HOCHSCHULE MITTWEIDA
University of Applied Sciences

MASTERARBEIT

Herr
Felix Fischer, B.Sc.

Forensische Analyse von Mesh-Netzwerken am Beispiel von Meshtastic

Mittweida, Februar 2023

Fakultät **Angewandte Computer- und Biowissenschaften**

MASTERARBEIT

Forensische Analyse von Mesh-Netzwerken am Beispiel von Meshtastic

Autor:
Felix Fischer

Studiengang:
Cybercrime / Cybersecurity

Seminargruppe:
CY20wC-M

Erstprüfer:
Prof. Ronny Bodach

Zweitprüfer:
M.Sc. Stefan Schildbach

Einreichung:
Mittweida, 20.02.2023

Verteidigung/Bewertung:
Mittweida, 2023

Faculty of **Applied Computer Sciences and Biosciences**

MASTER THESIS

Forensic analysis of mesh networks by using Meshtastic as an example

Author:
Felix Fischer

Course of Study:
Cybercrime / Cybersecurity

Seminar Group:
CY20wC-M

First Examiner:
Prof. Ronny Bodach

Second Examiner:
M.Sc. Stefan Schildbach

Submission:
Mittweida, 20.02.2023

Defense/Evaluation:
Mittweida, 2023

Bibliografische Beschreibung:

Fischer, Felix:
Forensische Analyse von Mesh-Netzwerken am Beispiel von Meshtastic. – 2023. – 75 S.
Mittweida, Hochschule Mittweida – University of Applied Sciences, Fakultät Angewandte Computer- und Biowissenschaften, Masterarbeit, 2023.

Referat:

Diese Masterarbeit prüft forensische Ansätze zur Analyse von Mesh-Netzwerken am Beispiel eines Meshtastic®-Netzes. Hierzu wurden Daten des Funknetzwerkes extern, sowie durch Aufzeichnung des internen Nachrichtenverkehrs erhoben. Mit diesen Daten konnte die Existenz des Netzwerkes nachgewiesen, dessen Teilnehmer identifiziert, sowie deren geographische Positionen offengelegt werden. Darüber hinaus konnte die Netzwerkstruktur partiell rekonstruiert und der Nachrichteninhalt protokolliert werden. Schließlich konnten Maßnahmen identifiziert werden, sich einer Analyse zu entziehen, was einerseits Perspektiven der Sicherheitsintensivierung offenbart und andererseits fortführende forensische Untersuchungen bedingt.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

AES ..... Advanced Encryption Standard

AP ..... Access-Point

API ..... Application Programming Interface

ATAK ..... Android Team Awareness Kit

BDS ..... Beidou

BLE ..... Bluetooth Low Energy

BSSID ..... Basic Service Set ID

CCM ..... Counter with CBC-MAC (Cipher Block Chaining Message Authentication Code)

CEP ..... Circular Error Propability

CSS ..... Chirp-Spread-Spektrum

EU ..... Europäische Union

GLONASS ..... Globalnaja nawigazionnaja sputnikowaja sistema

GNSS ..... Global Navigation Satellite Systems

GPIO ..... General Purpose Input/Output

GPS ..... Global Positioning System

HTML ..... Hypertext Markup Language

HTTPS ..... Hypertext Transfer Protocol Secure

I/O ..... Input and Output

ICMP ..... Internet Control Message Protocol

IEEE ..... Institute of Electrical and Electronics Engineers

IoT ..... Internet of Things

IP ..... Internet Protocol

JSON ..... JavaScript Object Notation

LoRa® ..... Long Range

LoRaWAN ..... Long Range Wide Area Network

MAC ..... Media Access Control

MQTT ..... Message Queueing Telemetry Transport

OLED ..... Organic Light-Emitting Diode

OSI ..... Open Systems Interconnection

OSPF . . . . . Open Shortest Path First

PSK . . . . . Pre-Shared-Key

PSRAM . . . . . Pseudostatic Random Access Memory

QR-Code . . . . . Quick Response Code

QZSS . . . . . Quasi-Zenith Satellite System

REST . . . . . Representational State Transfer

RFC . . . . . Request For Comments

RIP . . . . . Routing Information Protocol

ROM . . . . . Read Only Memory

RSSI . . . . . Received Signal Strength Iindication

RTC . . . . . Realtime-Clock

SDR . . . . . Software Defined Radio

SMS . . . . . Short Message Service

SNR . . . . . Signal to Noise Ratio

SoC . . . . . System on a Chip

SPI . . . . . Serial Peripheral Interface

SRAM . . . . . Static Random Access Memory

SSID . . . . . Service Set ID

TCP . