazu da, um die Fernsteuerung der Benutzerverwaltungskomponente des Bluetooth-Servers zu ermöglichen. Der Einsatz dieser Treiber ermöglicht die Übertragung der Comparison-Number, welche später vom Master abgegliechen werden kann.

3. Die Implementierung der Bluetooth-Protokolle

   Um den Treiber für die Benutzerverwaltungssoftware zu realisieren, müssen die Implementierungen für notwendigen Bluetooth-Protokolle zur Verfügung stehen.

5.2.2 Die Benutzerverwaltungssoftware

Hierbei handelt es sich um eine Anwendung, welche auf dem Smartphone des Masters ausgeführt wird. Sie ermöglicht die Fernsteuerung der Benutzerverwaltungskomponente des Bluetooth-Servers. Diese Software erlaubt die entfernte Speicherung, Entfernung und Anzeige von authentifizieren Benutzern. Weiterhin muss diese Anwendung den Master sofort informieren, falls eine Anfrage zur Aufnahme von einem neuem Benutzer kommt.

1. Der Bluetooth-Server-Treiber

   Dieser Treiber muss vorhanden sein, um mit der Benutzerverwaltungskomponente der Bluetooth-Server-Software zu kommunizieren.

2. Grafische Steuerung der Benutzerverwaltung

   Diese Komponente erlaubt eine Steuerung der Benutzerverwaltungskomponenten der Bluetooth-Server-Software. Sie erlaubt das Hinzufügen, Löschen und Aufnahme der Benutzer. Sie bietet eine grafische Benutzeroberfläche und eine Steuerung der Anwendung per Touchscreen.

5.3 Das Design des Gesamtsystems aus Software-Sicht

Da ein Bleutooth-Server sowohl eine Komponente des Zugangskontrollsystems als auch des Benutzerverwaltungssystems darstellt, bietet es sich an, das Software-Design so zu verändern, dass nur ein einziger Bluetooth-Server eingesetzt werden kann. Dies ist nicht zwingend notwendig, die beiden Systeme können auch getrennt von einander existieren. Um die Systeme in einem Server zu vereinen, müssen lediglich die Benutzerverwaltungskomponente und die Benutzerverwaltungssoftware-Treiber-Komponente aus dem Benutzerverwaltugssystem zum Modell des Zugangskontrollsystems hinzugefügt werden. Das folgende Komponentendiagramm veranschaulicht diesen Ansatz:

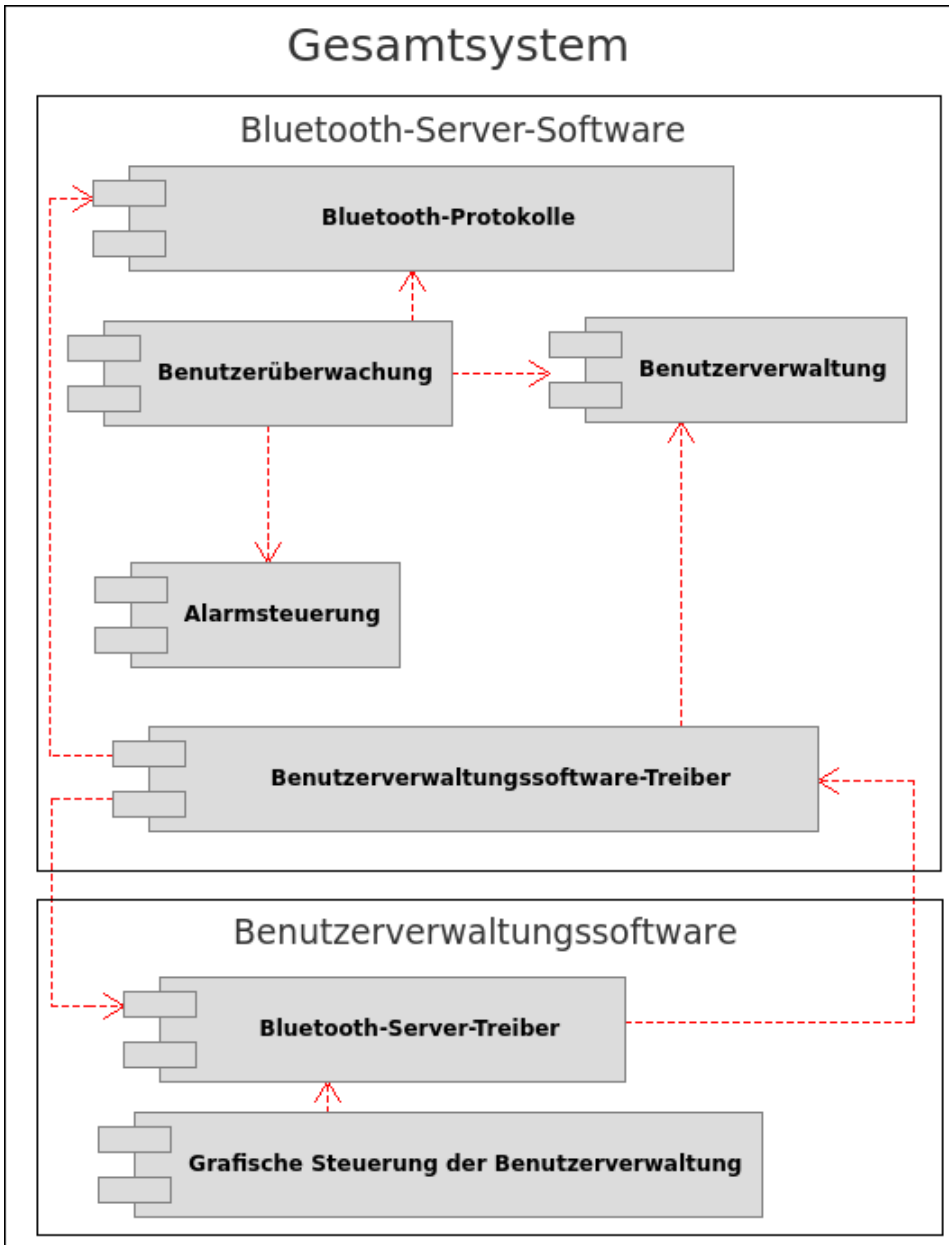

Abbildung 5.3: Das Komponentendiagramm stellt die Software-Komponenten des Gesamtsystems und ihre Abhängigkeiten unter Verwendung nur eines Bluetooth-Servers dar.

6 Die Realisierung der Soft- und Hardwarekomponenten des Gesamtsystems

6.1 Die gewählte Hardware

6.1.1 Der Bluetooth-Server

Um allen Anforderungen aus dem Kapitel 3.5 „Anforderungen an die Hardware“ gerecht zu werden, muss eine passende Hardware gewählt werden. Der Bluetooth-Server muss sehr kostengünstig, klein und stromsparend sein. Es bietet es sich deshalb an, eine mit einem preisgünstigen Mikrocontroller bestückte Hardware als Basis für ihn zu verwenden.

Die Wahl ist auf das Board STM32F4-Discovery der Firma „STMicroelectronics“ gefallen. Es verfügt über einen modernen 32-bit ARM Cortex-M4 Microcontroller aus der ARM Cortex-M-Familie, welcher genügend Leistung und Speicherplatz bietet, um das Zugangskontroll- und Verwaltungssystem zu realisieren. Ein weiterer Vorteil dieses Microkontrollers ist, dass es sehr verbreitet ist und daher von einer Vielzahl von Entwicklungstools unterstützt wird.

Weiterhin verfügt das Gerät über 1 MB Flash und192 KB RAM Speicher. Das „ST-LINK/V2“-System ist verfügbar, welches das Programmieren und Debuggen ohne den Einsatz weiterer Hardware ermöglicht. Eine USB On-the-go (USB-OTG)-Schnittstelle ist ebenfalls verfügbar, welche ein Anschließen des Bluetooth-Dongle möglich macht. [Belostotsky (2012)] Bei dieser Ausstattung ist das Development-Board im Vergleich sehr günstig. Der Preis des STM32F4-Discovery liegt derzeit bei circa 16 Euro. Aufgrund der oben genannten Eigenschaften eignet sich dieses Gerät hervorragend für die Realisierung des Bluetooth-Servers des Zugangskontrollsystems. [www.idealo.de (2013)]

6.1.2 Das STM32F4-Discovery-Board im Überblick

Key Features:

1. STM32F407VGT6 microcontroller featuring 32-bit ARM Cortex-M4F core, 1 MB Flash, 192 KB RAM in an LQFP100 package
2. On-board ST-LINK/V2 with selection mode switch to use the kit as a standalone ST-LINK/V2 (with SWD connector for programming and debugging)
3. Board power supply: through USB bus or from an external 5 V supply voltage
4. External application power supply: 3 V and 5 V
5. LIS302DL, ST MEMS motion sensor, 3-axis digital output accelerometer
6. MP45DT02, ST MEMS audio sensor, omni-directional digital microphone
7. CS43L22, audio DAC with integrated class D speaker driver
8. Eight LEDs
   a) LD1 (red/green) for USB communication
   b) LD2 (red) for 3.3 V power on
   c) Four user LEDs, LD3 (orange), LD4 (green), LD5 (red) and LD6 (blue)
   d) 2 USB OTG LEDs LD7 (green) VBus and LD8 (red) over-current
9. Two push buttons (user and reset)
10. USB OTG FS with micro-AB connector
11. Extension header for all LQFP100 I/Os for quick connection to prototyping board and easy probing

Based on the STM32F407VGT6, it includes an ST-LINK/V2 embedded debug tool, two ST MEMS, digital accelerometer and digital microphone, one audio DAC with integrated class D speaker driver, LEDs and push buttons and an USB OTG micro-AB connector.

Quelle: [STM32F4 (2011)]

6.1.3 Das Bluetooth-Dongle

Es gibt bereits ein Projekt unter dem Namen „Uubt“, welches die Bluetooth-Kommunikation auf diesem Board realisiert. [Belostotsky (2012)] Dabei wird das STM32F4-Discovery-Board über den USB-OTG-Anschluss mit einem Bloetooth-Dongle verbunden. Da der Source-Code dieses Projektes zur Verfügung steht, bietet es sich an, auf diesen zurückzugreifen. Die Uubt-Anwendung wurde laut dem Autor nur mit Bluetooth-Adaptern getestet, welche entweder auf dem „Atheros“ oder „CSR“ Chipsatz basieren. Bei dem Einsatz dieser Software ist zu beachten, dass darin ein Patch für die USB-Bibliotheken von „STMicroelectronics“ enthalten ist. Darüber hinaus muss der Dongle über das Secure Simple Pairing (SSP) verfügen, daher mindestens die Bluetooth-Spezifikation 2.1 unterstützen.

In dieser Arbeit wurde der Adapter „BT0007 USB 2.0 to Bluetooth V2.1 EDR Micro“ der Firma „LogiLink“ eingesetzt. Nach mehreren Tests hat sich gezeigt, dass dieser Adapter zuverlässig mit der „Uubt-Software“ funktioniert. Dieser wurde mit Hilfe eines USB-Adapters mit dem Micro-USB-Anschluss des STM32F4-Discovery verbunden. Natürlich können auch andere Adapter verwendet werden. Dies ist besonders dann interessant, wenn man die Reichweite des Bluetooth-Servers ändern möchte. In der Bluetooth-Spezifikation gibt es 3 Klassen, in welche die Geräte nach Sendeleistung eingeteilt werden:

1. Class 1: 20 dBm (100 mW), max. Reichweite ca. 100m
2. Class 2: 4 dBm (2,5 mW), max. Reichweite ca. 50m
3. Class 3: 0 dBm (1 mW), max. Reichweite ca. 10m

Um die Reichweite einschätzen zu können, ist es sehr ratsam, die Angaben der Hersteller zu beachten. Die oben genannte Zuordnung ist nicht bindend. Das ausgewählte Dongle von „LogiLink“ gehört zu der zweiten Klasse und verspricht nach Herstellerangaben eine maximale Reichweite von 20m. Bei der Aufstellung des Bluetooth-Servers muss beachtet werde, dass durch die Umgebung verkürzte Reichweite immer noch groß genug ist, um den Alarm rechtzeitig zu deaktivieren.

Ein Bluetooth-Gerät muss nicht unbedingt über die USB-Schnittstelle mit dem STM32F4-Discovery verbunden sein. Um Strom- oder Anschaffungskosten zu sparen, sind auch andere Lösungen denkbar. Jedoch bietet die vorgestellte Lösung den Vorteil, dass man vor der Initialisierung des Gesamtsystems den Bluetooth-Dongle leicht ersetzten kann.

6.1.4 Der Hardware für die Verwaltungssoftware

Die Verwaltungssoftware wird als eine Andrid-App realisiert. Sie kann somit auf jedem Gerät ausgeführt werden, auf dem Android als Betriebssystem installiert ist. Außerdem muss die Hardware über eine Bluetooth-Schnittstelle verfügen. In dieser Arbeit wurde das Smartphone „Huawei Ideos X3“ verwendet. Das Design der App wurde auf dessen Display angepasst. Eine Anpassung des Layouts an andere Android-Geräte würde den Rahmen dieser Arbeit sprengen.

6.1.5 Das Alarmsystem

Theoretisch kann jedes beliebige Alarmsystem durch das Discovery-Board gesteuert werden. Sollte ein autorisierter Benutzer in der Reichweite identifiziert werden, so wird das Alarmsystem deaktiviert. Sollte kein Benutzer in der Nähe sein, so muss es aktiviert werden. Die einzige Voraussetzung dabei ist, dass eine Schnittstelle zum Ein- bzw. Ausschalten durch das Development-Board vorhanden ist.

Bei dieser Realisierung wird ein sehr einfaches Alarmsystem verwendet. Das Gerät verfügt über einen Infrarotsensor zum Ein- bzw. Ausschalten. Außerdem ist ein Bewegungssensor vorhanden. Sollte die Alarmanlage angeschaltet sein und eine Bewegung detektieren, so ertönt nach wenigen Sekunden ein lautes Signal. Erst beim Ausschalten der Anlage, wird dieses unterbrochen. Zu der Anlage gehören zwei Infrarot-Fernbedienungen zur Steuerung. Um das Alarmsystem vom Discovery-Board zu steuern, wurde seine Schaltung entsprechend verändert. Anstelle des Infrarot-Signals von der Fernbedienung zu aktivieren bzw. deaktivieren, wurden Drahtleitungen verlegt, welche nun zu den GPIO-Anschlüssen des STM32F4 führen. Nun kann die Alarmanlage durch die Zugangskontrollsoftware gesteuert werden. Dabei hat sie eine eigene Stromversorgung, welche vor unbefugten unbedingt geschützt sein muss.

6.2 Die Realisierung des Gesamtsystems aus der Hardwaresicht

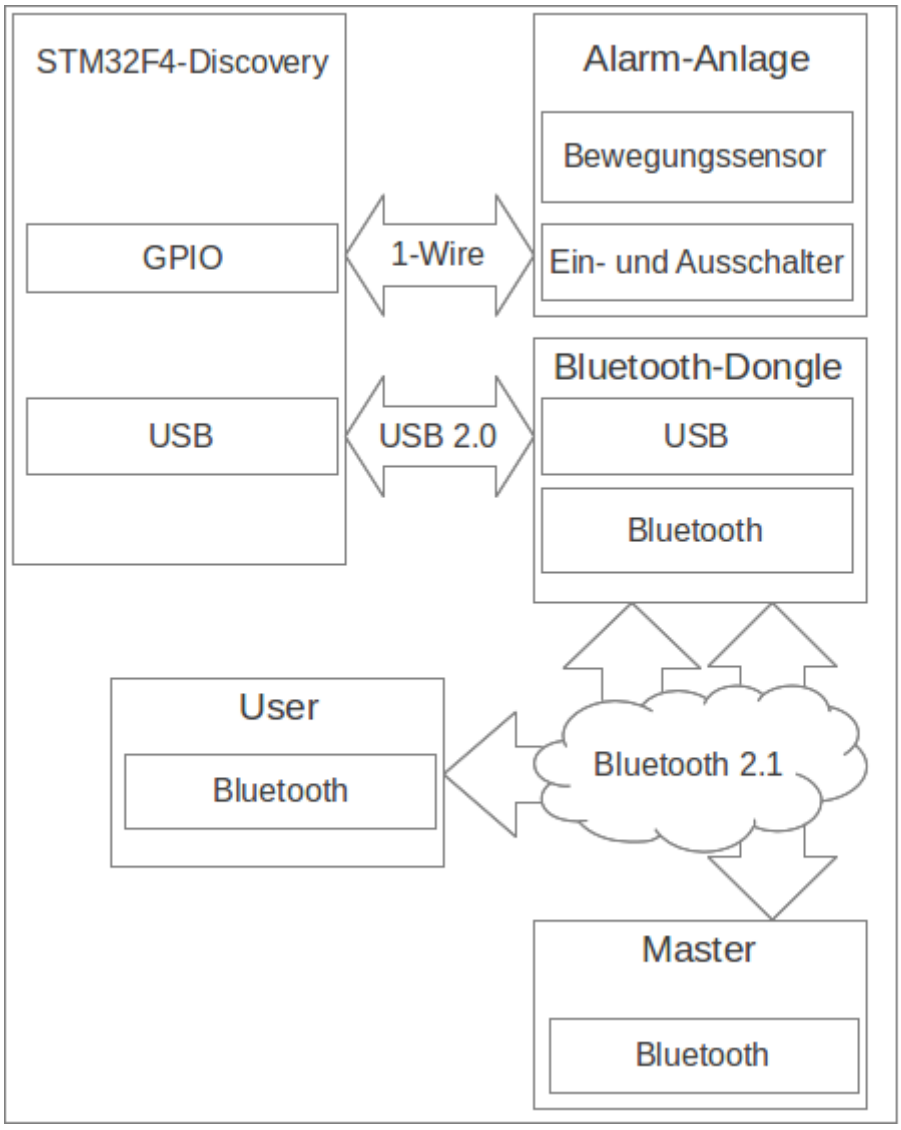

Abbildung 6.1: In diesem Diagramm wird das Zusammenspiel der eingesetzten Hardware veranschaulicht.

6.3 Die Entwicklungsumgebung für STM32F4-Discovery

6.3.1 CoIDE als Entwicklungsumgebung

Als Entwicklungsumgebung für das Zugangskontorollsystem ist die Wahl auf CooCox CoIDE gefallen. Es handelt sich um eine freie Entwicklungsumgebung für „ARM Cortex MCU“ basierte Mikrocontroller. Obwohl CoIDE frei zur Verfügung steht, gibt es keine Beschränkung der Code-Länge, wie es bei den meisten Alternativen der Fall ist.

Weiterhin wird das „STM32F4-Discovery-Board“ unterstützt. Daher können die notwendigen Bibliotheken und Konfigurationsdateien aus dem „CooCox-Repository“ geladen werden. Die Debug-Adapter ST-Link oder auch ST-Link/V2 werden unterstützt. Ein Assistent bei der Projekterstellung erlaubt die Auswahl der eingesetzten Hardware, was die Konfiguration erheblich erleichtert. Die CoIDE basiert auf Ecplise, es können daher meist die selben „Shortcuts“ benutzt werden.

CoIDE bietet die meisten Werkzeuge, welche auch teure, professionelle Lösungen der Konkurrenten liefern. Break- und Watchpoints können gesetzt werden. Die Werte der Register können in einer entsprechenden Ansicht dargestellt werden. Die Disassemblierung ist möglich. Speicheranzeige und weitere Funktionen stehen zur Verfügung.

6.3.2 STM32F4-Discovery-Board und CoIDE

CooCox-CoIDE unterstützt das ST32MF4-Discovery-Board, daher sind nach der Projekterstellung die wichtigsten Bibliotheken und Konfigurationsdateien sofort verfügbar. Um mit der Entwicklung zu beginnen, sind jedoch eventuell folgende Schritte erforderlich:

1. Einstellung des Compilers

   Als Compiler wurde der GCC-ARM-Compiler verwendet. Nach der Installation des Compilers muss dieser unter „Project/SelectToolchainPath"“ in der CoIDE gewählt werden.

2. Semihosting-Ausgabe Aktivierung

   Definition 23 (*Semihosting*)

   > *„Semihosting is a mechanism for ARM targets to communicate input/output requests from application code to a host computer running a debugger. This mechanism could be used, for example, to enable functions in the C library, such as printf() and scanf(), to use the screen and keyboard of the host rather than having a screen and keyboard on the target system. This is useful because development hardware often does not have all the input and output facilities*

> *of the final system. Semihosting enables the host computer to provide these facilities.“ [ARM Limited (2005)]*

Um das SEMIHOSTING zu aktivieren, müssen folgende Schritte durchgeführt werden.

a) In der Ansicht „Repository“ muss das Semihosting ausgewählt werden. Dabei werden die notwendigen Bibliotheken nachgeladen.

b) Nun muss die Funktion printchar(char c) in der Datei printf.c so verändert werden, dass sie die Funktion SH_SendChar(c) aus der semihosting.h Datei aufruft.

c) Unter dem Menü „DebugConfiguration“ muss Semihosting aktiviert werden.

3. Die Taktung der externen MCU-Komponenten aktivieren.

a) Die Zeile 142 in startup_stm32f4xx.c Datei („//extern void SystemInit(void); /*!< Setup the microcontroller system(CMSIS) */“) muss auskommentiert werden.

b) Der SystemInit(void); Aufruf in die Datei startup_stm32f4xx.c in Zeile 291 muss vor vor main(); eingefügt werden.

c) Die Konfigurationsparameter können in der SystemInit(void) angepasst werden.

6.4 Die eingesetzte Software

6.4.1 Das BT-Stack-Framework als Grundlage für den BT-Server

Der Author, Matthias Ringwald, beschreibt das Projekt „BT-Stack“ wie folgt:

„The aim of this project is to support devices for which the OS either does not provide a Bluetooth Stack or the available stack is severely limited (e.g., on the iPhone - more than 2.5 million installations). In addition, BTstack is well suited for small, resource-constraint devices such as 8 or 16 bit embedded systems as it is highly configurable and comes with an ultra small memory footprint. A minimal configuration for an SPP server on a MSP430 can run in 32 kB FLASH and only 4 kB of RAM.

On larger POSIX systems, it provides a user-space daemon that connects to the Bluetooth modules via different Bluetooth HCI transport layers (e.g., HCI H4 UART and H5 the "Tree-Wire"protocol). Multiple applications can communicate with this daemon over different inter-process communication methods.

On embedded systems, a minimal run loop implementation allows to use BTstack without a Real Time OS (RTOS). If a RTOS is already used, BTstack can be integrated and run as a single

thread. The source repository provides ports for different MSP430 development boards. Other platforms can be targeted by providing the necessary UART, CPU, and CLOCK implementations, see MSP430GettingStarted and EmbeddedSystems. “ [Ringwald (2012)]

Das Projekt kann für nicht kommerzielle Zwecke frei genutzt werden. Es bildet die Grundlage für den Bluetooth-Server des Zugangskontrollsystems in meiner Arbeit. Diese Implementierung des Bluetooth-Stacks kann ohne den Einsatz eines Betriebssystems und mit sehr geringen Anforderungen an die Hardware genutzt werden. Vor allem wurden die Implementierungen des HCI - , des L2CAP - , des RFCOMM-, und des SDP - Protokolls verwendet.

6.4.2 Uubt-Projekt als Schnittstelle zwischen USB und BT-STack

Da das Discovery-Board die Funktion eines Bluetooth-Servers erfüllen soll, muss ein Bluetooth-Gerät an das Board angeschlossen werden. Das STM32F4-Discovery-Board hat zwei USB-Anschlüsse. Eines davon unterstützt den USB-OTG -Standard und kann somit die Host-Funktionalität übernehmen. Da „STMicroelectronics“ (der Hersteller des STM32F4-Discovery) sowohl USB- als auch USB-OTG-Treiber bereits zur Verfügung stellt, bietet es sich an, diese Bibliotheken zu nutzen und einen Bluetooth-Dongle über USB anzuschließen. Um das Bluetooth-Gerät zu steuern, ist die Implementierung einer Schnittstelle zwischen dem BT-Stack-Projekt und dem USB-OTG-Treiber notwendig. Genau diese Schnittstelle ist im Uubt-Projekt implementiert.

Der Autor, Vitaly Belostotsky, beschreibt das Uubt-Projekt wie folgt:

„This is a demo application for bluetooth USB dongle connected to STM32F4DISCOVERY (http://www.st.com/internet/evalboard/product/252419.jsp) boardbased on BTstack (http://code.google.com/p/btstack) project and ST USB libraries. “ [Belostotsky (2012)]

Diese Software erlaubt es, einen mit dem STM32F4-Discovery-Board verbundenen Bluetooth-Dongle zu nutzen. Als Grundlage dient dabei das BT-Stack-Framework. Das Beispiel „SSP-Counter“ (http://code.google.com/p/btstack/source/browse/trunk/MSP-EXP430F5438-CC256x/example/spp_counter.c) wurde dabei so angepasst, dass es auf dem Discovery-Board in Verbindung mit einem Bluetooth-Gerät lauffähig ist. Die Anwendung erlaubt es, eine Verbindung zum Board über das Radio Frequency Communication (RFCOMM)-Protokoll aufzubauen und in bestimmten Abständen eine Nachricht mit dem aktuellen Zählerwert zu erhalten. Hier muss angemerkt werden, dass diese Anwendung nur mit USB-Doungles der Chip-Hersteller „CSR“ und „Atheros“ funktioniert. Je nach Chipsatz des Gerätes, muss die Software von Hand angepasst werden. Der Chipsatz wird meist nicht auf der Verpackung angegeben und kann daher meist erst nach dem Kauf ausgelesen werden.

Es wurde also die Schnittstelle zwischen dem STM-USB-Treiber und dem BT-Stack-Framework in dieser Anwendung umgesetzt. Das folgende Bild zeigt die Softwareschichten, welche in diesem Projekt für die Realisierung der Bluetooth-Kommunikation zusammenwirken. Die Bluetooth-Server-Software nutzt das BT-Stack-Framework, welches den Bluetooth-Stack implementiert. Dieses nutzt wiederum die Implementierung der Schnittstelle zum STM-USB-Treiber aus dem Uubt-Projekt.

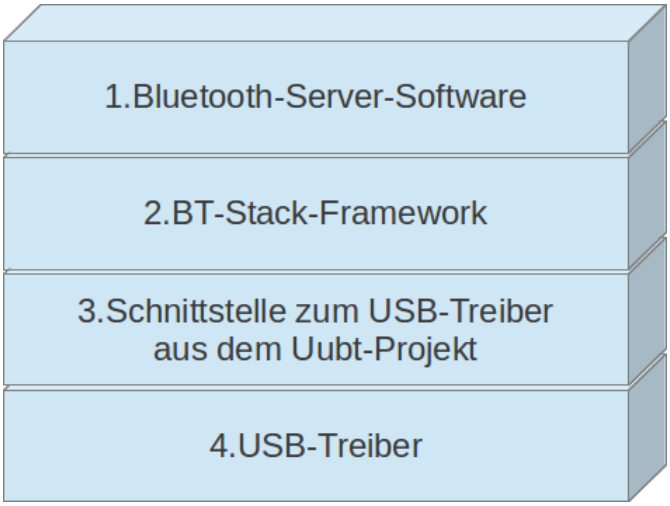

Abbildung 6.2: Vier Software-Schichten zur Ansteuerung des Bluetooth-Dongles.

6.4.3 Die Bluetooth-Bibliothek als Grundlage der auf Android basierenden Benutzerverwaltungssoftware

Die Benutzerverwaltungssoftware ist als eine Android-App realisiert, während alle anderen Software-Komponenten auf dem STM32F4-Discovery-Board ausgeführt werden. Damit die Kommunikation zwischen dem Master und dem Bluetooth-Server erfolgen kann, muss die Benutzerverwaltungssoftware den Bluetooth-Protokoll implementieren. Zum Einsatz kam die Bluetooth-Application Programming Interface (API) aus dem Android-Software Development Kit (SDK)-Packet. Diese Bibliothek bietet die notwendigen Werkzeuge, um einen Bluetooth-Server zu implementieren. Weiterhin ist diese Bibliothek gut dokumentiert und in einer stabilen Version verfügbar. Obwohl die notwendigen Bluetooth-Protokolle (HCI, L2CAP, RFCOMM und SDP) realisiert sind, bietet diese Bibliothek nur wenige Funktionen. So ist eine direkte Ansteuerung

des Bluetooth-Gerätes nicht möglich. In der folgenden Tabelle sind die Möglichkeiten dieser API zusammengefasst.

Using the Bluetooth APIs, an Android application can perform the following:

1. Scan for other Bluetooth devices
2. Query the local Bluetooth adapter for paired Bluetooth devices
3. Establish RFCOMM channels
4. Connect to other devices through service discovery
5. Transfer data to and from other devices
6. Manage multiple connections

Quelle: [ANDROIDAPI (2013)]

Durch den Einsatz dieser Bibliothek wird festgelegt, dass der Master mit dem Bluetooth-Server über RFCOMM kommunizieren muss.

6.4.4 Der BluetoothChat-Beispiel als Grundlage für die Services der auf Android basierenden Benutzerverwaltungssoftware

Der Android-SDK enthält unter anderem auch Programmbeispiele. Diese Beispiele sind unter dem Verzeichnis „Samples“ nach der Installation des SDK-Paketes zu finden. Im Beispiel ist eine Anwendung unter dem Namen „BluetoothChat“ realisiert, welche das Versenden von Nachrichten über Bluetooth erlaubt. [BTCHAT (2009)] Dabei wird die Bluetooth-Bibliothek aus dem Kapitel 6.4.3 „Die Bluetooth-Bibliothek als Grundlage der auf Android basierenden Benutzerverwaltungssoftware“ verwendet.

Diese Anwendung realisiert sowohl einen Bluetooth-Server als einen Bluetooth-Client. Über RFCOMM werden die Nachrichten zwischen zwei in Verbindung stehenden Benutzern ausgetauscht. Dieses Programm wurde zur Grundlage der Benutzerverwaltungssoftware. Vor allem die Softwarekomponenten „BluetoothLocalServer“ und „ConnectionManager“, welche für die Unterhaltung der Verbindung zuständig sind, wurden auf Grundlage des BluetoothChat-Beispiels erstellt.

6.5 Die eingestetzen Debugging-Werkzeuge

Um die Funktion des Software nachvollziehen und somit mögliche Fehler erkennen zu können, wurden spezielle Werkzeuge eingesetzt. In diesem Kapitel werden diese Werkzeuge und ihre Funktion näher erläutert.

6.5.1 Der ST-LINK-Debugger

Das STM32F4-Discovery-Board bietet eine eingebaute „ST-LINK/V2"-Schaltung der Firma „STMicroelectronics". Diese macht es möglich, das Board ohne den Einsatz zusätzlicher Hardware zu programmieren und zu debuggen. Als Debugger muss in der CoIDE der ST-LINK-Debugger im Serial Wire Debugging (SWD)-Mode gewählt werden, dann kann der Programmcode analysiert werden.

6.5.2 USB Explorer der Firma Ellisys

Allein die Möglichkeit, den Programmcode mit einem Debugger zu analysieren, hilft oft nicht weiter. Bei der Realisierung der Bluetooth-Kommunikation müssen die Abläufe in den einzelnen Protokollen streng überprüft werden. Versendete Pakete müssen stets auf ihre Richtigkeit, Vollständigkeit und das richtige Timing geprüft werden. Diese Aufgabe kann durch den alleinigen Einsatz eines Debuggers nicht bewältigt werden.

Zum Einsatz kam das Gerät „USB Explorer" der Firma „Ellisys". Der „USB Explorer" macht eine Aufzeichnung der Kommunikation auf der USB-Schiene möglich. Die mitgelieferte Software bildet die gespeicherten Daten entsprechend dem USB 2.0-Protokoll ab. Das Gerät wird zwischen den STM32F4-Discovery und den Bluetooth-Dongle geschaltet. Die Zwischenschaltung des Explorers wird von den beiden kommunizierenden Komponenten nicht wahrgenommen. Es sind keine Nebeneffekte zu erwarten.

Abbildung 6.3: Der USB Explorer wird zwischen das STM32F4-Discovery-Board und dem Bloetooth-Dongle geschaltet.

Durch den Zukauf zusätzlicher Upgrades kann die Ellisys-Software die Analyse erleichtern. Dabei werden Datenpakete entsprechend der verwendeten Protokolle (zum Beispiel Host Controller

Interface (HCI)) aufgearbeitet und dargestellt. So kann die Kommunikation leicht nachvollzogen werden, da die Übersicht der Pakete mit entschlüsselten Parametern zur Verfügung steht. Das ist besonders dann von Bedeutung, wenn ein Protokoll der höheren Ebene des Bluetooth-Protokollstapels verwendet wird. Wird zum Beispiel ein RFCOMM Paket analysiert, so enthält es die Header des RFCOMM, des L2CAP und des HCI Protokolls. In diesem Fall ist die Analyse besonders schwierig, da jedes einzelne Paket entschlüsselt werden muss.

6.5.3 Die Wireshark Software

Um die Abläufe der Protokolle zu analysieren wurde Analysesoftware Wireshark eingesetzt. Diese Anwendung unterstützt die wichtigsten Protokolle des Bluetooth-Protokollstapels. Durch die Entschlüsselung der Pakete wird die Analyse sehr erleichtert. Eine Java-Anwendung von Prof. Dr. rer. nat. Hans Heinrich Heitmann hat es ermöglicht, die aufgezeichneten Daten der USB-Schiene in die Wireshark Software zu importieren. Dabei wurde die Funktion der Ellisys-Software genutzt, um die Daten in Form einer CVS-Datei zu exportieren. Diese CVS-Datei wurde dann mit Hilfe der oben genannten Java-Anwendung in das für Wireshark lesbare pppdump-Format konvertiert.

6.6 Die Softwarekomponenten des Bluetooth-Servers

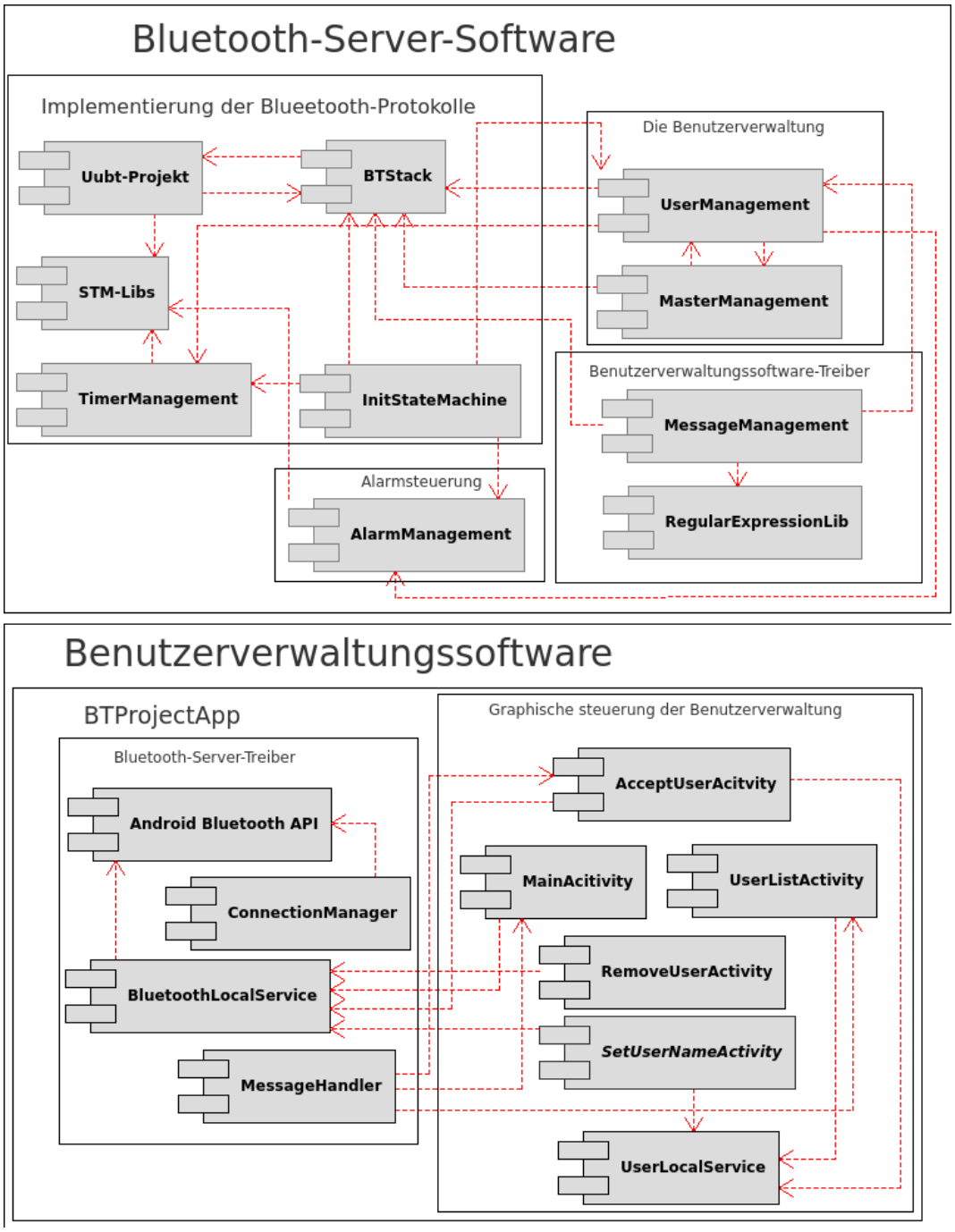

Abbildung 6.4: Dieses Komponentendiagramm veranschaulicht die Software-Komponenten des Gesamtsystems. Die Pfeile stellen dabei nicht die logischen, sondern die durch die Implementierung bedingten Abhängigkeiten der Softwarekomponenten untereinander dar.

6.6.1 Die InitStateMachine-Komponente

Diese Komponente übernimmt die Initalisierung des Bluetooth-Servers. Der „InitBTServer“-Automat wurde mit Hilfe von „YAKINDU“ (`http://www.yakindu.org`) realisiert. Bei „YAKINDU“ handelt es sich um einen Plugin für Eclipse. Der Automat wurde grafisch erstellt, wobei der C-Code vom Plugin automatisch generiert wurde. Um den Automaten zu nutzen, wurde der „InitBTServerSMController“ implementiert. Dieser erlaubt es, die Funktionalität der einzelnen Zustände vom eigentlichen „InitBTServer“-Automaten zu trennen. Auf diese Weise konnte der Automat verändert werden, ohne das jedes Mal die Funktionalität aller Zustände angepasst werden musste. Die Namen der Zustände sind schwer lesbar, da sie zum Teil auch aus den Bezeichnungen der Regionen bestehen, in den die Zustände graphisch defeniert wurden. Dies wird aus dem folgenden Beispiel deutlich:

```
case _InitBTServer_mainRegion_Init_HCI_region0_hci_ssp_enable:
        // 0x01 -> enable
        hci_send_cmd(&hci_write_simple_pairing_mode, 0x01);
break;
```

Der Übergang zum nächsten Zustand wird durch die Funktion

```
void fire_init_bt_server_event(const uint32_t evid);
```

ermöglicht. Die zu übergebenden EventIDs sind durch „YAKINDU“ nach der automatischen Code-Generierung vorgegeben. Sollte sich ein Zustand nach der Neugenerierung des Automaten geändert haben, so wird ein Fehler im „InitBTServerSMController“ durch den Compiler angezeigt.

Der Subautomat „Init_HCI“ übernimmt die Initialisierung des Bluetooth-Dongles. Diese erfolgt durch das Versenden von HCI-Packeten. Die Konfiguration ist in dieser Arbeit auf ein Minimum beschränkt.

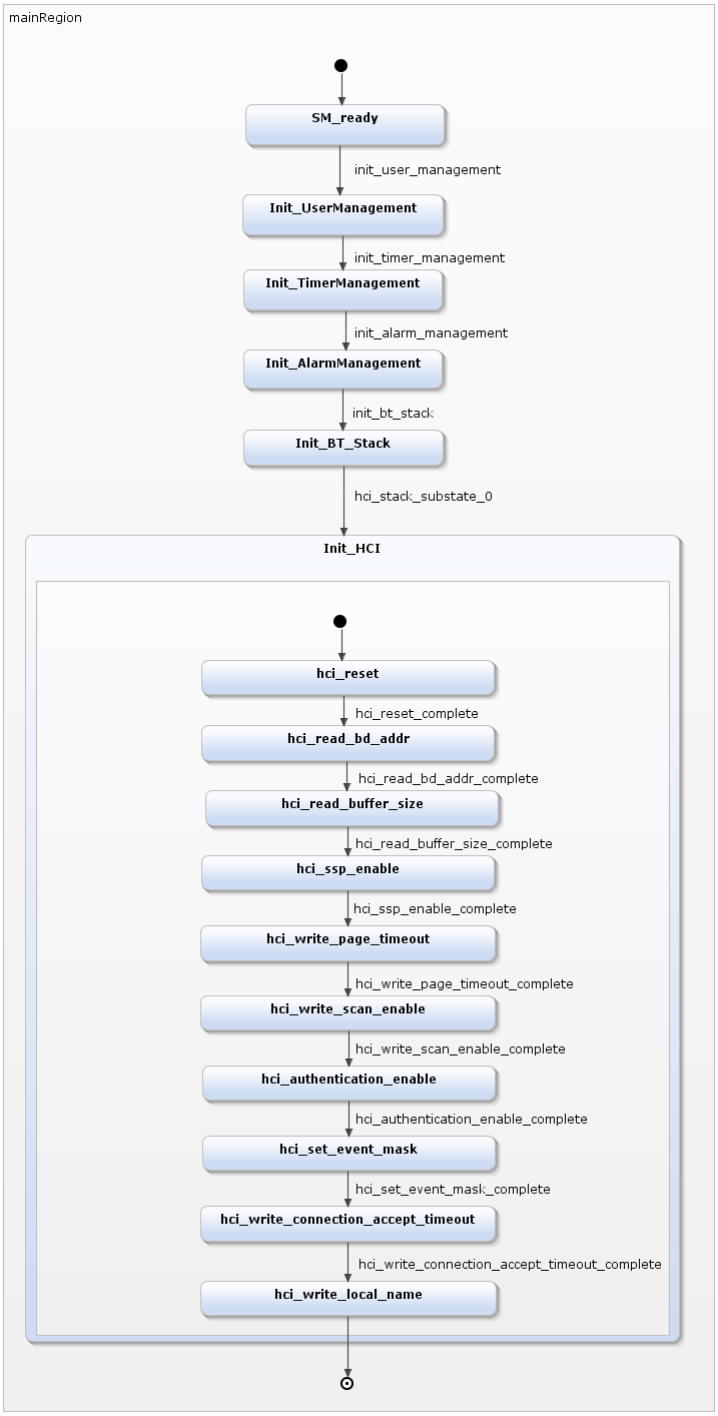

Abbildung 6.5: In diesem Zustandsdiagramm sind die Zustände des InitBTServer-Automaten abgebildet. Dieser Automat ist der Hauptbestandteil der InitStateMachine-Komponente. Der Subautomat „Init_HCI“ ist für die Initialisierung der Bluetooth-Dongles zuständig.

6.6.2 Die UserManagement-Komponente

Die Methoden der UserManagement-Komponente sind in der Datei user_management.c realisiert. Bei der Initialisierung wird durch diese Komponente ein Array über die Struktur „haw_user“ erstellt.

```
typedef struct {
        uint16_t secondsAfterLastDummyPacket;
        bd_addr_t address;
        hci_con_handle_t con_handle;
        char name[25];
        uint8_t isMaster;
} haw_user;
```

Die UserManagement-Komponente erlaubt es, auf die Werte der „haw_user“-Struktur zuzugreifen. Diese Struktur speichert alle Daten zu authentifizierten Usern ab. Das Link Key wird ebenfalls abgespeichert, aber die Verwaltung dieser Schlüssel übernimmt das BT-Stack-Framework. Somit ist dieser Wert nicht in der „haw_user“-Struktur vorhanden. Die UserManagement-Komponente bietet die Funktion „getLinkKey()“, welche den Link Key zu einem bestimmten User zurück gibt.

In dieser Arbeit wird das Array im Random Access Memory (RAM) abgespeichert. Es ist aber zu empfehlen, das Array im Flash-Speicher abzulegen, damit auch nach einem Neustart die Benutzerliste wieder verfügbar ist. Die Speicherung im Flash-Speicher würde einen zusätzlichen Aufwand mit sich bringen, welcher den Rahmen dieser Arbeit sprengen würde. Das Array über die „haw_user“-Strukturen wird zunächst mit Dummy-Usern befüllt, welche als Platzhalter dienen. Die Größe dieses Arrays hängt von der vordefinierten Konstanten „MAX_USER_COUNT“ ab. Sobald ein User vom Master akzeptiert ist, wird dieser durch die UserManagement-Komponente dem haw_user_array hinzugefügt. Die Struktur

```
hci_con_handle_t con_handle;
```

enthält den Wert Null, solange keine Verbindung zum User besteht. Sobald ein Benutzer in der Reichweite vom Bluetooth-Server entdeckt worden ist und eine sichere Verbindung zu diesem besteht, wird dieser Wert entsprechend gesetzt. Der Name des Benutzers wird vom Master vergeben. Während der Authentifizierung durch die Verwaltungssoftware hat er die Möglichkeit, jedem Benutzer einen Namen zuzuordnen. Sollte die maximale Anzahl von Benutzern erreicht werden, so wird ein Fehler ausgegeben und es werden keine weiteren Einträge akzeptiert. Obwohl die Struktur für jeden Benutzer den Wert

```
uint8_t isMaster;
```

abspeichert, darf zur gleichen Zeit nur ein Master im System angemeldet sein. Der Master selbst darf keine weiteren Master-Benutzer festlegen. Das Löschen bereits vorhandener Nutzer ist in dieser Komponente ebenfalls vorgesehen.

6.6.3 Die TimerManagement-Komponente

Bei der Durchführung von Tests mit unterschiedlichen Smartphones musste ich feststellen, dass je nach Gerät das Verhalten nach dem Aufbau einer Logical Link Control and Adaptation Layer Protocol (L2CAP)-Verbindung unterschiedlich ist. Das liegt daran, dass die Bluetooth-Spezifikation nicht genau festlegt, wie sich ein Gerät bei einer offenen Verbindung verhalten soll. Viele Geräte schließen nach einem Timeout automatisch die Verbindung, falls kein Datenverkehr besteht. Die eingestellten Time-Outs sind aber von Gerät zu Gerät unterschiedlich. Es gab auch einige Geräte, welche den Transfer nicht überwacht haben, und somit die Verbindung weiter offen blieb. Um die Verbindung zum User in der Reichweite aufrecht zu erhalten, wurde die Timemanagement-Komponente eingeführt. Ein Timer erhöht dabei jede Sekunde den Wert der Variable „secondsAfterLastDummyPacket“ aus der „haw_user“-Struktur um 1.

```
1 uint16_t secondsAfterLastDummyPacket;
```

Sollte dieser Wert den Wert der Konstante

```
1 #define MAX_TIME_TO_NEXT_DUMMY_PACKET 15 // in seconds
```

übersteigen, so wird an den jeweiligen User eine Dummy-Nachricht versendet. Durch Ausprobieren wurde ein Wert von 15 Sekunden ermittelt. Alle getesteten Bluetooth-Geräte unterhielten die Verbindung mindestens 15 Sekunden lang. Dieser Wert muss gegebenenfalls angepasst werden.

6.6.4 Die MasterManagement-Komponente

Die MasterManagement-Koponente erlaubt es, einen Master-User aufzunehmen. Es ist lediglich eine Funktion in dieser Komponente implementiert:

```
1 void addMasterUser(bd_addr_t bdAddr) ;
```

Auf den ersten Blick scheint es unangemessen, ein eigene Komponente mit lediglich einer Funktion zu realisieren. Die Verfügbarkeit dieser Komponente soll jedoch eventuelle Erweiterungen des Zugangskontrollsystems erleichtern. Sollte es nötig sein, mehr als einen Master zur gleichen Zeit zu unterstützen, kann die Verwaltung der Master-User in der MasterManagement-Komponente realisiert werden.

6.6.5 Die AlarmManagement-Komponente

Diese Komponente ermöglicht die Steuerung der Alarmanlage. Sie verfügt über folgende Funktionen:

```
1 void alarmInit(void);
2 void activateAlarm(void);
3 void deactivateAlarm(void);
```

Um die Alarmanlage von „STM32F4-Discovery-Board“ ein - bzw. auszuschalten musste die Alarmanlage umgebaut werden. Im Originalzustand konnte die Alarmanlage durch eine entsprechende Infrarotfernbedienung aktiviert und deaktiviert werden.

Um die Alarmanlage an das Development-Board anschließen zu können, wurde der Infrarotsensor der Alarmanlage ausgebaut. Die Schaltung der Alarmanlage wurde angepasst, und mit Hilfe von zwei Kabeln mit einem General Purpose Input/Output (GPIO)-Pin und der Masse des „STM32F4-Discovery-Boards“ verbunden. Die „alarmInit()“ - Funktion initialisiert den benötigten GPIO-Pin und muss nur einmal aufgerufen werden.

6.6.6 Die MessageManagement-Komponente

Die MessageManagement-Komponente implementiert das Protokoll zur Kommunikation mit dem Master. Es handelt sich hierbei um ein frei erfundenes Protokoll. Die Details zu diesem Protokoll können der message_management.c entnommen werden. Sowohl die Funktionen zum Versand der Nachrichten zum Master, als auch der Handler der RFCOMM-Nachrichten vom Master sind in dieser Komponente realisiert.

Die Nachrichten bestehen aus Strings, welche ausgewertet werden müssen. Eine einzelne Nachricht besteht aus mehreren Strings, welche von eine Separator-String getrennt werden. Dieser Separator-String wird durch die Konstante

```
1 HAW_RFCOMM_MESSGAE_CONTENT_SEPARATOR
```

definiert. Der erste Teilstring bezeichnet stets den Typ der Nachricht, und stellt eine Zahl dar.

Folgende Nachrichtentypen sind dabei definiert:

```
1 #define HAW_RFCOMM_MSG_USER_ACCEPT_REQUEST 0x0000
2 #define HAW_RFCOMM_MSG_POSITIVE_USER_ACCEPT_RESPONSE 0x0001
3 #define HAW_RFCOMM_MSG_NEGATIVE_USER_ACCEPT_RESPONSE 0x0002
4 #define HAW_RFCOMM_MSG_USER_NAME_REQUEST 0x0003
5 #define HAW_RFCOMM_MSG_USER_NAME_RESPONSE 0x0004
6 #define HAW_RFCOMM_MSG_USER_LIST_REQUEST 0x0005
```

```
7 #define HAW_RFCOMM_MSG_USER_LIST_START_RESPONSE 0x0006
8 #define HAW_RFCOMM_MSG_USER_LIST_END_RESPONSE 0x0007
9 #define HAW_RFCOMM_MSG_NEXT_USER_REQUEST 0x0008
10 #define HAW_RFCOMM_MSG_NEXT_USER_RESPONSE 0x0009
11 #define HAW_RFCOMM_USER_REMOVE_REQUEST 0x000A
12 #define HAW_RFCOMM_USER_REMOVE_POSITIVE_RESPONSE 0x000B
13 #define HAW_RFCOMM_USER_REMOVE_NEGATIVE_RESPONSE 0x000C
```

6.6.7 Die Benutzerüberwachung

Die Benutzerüberwachung wurde nicht als eine eigenständige Komponente realisiert. Die Funktion der Überwachung ist in die Abläufe des Bluetooth-Servers integriert. Die Überwachung der bestehenden Verbindungen ist bereits im Bluetooth-Protokoll implementiert. Im Bluetooth-Protokoll ist ein „Disconnection-Complete-Event“ vorgesehen. Dieser HCI-Event tritt nach einem Timeout auf und signalisiert, dass eine bestehende Verbindung unterbrochen wurde. Somit tritt dieses HCI-Event auch dann auf, wenn sich der Kommunikationspartner aus dem Bluetooth-Bereich entfernt hat.

6.6.7.1 Das Auffinden neuer Benutzer im Bluetooth-Bereich

Sobald ein Master bestimmt wurde, fängt der Bluetooth-Server die Suche nach Benutzern im Bluetooth bereich an. Zur Suche wird aber nicht das Inquiry-Scan-Verfahren eingesetzt. Das Problem ist, dass Android und andere mobile Betriebssysteme die Aktivierung der Bluetooth-Sichtbarkeit eines Gerätes nur für eine kurze Zeit erlauben. Dies geschieht vor allem aus Sicherheitsgründen, aber senkt ebenfalls den Stromverbrauch. So startet Android einen Timer, sobald die Sichtbarkeit des Bluetooth-Gerätes aktiviert wurde. Nach dem Ablauf dieses Timers, wird die Sichtbarkeit wieder deaktiviert. Die dauerhafte Aktivierung der Bluetooth-Sichtbarkeit ist nur mit zusätzlicher Software möglich.

Um einen neuen Benutzer im Bluetooth-Bereich zu detektieren, wird der Reihe nach zu jedem Benutzer eine Verbindung aufgebaut. Sollte der Benutzer nicht auf die Verbindungsanfrage antworten, wird der Verbindungsversuch zum nächsten Benutzer gestartet. Sollte ein Benutzer die Verbindung akzeptieren, so wird der Link Key überprüft. Sollte es sich um einen authentifizierten Nutzer handeln, so wird die Alarmanlage deaktiviert und die Suche unterbrochen. Die Präsenz auch nur eines Users reicht aus, um das Alarmsystem abzustellen.

6.6.7.2 Die Überwachung eines erkannten Benutzers

Zur Überwachung eines Benutzers im Bluetooth-Bereich horcht der Server auf einen „Disconnection-Complete“ HCI-Event und erkennt mit Hilfe der Benutzerverwaltungskomponente, zu welchem Benutzer die Verbindung nicht mehr besteht. Sobald die Verbindung nicht mehr besteht wird sofort die Alarmanlage aktiviert und die Suche nach dem nächsten Benutzer zu starten.

6.6.7.3 Das Problem der parallelen Benutzerüberwachung

Es bietet sich an, alle Nutzer im Bluetooth-Bereich zu erkennen und zu jedem einzelnen eine Verbindung aufzubauen. In diesem Fall muss die Suche nach einem neuen Benutzer nur dann ausgeführt werden, wenn alle zuvor erkannten User den Bluetooth-Bereich verlassen haben.

Dies würde bedeuten, dass bei N Benutzern auch N Bluetooth-Verbindungen vom Bluetooth-Gerät des Bluetooth-Servers parallel unterhalten werden müssten. Nach der Bluetooth-Spezifikation kann ein Piconet maximal 255 Teilnehmer enthalten. Dabei ist jedoch zu beachten, dass nur 8 Teilnehmer gleichzeitig aktiv sein können und alle anderen Teilnehmer müssen sich im Park-Zustand befinden. [BTSPEZ21 (2007)]

Es wäre natürlich möglich, mehrere Bluetooth-Server mit mehreren Bluetooth-Dongles zu betreiben um dieses Problem zu umgehen. Je nach Anzahl der Benutzer müsste das System, um weitere Server erweitert werden. In dieser Arbeit wird jedoch die Ein-Server-Lösung realisiert. Um das Problem zu umgehen, werden in dieser Realisierung maximal 2 Verbindungen parallel unterhalten. Bei einer Authentifizierungsanfrage wird eine Verbindung zum neuen Benutzer und eine zum Master unterhalten.

6.6.7.4 Die Optimierung der Benutzerüberwachung

Bei einer großen Anzahl von Benutzern dauert es einige Zeit, bis ein Benutzer im Bluetooth-Bereich gefunden wurde. Es muss jedes Mal auf eine Antwort von Benutzern gewartet werden. Sollte bei einer Verbindungsanfrage keine Antwort kommen, so signalisiert der „PAGE TIMEOUT“, dass keine Verbindung aufgebaut werden kann.

Definition 24 (*Page Timeout*)

> *„The Page_Timeout configuration parameter defines the maximum time the local Link Manager will wait for a baseband page response from the remote device at a locally initiated connection attempt. If this time expires and the remote device has not responded to the page at baseband level, the connec- tion attempt will be considered to have failed.“*
>
> *Quelle: [BTSPEZ21 (2007)], Chapter 6.6 PAGE TIMEOUT*

Die Antwortzeit des Benutzers im Bluetooth-Bereich auf eine Verbindungsanfrage kann unterschiedlich lange dauern. Um diesen Prozess zu beschleunigen, muss der „Page Timeout“ so kurz wie möglich eingestellt werden. Die Timeout-Zeit erlaubt eine Einstellung bis auf den Wert von 0.625 msec. Aber sollte der „Page Timeout“ zu kurz gewählt werden, so wird die Antwort vom Benutzer nie ankommen, da die Nachricht vom Benutzer zum Master auch im Best-Case eine bestimmte Zeit braucht. Bei der Durchführung der Tests mit unterschiedlichen Geräten, hat sich eine Zeit von 0.4 bis 3 Sekunden für den „Page Timeout“ bewährt. Der Timeout muss je nach eingesetzten Bluetooth-Geräten angepasst werden.

6.7 Die Softwarekomponenten der auf Android basierenden Benutzerverwaltungssoftware

6.7.1 Die BluetoothLocalService-Komponente

Dieser Service bildet die Schnittstelle zwischen der ConnectionManager-Komponente und der Activity-Komponenten. Dieser Service kann von jeder Komponente der Benutzerverwaltungssoftware aufgerufen werden. Durch die Funktion

```
1 public static void connectToServer(Activity activitiy)
```

haben die Activity-Komponenten die Möglichkeit, sich mit dem Bluetooth-Server zu verbinden. Die Funktion

```
1 public static String createMessage(MessageType messageType,
2 List<String> paramList)
```

erlaubt die Generierung einer Nachricht an den Server. Durch die Funktion

```
1 public static void sendMessage(Activity activitiy, String message)
```

können Nachrichten dann an den Bluetooth-Server versendet werden.

6.7.2 Die ConnectionManager-Komponente

Die ConnectionManager-Komponente übernimmt die Verwaltung der Verbindungen. Vor allem steuert diese Komponente Threads, welche für die Bluetooth-Kommunikation notwendig sind. Durch die Benutzung dieser Komponente ist es der BluetoothLocalService-Komponente möglich, die Verbindung zu Bluetooth-Server aufzubauen, zu unterhalten und wieder schließen. Auch das Versenden von Nachrichten an dem Bluetooth-Server wird durch dieser Komponente ermöglicht.

6.7.3 Die UserLocalService-Komponente

Diese UserLocalService-Komponente verwaltet ein Set von Benutzern.

```
1 private static Set<HawUser> userSet = new TreeSet<HawUser>();
```

Das Modell HawUser enthält die Benutzerinformationen, welche später grafisch dargestellt werden sollen.

```
1  package com.btprojectapp.model;
2  public class HawUser implements Comparable<HawUser>{
3          private String name;
4          private String adress;
5          private boolean connected;
6
7          public HawUser(String name,
8                                  String adress,
9                                  boolean connected){
10                 super();
11                 this.name = name;
12                 this.adress = adress;
13                 this.connected = connected;
14         }
15
16         public String getName() {
17                 return name;
18         }
19
20         public void setName(String name) {
21                 this.name = name;
22         }
23
24         public String getAdress() {
25                 return adress;
26         }
27
28         public void setAdress(String adress) {
29                 this.adress = adress;
30         }
31
32         public boolean isConnected() {
33                 return connected;
```

```
34          }
35
36          public void setConnected(boolean connected) {
37                  this.connected = connected;
38          }
39
40          public boolean equals(HawUser user){
41                  if(this.name.equals(user.name)){
42                          return true;
43                  }else{
44                          return false;
45                  }//if
46          }
47
48          public int compareTo(HawUser user){
49                  return this.name.compareTo(user.getName());
50          }
51  }//class
```

Ein Benutzer kann jederzeit zu diesem Set hinzugefügt oder wieder gelöscht werden. Dieses Set von Benutzern wird in der UserListActivity-Komponente grafisch dargestellt.

6.7.4 Die MessageHandler-Komponente

Wie der Name schon sagt, enthält die MessageHandler-Komponente einen MessageHandler für die Nachrichten vom Bluetooth-Server. Als Reaktion auf eine Nachricht, wird entweder eine Antwort versendet, oder in eine andere Activity-Komponente gewechselt.

6.7.5 Die MainActivity-Komponente

Das Layout und die Funktionalität der Buttons gleich nach dem Start des Benutzerverwaltungssoftware ist in der MainActivity-Komponente beschreieben.

6.7.6 Die AcceptUserActivity-Komponente

Das Layout und die Funktionalität der Buttons gleich nach dem Erhalt einer HAW_RFCOMM _MSG_USER_ACCEPT_REQUEST -Nachricht, ist in der AcceptUserActivity-Komponente beschrieben.

6.7.7 Die SetUserNameActivity-Komponente

Das Layout und die Funktionalität der Buttons und Textfelder gleich nach dem Erhalt einer HAW_RFCOMM_ MSG_USER_NAME_REQUEST -Nachricht, ist in der SetUserNameActivity-Komponente beschrieben.

6.7.8 Die RemoveUserActivity-Komponente

Das Layout und die Funktionalität der Buttons gleich nach dem Erhalt einer HAW_RFCOMM _USER_REMOVE_REQUEST -Nachricht, ist in der RemoveUserActivity-Komponente beschrieben.

6.7.9 Die UserListActivity-Komponente

Das Layout und die Funktionalität der Buttons gleich nach dem Erhalt einer HAW_RFCOMM _MSG_USER_LIST_REQUEST -Nachricht, ist in der UserListActivity-Komponente beschrieben. Im Grunde stellt die UserListActivity-Komponente die Liste der Benutzer grafisch dar.

6.8 Die Realisierung des Sicherheitskonzeptes

6.8.1 Der Aufbau der Bluetooth-Verbindung

Um ausreichende Sicherheit zu gewährleisten, wird für alle Bluetooth-Verbindungen das Secure Simple Pairing (SSP)-Verfahren eingesetzt. Daher können sich nur Bluetooth-Geräte ab der Version 2.1 mit dem System verbinden. Sollte ein Bluetooth-Gerät eine Verbindung starten, welches SSP nicht unterstützt, so schließt der Bluetooth-Server sofort die Verbindung. Das Pairing wird nicht ausgeführt, da in diesem Fall das Link Manager Protocol (LMP)-Pairing eingesetzt werden müsste, welches nicht genügend Sicherheit liefert.

Um mögliche Man In The Middle (MITM)-Angriffe zu unterbinden, wird stets das „Numeric Comparison“ Assoziationsmodell verwendet. Das Zugangskontrollsystem erlaubt dabei nicht, ein anderes Modell zu wählen. Sollte es zu einer Aufforderung kommen, ein anderes Assoziationsmodell zu verwenden, schließt der Bluetooth-Server sofort die Verbindung. Das Pairing wird in diesem Fall nicht ausgeführt. Der Angreifer erhält nicht die Möglichkeit, einen MITM-Angriff zu starten.

Das Development-Board, auf welchem das Zugangskontrollsystem läuft, verfügt weder über Eingabe - noch über Ausgabemöglichkeiten. Dennoch versendet das System bei dem „IO-Capability-Austausch“ ein Paket, welches dem Partner signalisiert, dass Ein- und Ausgabeschnittstellen

verfügbar sind. Die „Numeric Comparison“ wird später an den Master weitergeleitet. Da die Benutzer sich über ihre Mobilfunktelefone mit dem Bluetooth-Server verbinden, signalisieren diese ebenfalls, dass Ein- und Ausgabeschnittstellen verfügbar sind.

Sollte ein User ein „IO-Capability“-Paket versenden, welches nicht die Eigenschaft „DisplayYesNo“ enthält, wird die Verbindung geschlossen, und das Pairing wird nicht ausgeführt. Der Grund dafür ist, dass das „Numeric Comparison“ Assoziationsmodell nur dann ausgewählt wird, wenn beide Partner die „DisplayYesNo“ signalisieren. Es wird zwar auch dann ausgewählt, wenn einer der Parteien nicht über Ein - oder Ausgabemöglichkeiten verfügt, aber in diesem Fall wird die Bestätigung der Comparison-Number nur von einem der Partner verlangt. Dies würde einen MITM-Angriff möglich machen. Die aufgebaute Verbindung wird stets verschlüsselt. Sollte der Partner der Verschlüsselung nicht zustimmen, wird die Verbindung sofort unterbrochen.

6.8.2 Weiterleitung der Verbindungsbestätigung an den Master

Da das Development-Board nicht über die nötigen Ein - und Ausgabeschnittstellen verfügt, wird die „User-Confirmation-Request“ - Nachricht an den Master weitergeleitet. Dazu muss die Verwaltungssoftware auf seinem Smartphone aktiviert und mit dem Zugangskontrollsystem verbunden sein. Sollte das der Fall sein, so muss das Pairing unterbrochen und die Verbindung geschlossen werden.

Im Erfolgsfall wird diese Nachricht über das Radio Frequency Communication (RFCOMM)-Protokoll weiter an den Master geleitet. Die 6-Stellige Comparison-Number wird dann dem Master angezeigt. Der Administrator hat nun die Möglichkeit, diese mit der Comparison-Number des neuen Nutzers zu vergleichen und gegebenenfalls zu bestätigen oder abzulehnen. Die Antwort an das Zugangskontrollsystem wird ebenfalls über das RFCOMM- Protokoll versendet. Je nach Entscheidung des Masters, wird das Pairing zwischen dem User und dem System entweder zu Ende geführt, oder abgebrochen.

6.8.3 Die Erkennung eines Users

Nach dem Pairing, wird für jeden Benutzer ein eindeutiger Name, die Bluetooth Address (BDADDR) und der Link Key abgespeichert. Der Benutzer wird anhand seiner Bluetooth-Adresse und des Link Keys erkannt. Sollte das Link Key nicht zu der abgespeicherten Adresse passen, wird der User sofort vom Zugangskontrollsystem gelöscht. In diesem Fall muss er sich erneut authentifizieren.

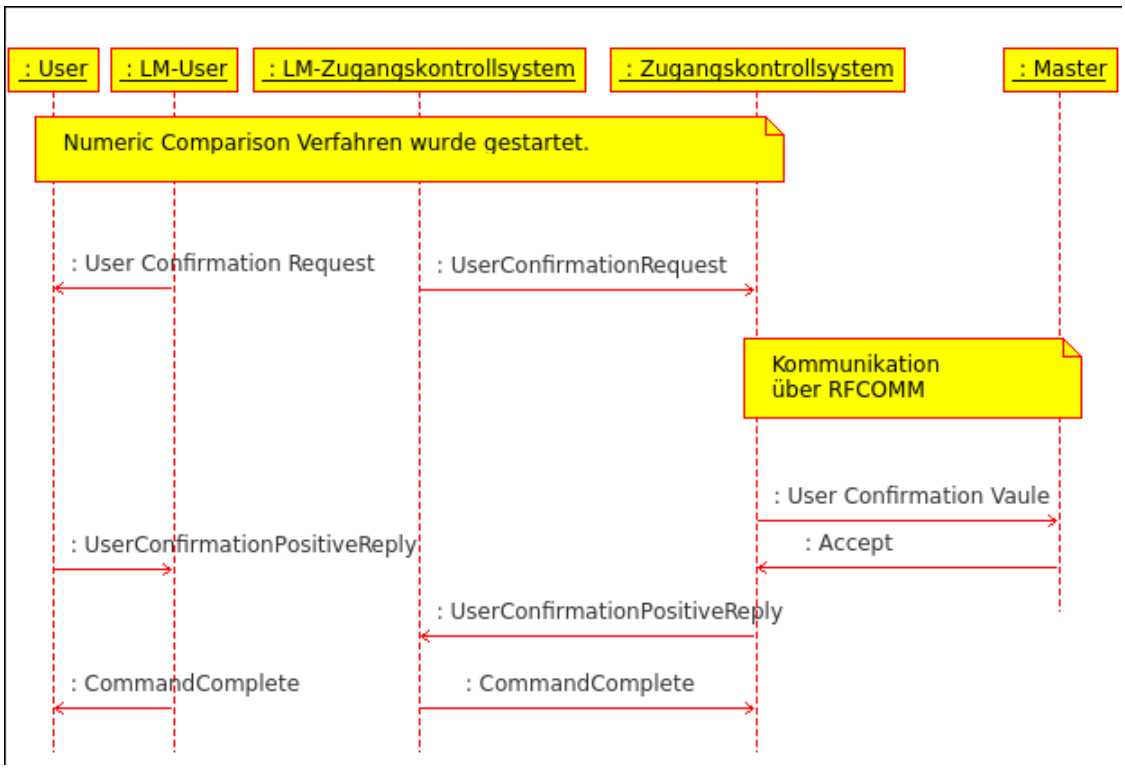

Abbildung 6.6: Das Sequenzdiagramm veranschaulicht den Ablauf eines erfolgreichen Pairing-Prozesses zwischen einem neuem User und dem Zugangskontrollsystem. Die Comparison Number wird an dem Master weitergeleitet, damit diese dargestellt und abgeglichen werden kann.

7 Bewertung und Ausblick

7.1 Funktion

7.1.1 Die Festlegung des Masters

Sobald die Initialisierung des auf dem STM32F4-Board laufenden Bluetooth-Servers erfolgt ist, kann ein Master festgelegt werden. Das Bluetooth-Gerät, welches als erstes erfolgreich eine sichere Verbindung zum Bluetooth-Server aufbaut, wird als Master festgelegt. Danach wird sofort die Benutzerüberwachung aktiviert. Dabei wird zunächst nur das Master-Gerät überwacht. Der Master wird vom Benutzerüberwachungssystem wie ein normaler Benutzer behandelt. Sollte der Master sich aus dem Bluetooth-Bereich entfernen oder nicht mehr erreichbar sein, wird sofort die Alarmanlage aktiviert.

7.1.2 Das hinzufügen eines neuen Benutzers

Eine Android-App erlaubt es dem Master, sich mit dem auf dem STM32F4-Board laufendem Bluetooth-Server zu verbinden. Sobald nun ein nicht authentifiziertes Gerät den Versuch startet, sich mit dem Bluetooth-Server zu verbinden, wird das Secure Simple Pairing (SSP) gestartet. Sollte das nicht authentifizierte Gerät SSP nicht unterstützen, wird die Verbindung sofort geschlossen. Der Bluetooth-Server sendet die Comparison-Number verschlüsselt an den Master, damit diese von der Android-App zum Abgleich angezeigt werden kann. Um ein neues, bisher nicht authentifiziertes Gerät als einen Benutzer aufzunehmen, muss der Master die Comparison-Number abgleichen, und durch den Druck des „Accept“-Buttons bestätigen. Für den Abgleich hat der Master dabei 29 Sekunden Zeit. Dies ist die maximale Zeit, welche der durch die Bluetooth-Spezifikation definierte „Connection-Accept-Timeout“ erlaubt.

Sollte das SSP-Verfahren erfolgreich ablaufen, wird der neue Benutzer in die Benutzerliste des Benutzerüberwachungssystems aufgenommen. Nach dem der Master die Comparison-Number akzeptiert hat, kann er im nächsten Menü den Namen des neuen Benutzers festlegen.

7.1.3 Die Benutzerüberwachung

Sowohl der Master als auch alle Benutzer werden vom auf dem STM32F-Board laufendem Bluetooth-Server sofort nach der Aufnahme in die Benutzerliste überwacht. Sollte sich ein Benutzer aus dem Bluetooth-Bereich entfernen oder nicht mehr erreichbar sein, wird sofort der Alarm aktiviert und die Suche nach Benutzern in Reichweite gestartet. Bei jedem Verbindungsaufbau zu einem Benutzer wird stets eine Authentifizierung verlangt. Sollte die Authentifizierung erfolgreich, der „Link-Supervision-Timeout“ erfolgreich eingestellt und die Verbindung erfolgreich verschlüsselt sein, wird der Benutzer als solcher erkannt. Sobald ein Benutzer erkannt worden ist, wird der Alarm deaktiviert. Sollte eine dieser Bedingungen nicht erfüllt werden, wird die Verbindung sofort geschlossen. Der „Link-Supervision-Timeout“ dient zur Überwachung der Bluetooth-Verbindung. Dieser Timeout bestimmt die Zeit bis zu einem „Host Controller Interface (HCI)-Disconnection-Event“, nach dem eine Bluetooth-Verbindung nicht mehr besteht. Anhand des Disconnection-Events erkennt der Bluetooth-Server, dass der Benutzer nicht mehr in Reichweite ist.

7.2 Einschränkungen

Dadurch, dass gleichzeitig nur zu einem Benutzer (den Masters ausgenommen) eine Verbindung besteht, ist die Anzahl der Benutzer nur durch die Speicherkapazität des Benutzerverwaltungssystems beschränkt. Je mehr Benutzer jedoch registriert sind, desto mehr Zeit wird benötigt, um einen Benutzer in Bluetooth-Reichweite zu finden. Da die Bluetooth-Adresse nicht zur Identifikation ausreicht, muss für jeden Benutzer ein Verbindungsversuch gestartet werden. Es nimmt jedoch einige Zeit in Anspruch um festzustellen, dass ein Benutzer nicht in Reichweite ist. Es sind natürlich auch andere Lösungen dieses Problems denkbar, zum Beispiel durch parallele Überwachung mehrerer Benutzer in Bluetooth-Reichweite.

Eine weitere Einschränkung ergibt sich aus der Bluetooth-Spezifikation. Um den „Link-Supervision-Timeout“ einzustellen, muss der Bluetooth-Server als Piconet-Master fungieren. Daher können nur Geräte überwacht werden, welche die „Slave“-Rolle im Piconet einnehmen.

Ich habe im Rahmen dieser Arbeit keine Möglichkeit gefunden, eine Bluetooth-Verbindung zwischen einer auf dem „Huawei Ideos X3“ installierten Android-App und auf dem STM32F4-Bard laufenden Bluetooth-Server stets zuverlässig zu unterhalten. Nach dem die Verbindung ein bis vier Mal erfolgreich auf- und abgebaut wurde, lässt sich eine weitere Verbindung nicht mehr aufbauen. Bei einem erfolglosen Verbindungsaufbau tritt eine „Service-Discovery-Failed-Exception“ auf. Das Problem scheint hardwareabhängig zu sein, und ist nur durch „Workarounds“

lösbar. Dieses Problem wurde hier nicht gelöst, da dies den Rahmen meiner Arbeit sprengen würde.

Durch das System können zur Zeit 10 Benutzer mit einem Namen von bis zu 7 Zeichen verwaltet werden. Um längere Namen zu unterstützen, kann der Name zum Beispiel in einem einzelnen Radio Frequency Communication (RFCOMM)-Packet versendet werden. Alternativ kann auch eine Unterstützung von Fragmentierten RFCOMM-Packeten implementiert werden.

7.3 Sicherheit

Sollte ein Benutzer ein registriertes Bluetooth-Gerät verlieren, muss der Master dieses Gerät durch die Benutzerverwaltungssoftware sofort löschen. Sollte das nicht geschehen, so ist das Zugangskontrollsystem nicht mehr sicher. Unbefugte könnten sich in diesem Fall mit einem gestohlenen, am Gesamtsystem registrierten Bluetooth-Gerät Zutritt verschaffen. Sollte das Master-Bluetooth-Gerät auch nur kurzzeitig in die Hände von Unbefugten gelangen, so ist es empfehlenswert, einen „Reset“ des Gesamtsystems durchzuführen. Dabei werden alle bestehenden Benutzer gelöscht. Jeder einzelne Benutzer müsste dann vom Master erneut am Gesamtsystem registriert werden. Der Grund für dieses Vorgehen ist, dass die Benutzerverwaltungssoftware die Benutzer unter frei wählbaren Namen abspeichert. Durch eine Manipulation könnte ein Benutzer mit dem gleichen Namen gespeichert sein, aber der Link Key kann nicht mehr abgeglichen werden. Der Abgleich würde von allen Benutzern verlangen, dass eine spezielle Software installiert ist, welche die gespeicherten Link Keys anzeigen könnte.

7.4 Ausblick

Bei der beschriebenen Realisierung des Gesamtsystems handelt es sich um einen Prototypen, welcher die Machbarkeit der Realisierung nachweist. Dieser Prototyp bietet eine Grundlage für ein fertiges Produkt, erfordert jedoch Verbesserungen und Erweiterung um weitere Funktionen.

Der Prototyp unterstützt nur einen Master. In der Praxis wird jedoch die Verwaltung der Benutzer nicht nur von einem Administrator ausgeführt. Um mehrere Master gleichzeitig zu unterstützen, müsste das Benutzerverwaltungssystem erweitert werden. Weiterhin ist empfehlenswert, die Benutzerverwaltungssoftware auch für andere Betriebssysteme zu implementieren. Die Benutzerverwaltung sollte von Anwendungen auf Basis von Betriebssystemen wie zum Beispiel „Apple iOS“, „Linux“, oder „Windows“ möglich sein. Die Kommunikation zwischen dem Master und dem Zugangskontrollsystem sollte sich nicht auf die Bluetooth-Reichweite beschränken. Der Master sollte zum Beispiel sofort informiert werden, falls ein Alarm ausgelöst wurde, auch wenn

er sich nicht gerade im Bluetooth-Bereich befindet. Auch das Abstellen der Alarmanlage sollte außerhalb der Bluetooth-Reichweite möglich sein. Die Daten der Benutzer werden im Random Access Memory (RAM) abgespeichert. Damit die Liste der Benutzer auch nach einem Reset wieder verfügbar ist, sollte sie im Flash-Speicher abgelegt werden.

Die Realisierung des Prototyps erlaubt die gleichzeitige Überwachung nur eines Benutzers im Bluetooth-Bereich. Sobald dieser den Bereich verlässt, wird die Suche nach dem nächsten Benutzer in Reichweite gestartet. Diese führt zur einer zeitlichen Verzögerung. Je größer die Zahl der gespeicherten Benutzer, desto größer ist die Zeit bis zur Erkennung eines Nutzers im Bluetooth-Bereich. Um eine Vielzahl von Benutzern unterstützen zu können, müssten mehrere Benutzer im Bluetooth-Bereich parallel überwacht werden. Dies würde die Reaktionszeit des Zugangskontrollsystems steigern.

Die meisten Mobilfunktelefone schließen die Logical Link Control and Adaptation Layer Protocol (L2CAP)-Verbindung nach einem Timeout, falls kein Datenverkehr besteht. Eine Software welche die Timeouts der einzelnen Benutzer erkennt und zum richtigen Zeitpunkt ein Dummy-Paket an einen bestimmten Benutzer versendet, würde den Stromverbrauch der überwachten Geräte senken, da weniger Dummy-Pakete bearbeitet werden müssten.

8 Zusammenfassung

In dieser Arbeit wurde ein Mikrocontroller-gesteuertes, Bluetooth-basiertes Zugangskontrollsystem mit einer auf Android aufbauenden Verwaltungssoftware entworfen und die Realisierbarkeit mit Hilfe eines einfachen Prototyps nachgewiesen. Strukturell gesehen, besteht das Konzept aus zwei Bestandteilen. Aus dem Zugangskontrollsystem und dem Benutzerverwaltungssystem. Das Zugangskontrollsystem ist für die Überwachung und Identifizierung der Benutzer zuständig. Weiterhin steuert das Zugangskontrollsystem eine Alarmanlage, um diese bei Notwendigkeit aktivieren bzw. deaktivieren. Das Benutzerverwaltungssystem ist für die Abspeicherung und Verwaltung aller für die überwachten Benutzer relevanten Daten zuständig. Das Benutzerverwaltungssystem bietet eine Android basierte Software zur Verwaltung der Benutzer durch den Master, wecher die Rolle des Administrators einnimmt. Das Zugangskontrollsystem und das Benutzerverwaltungssystem bilden zusammen das Gesamtsystem.

Das Gesamtsystem macht es möglich, Benutzer anhand der Bluetooth-Chips ihrer Bluetooth-Geräte sicher zu identifizieren und zu überwachen. Eine sichere Identifizierung ist nur durch den Einsatz von Bluetooth-Geräten, welche das Secure Simple Pairing (SSP)-Verfahren unterstützen, möglich. Das sind Geräte, welche die Bluetooth-Spezifikation in der Version 2.1, oder einer höheren Version unterstützen.

Allein der Einsatz SSP-Verfahrens reicht jedoch nicht aus, um die Sicherheit zu gewährleisten. Die Auswahl richtiger Assoziationsmodelle so wie die fehlerlose Implementierung des Zugangskontrollsystems sind notwendig, um eine sichere Zugangskontrolle zu realisieren. Dabei ist es auch notwendig, dass der Master bei der Registrierung eines neuen Benutzers stets die Comparison-Number genau abgleicht, damit kein Man In The Middle (MITM)-Angriff durchgeführt werden kann.

Die Reichweite, in der die Benutzerüberwachung stattfinden kann, ist sehr von den eingesetzten Bluetooth-Geräten abhängig. Da aber Bluetooth-Geräte der Klasse 3 (der niedrigsten Klasse) im freien Feld eine Reichweite von ungefähr 10 Metern erreichen, kann diese als minimale Reichweite angenommen werden. Bei der Aufstellung des Zugangskontrollsystems ist darauf zu achten, dass die Reichweite der verwendeten Bluetooth-Geräte groß genug ist, um die Alarmanlage rechtzeitig deaktivieren zu können.

Da das Zugangskontrollsystem für eine Installation an einem überwachten Objekt möglichst klein und kostengünstig sein sollte, wurde als Basis für den Prototyp das „STM32F4-Discovery-Board“ (bestückt mit einem ARM Cortex-M4F- Mikrocontroller) in Verbindung mit einem Bluetooth-Dongle eingesetzt. Die Frameworks „BT-Stack“ [Ringwald (2012)], „Uubt“ [Belostotsky (2012)] sowie die „Android Bluetooth API“ [ANDROIDAPI (2013)] wurden als Grundlage für die Bluetooth-Kommunikation eingesetzt. Die im Prototyp vorgestellte Benutzerverwaltungssoftware ist als eine Android-App realisiert und bietet eine grafische Benutzeroberfläche. Durch die sehr kostengünstigen und gleichzeitig auch sehr leistungsstarken, auf dem Markt verfügbaren Mikrocontroller, war eine sehr preiswerte Realisierung des vorgestellten Konzeptes möglich. Die Gesamtkosten für die verwendete Hardware lagen unter 50 Euro.

Glossar

AU_RAND

Eine 128 Bit lange Zufallszahl.

Benutzer

Ein Benutzer bzw. User ist ein Bluetooth-Gerät welches im Zugangskontrollsystem gespeichert ist. Sobald das Benutzergerät sich im Bluetooth-Bereich des Zugangskontrollsystems befindet, wird es vom Zugangskontrollsystem als solches erkannt und angemeldet.

Benutzerverwaltungssoftware

Die Benutzerverwaltungssoftware ist eine der Komponenten des Benutzerverwaltungssystems. Diese Software läuft jedoch nicht auf dem Bluetooth-Server wie alle anderen Software-Komponenten des Gesamtsystems, sondern auf dem Smartphone des Masters. Dieses Software erlaubt eine grafische Steuerung der Benutzerverwaltung durch den Master.

Benutzerverwaltungssystem

Das Benutzerverwaltungssystem (verkürzt auch Verwaltungssystem) ist für die Verwaltung und Speicherung von authentifizierten Benutzern verantwortlich. Das Benutzerverwaltungssystem besteht aus einer Menge von Soft- und Hardware-Komponenten.

Bluetooth Baseband

„The part of the Bluetooth system that specifies or implements the medium access and physical layer pro- cedures to support the exchange of real-time voice, data information streams, and ad hoc networking between Bluetooth Devices."
[BTSPEZ21 (2007)]

Bluetooth Radio

Die physikalische Realisierung des Bluetooth-Standards

Bluetooth-Controller

„A sub-system containing the Bluetooth RF, baseband, resource controller, link manager,

device manager and a Bluetooth HCI."
[BTSPEZ21 (2007)]

Bluetooth-Server

Der Bluetooth-Server ist sowohl eine Komponente des Zugangskontrollsystems, als auch eine Komponente des Benutzerverwaltungssystems. Das ganze Zugangskontrollsystem und ein Teil des Benutzerverwaltungssystems laufen auf einem Bluetooth-Server. Der Bluetooth-Server kontrolliert ob authentifizierte Nutzer in der Nähe sind, und aktiviert falls notwendig die Alarmanlage. Alle Software-Komponenten des Gesamtsystems mit Ausnahme der Benutzerverwaltungssoftware werden auf dem Bluetooth-Server a

Comparison-Number

Eine 6-Stellige Nummer von 000000 bis 999999 welche beim Numeric-Comparison-Verfahren eingesetzt wird.

Ecplise

Eine Software basierte Entwicklungsumgebung zur Entwicklung von Software verschiedenster Art.

Elliptic Curve Diffie Hellman-Verfahren

„Das Elliptic Curve Diffie Hellman-Verfahren (ECDH) setzt auf dem Verschlüsselungsalgorithmus des Diffie-Hellman-Verfahrens (DH) auf, das auf der Potenzierung der zu verschlüsselnden Daten mit großen Exponenten basiert. Das Verfahren, dessen Grundlage ECC bildet, bietet hohe Sicherheit bei relativ kurzen Schlüssellängen."
[ITWISSEN2012]

Encryption Key

Ein Schlüssel, welcher zur Verschlüsselung einer Bluetooth-Verbindung dient.

f1

Funktion f1 aus dem Kapitel 7.7.1 der Bluetooth-Spezifikation [BTSPEZ21 (2007)].

Gesamtsystem

Das Gesamtsystem besteht aus dem Zugangskontrollsystem und dem Benutzerverwaltungssystem. Das Gesamtsystem beschreibt die Menge aller in dieser Arbeit vorgestellten Soft- und Hardware-Komponenten.

HCI-Befehl

Ein HCI-Befehl ist eine Nachricht, welche vom Bluetooth-Host an den Bluetooth-Controller versendet werden kann, um diesen zu steuern.
[BTSPEZ21 (2007)

HCI-Event

Ein HCI-Event ist eine asynchrone Nachricht vom Bluetooth-Controller zum Bluetooth-Host. Diese Nachrichten informieren den Host, über den aktuellen Status des Bluetooth-Controllers.
[BTSPEZ21 (2007)

HCI-Event-Packet

„The HCI Event Packet is used by the Controller to notify the Host when events occur."
[BTSPEZ21 (2007)]

Host Controller Interface

„The Bluetooth Host Controller Interface (HCI) provides a command interface to the baseband controller and link manager and access to hardware status and con- trol registers. This interface provides a uniform method of accessing the Bluetooth baseband capabilities."
[BTSPEZ21 (2007)]

IN_RAND

Eine 128 Bit lange Zufallszahl.

Inquiry-Scan-Verfahren

„A procedure where a Bluetooth device transmits inquiry messages and listens for responses in order to dis- cover the other Bluetooth devices that are within the coverage area."
[BTSPEZ21 (2007)]

L2CAP-Channel

„L2CAP provides a multiplexing role allowing many different applications to share the resources of an ACL-U logical link between two devices. Applications and service protocols interface with L2CAP using a channel-oriented interface to create connections to equivalent entities on other devices. L2CAP channel endpoints are identified to their clients by a Channel Identifier (CID). This is assigned by L2CAP, and each L2CAP channel endpoint on any device has a different CID."
[BTSPEZ21 (2007)]

Link Key

„A secret key that is known by two devices and is used in order to authenticate each device to the other."
[BTSPEZ21 (2007)]

LK_RAND

Eine 128 Bit lange Zufallszahl.

MAC-Adresse

MAC-Adresse (Media-Access-Control-Adresse) ist die Hardware-Adresse jedes einzelnen Bluetooth-Gerätes. Sie dient zur Identifikation in Netwerken, und sollte Weltweit einmalig sein. Sie ist Bestandteil der Sicherungsschicht des OSI-Modells.

Master

Ein Master ist ein User und wird auch wie ein User vom Zugangskontrollsystem behandelt. Weiterhin übernimmt der Master die Verwaltung des Zugangskontrollsystems. Der Master legt fest, welche Geräte als User im Zugangskontrollsystem gespeichert sind. Ohne die Zustimmung des Masters, kann kein neuer User im Zugangskontrollsystem aufgenommen werden. Der Master kann bereits bestehende User vom Zugangskontrollsystem löschen. Um Verwaltungsaufgaben übernehmen zu können, muss der Master über eine Verwaltungssoftware verfügen und als Master am Zugangskontrollsystem angemeldet sein.

Park-Zustand

„ When a slave does not need to participate on the piconet channel, but still needs to remain synchronized to the channel, it can enter PARK state. PARK state is a state with very little activity in the slave.
[BTSPEZ21 (2007)]"

Participant in Multiple Piconet

„A device that is con- currently a member of more than one piconet, which it achieves using time division multiplexing (TDM) to interleave its activity on each piconet physical channel."
[BTSPEZ21 (2007)]

PassKey

Eine 6-Stellige Nummer von 000000 bis 999999 welche beim Passkey-Entry-Verfahren eingesetzt wird.

Piconet

„A collection of devices occupying a shared physical channel where one of the devices is the Piconet Master and the remaining devices are connected to it.“
[BTSPEZ21 (2007)]

Private Key

Public Key bezeichnet den nach dem Elliptic Curve Diffie Hellman-Verfahren resultierenden private-Schlüssel aus dem ECDH-Public-Private-Schlüsselpaar.

Public Key

Public Key bezeichnet den nach dem Elliptic Curve Diffie Hellman-Verfahren resultierenden public-Schlüsssel aus dem ECDH-Public-Private-Schlüsselpaar.

Scatternet

„Two or more piconets that include one or more devices acting as PMPs.“
[BTSPEZ21 (2007)]

SRES

Eine 32 Bit lange Zahl, welche bei der Authentifizierung verglichen wird.

STMicroelectronics

Hersteller Firma des STM32F4-Discovery

User

Ein User bzw. Benutzer ist ein Bluetooth-Gerät, welches im Zugangskontrollsystem gespeichert ist. Sobald das Benutzergerät sich im Bluetooth-Bereich des Zugangskontrollsystems befindet, wird es vom Zugangskontrollsystem als solches erkannt und angemeldet.

Zugangskontrollsystem

Das Zugangskontrollsystem ist für die Erkennung und Überwachung von authentifizierten Benutzern zuständig. Weiterhin ist es für die Aktivierung einer Alarmanlage verantwortlich, um den Zugang zu einem geschützten Objekt (zum Beispiel einer Garage) für nicht authentifizierte Benutzer zu erschweren. Das Zugangskontrollsystem besteht aus einer Menge von Soft- und Hardware-Komponenten.

Abbildungsverzeichnis

Literaturverzeichnis

[RFCOMM 2003] *RFCOMM with TS 07.10 Serial Port Emulation*, 2003. – URL https://www.bluetooth.org/docman/handlers/DownloadDoc.ashx?doc_id=40909

[BTCHAT 2009] *BluetoothChat - Bluetooth Chat*, 2009. – URL http://jayxie.com/mirrors/android-sdk/resources/samples/BluetoothChat/index.html

[ELYSISBT 2011] *Secure Simple Pairing Explained*, 2011. – URL www.ellisys.com/technology/een_bt07.pdf

[ANDROIDAPI 2013] *Bluetooth*, 2013. – URL http://developer.android.com/guide/topics/connectivity/bluetooth.html

[ARM Limited 2005] ARM Limited (Veranst.): *RealViewÂ® Compilation Tools Compiler and Libraries Guide - 7.1.1. What is semihosting?* Version 2.2. 2005. – URL http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dui0205f/Bgbjjgij.html

[Belostotsky 2012] Belostotsky, Vitaly: *Uubt is for McU UsB BlueTooth*, 2012. – URL https://github.com/byly/uubt

[BITKOM 2012] BITKOM: *Jeder Dritte hat ein Smartphone.* April 2012. – URL http://www.bitkom.org/60376.aspx?url=BITKOM_Presseinfo_Besitz_von_Smartphones_16_04_2012(1).pdf&mode=0&b=Presse&bc=Presse%7cPresseinformationen%7c2012

[BTSPEZ21 2007] Bluetooth-Special-Interest-Group: *BLUETOOTH SPECIFICATION Version 2.1 + EDR*, 2007. – URL https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=241363

[Dr. Kai-Oliver Detken 2007] Dr. Kai-Oliver Detken, Prof. Dr. Evren E.: Bluetooth-Sicherheit - Schwachstellen und potenzielle Angriffe / FH Dortmund, DECOIT GmbH. URL http://www.detken.net/papers/Detken-Eren_Bluetooth-Beitrag.pdf, 2007. – Forschungsbericht

[Frontline Test Equipment 2007] Frontline Test Equipment, Inc.: Bluetooth Secure Simple Pairing, Simple Pairing Debug Mode, and Interoperability Testing / Frontline Test Equipment, Inc. URL www.fte.com/docs/ssp.pdf, 2007. – Forschungsbericht

[Hentschelmann 2005] Hentschelmann, AndrÃ©: Angriffe auf Bluetooth - Systeme. In: *happy-security.de* (2005). – URL http://digilib.happy-security.de/files/Angriffe_auf_Bluetooth_Systeme_Ausarbeitung.pdf

[Hildebrandt 2004] Holger Hildebrandt, Kay P.: Bluetooth-Anwendungen / TU-Ilmenau. URL http://yukon.e-technik.tu-ilmenau.de/~webkn/Abschlussarbeiten/Studienarbeiten/sta_hildebrandt+Pein.pdf, 2004. – Forschungsbericht

[Holtkamp 2003] Holtkamp, Heiko: EinfÃ¼hrung in Bluetooth / Technische Fakultät, AG RVS Universität Bielefeld. URL http://www.sorex-austria.com/tl_files/news/News_Pdf/bluetooth.pdf, 2003. – Forschungsbericht

[Iman ALMomani und AL-Akhras 2011] Iman ALMomani, Mohammed Al-Saruri ; AL-Akhras, Mousa: Secure Public Key Exchange against Man-In-The-Middle Attacks during Secure Simple Pairing (SSP) in Bluetooth / King Abdullah II School for Information Technology, The Jordan University. URL http://idosi.org/wasj/wasj13(4)/22.pdf, 2011. – Forschungsbericht

[ITWISSEN2012] : *ECDH (elliptic curve diffie hellman)*. – URL http://www.itwissen.info/definition/lexikon/elliptic-curve-diffie-hellman-EC-DH.html

[K.Saravanan u. a.] K.Saravanan, L.Vijayanand ; R.K.Negesh, Lecturer/CSE ; 2, 3Lecturer/EEE: A Novel Bluetooth Man-In-The-Middle Attack Based On SSP using OOB Association model / Erode Sengunthar Engineering college and 3Marthandam college of Engineering and Technology. URL arxiv.org/pdf/1203.4649. – Forschungsbericht

[Kutzner 2006] Kutzner, Sebastian: Angriffe via Bluetooth / Bocum-Ruhr-Universität. URL http://nds.hgi.rub.de/lehre/seminar/SS06/Kutzner_Bluetooth.pdf, 2006. – Forschungsbericht

[Lindell 2008] Lindell, Andrew Y.: Attacks on the Pairing Protocol of Bluetooth v2.1. In: *www.blackhat.com* (2008). – URL http://www.blackhat.com/presentations/bh-usa-08/Lindell/BH_US_08_Lindell_Bluetooth_2.1_New_Vulnerabilities.pdf

[Mettala 1999] Mettala, Riku: *Bluetooth Protocol Architecture*, 1999. – URL https://www.bluetooth.org/docman/handlers/DownloadDoc.ashx?doc_id=89&vId=128

[PcWelt 2008] PcWelt: Handy als Autoschlüssel für Fahrzeuge. In: *pcwelt.de* (2008). – URL http://www.pcwelt.de/news/Nissan-Handy-als-Autoschluessel-fuer-Fahrzeuge-73395.html

[Ringwald 2012] Ringwald, Matthias: BTstack - Bluetooth for All! (2012). – URL http://code.google.com/p/btstack/

[Roger Cheng 2011] Roger Cheng, Stefan B.: Handymarkt: Smartphone-Anteil wächst 2011 auf 28 Prozent. In: *zdnet.de* (2011). – URL http://www.zdnet.de/41555248/handymarkt-smartphone-anteil-waechst-2011-auf-28-prozent/

[Rudi Latuske 2009] Rudi Latuske, ARS Software G.: Sichere und robuste Bluetooth-Kommunikation. In: *http://www.elektroniknet.de* (2009). – URL http://www.elektroniknet.de/automotive/technik-know-how/prozesse-standards-und-qualitaet/article/1602/0/Sichere_und_robuste_Bluetooth-Kommunikation/

[Shaked und Wool 2005a] Shaked, Yaniv ; Wool, Avishai: Cracking the Bluetooth PIN / School of Electrical Engineering Systems, Tel Aviv University, Ramat Aviv 69978, ISRAEL. URL http://static.usenix.org/event/mobisys05/tech/full_papers/shaked/shaked.pdf, 2005. – Forschungsbericht

[Shaked und Wool 2005b] Shaked, Yaniv ; Wool, Avishai: Cracking the Bluetooth PIN / School of Electrical Engineering Systems, Tel Aviv University, Ramat Aviv 69978, ISRAEL. URL http://www.eng.tau.ac.il/~yash/shaked-wool-mobisys05/, 2005. – Forschungsbericht

[sicherheit.info 2012] sicherheit.info: Airkey macht das Handy zum Schlüssel. In: *sicherheit.info* (2012), Oktober. – URL http://www.sicherheit.info/SI/cms.nsf/si.ArticlesByDocID/1126728?Open

[Sikora 2008] Sikora, Prof. Dr. A.: Bluetooth-Grundlagen. In: *http://www.tecchannel.de* (2008). – URL http://www.tecchannel.de/netzwerk/wlan/401459/bluetooth_grundlagen_herkunft_und_funktionsweise/

[STM32F4 2011] *STM32F4 high-performance discovery board*, 2011. – URL http://www.st.com/internet/com/TECHNICAL_RESOURCES/TECHNICAL_LITERATURE/DATA_BRIEF/DM00037955.pdf

[www.idealo.de 2013] WWW.IDEALO.DE: *Preisvergleich > Suche stm32f4discovery*. Januar 2013. – URL `http://www.idealo.de/preisvergleich/MainSearchProductCategory.html?q=stm32f4discovery`

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 7. Februar 2012 Michael Rifkin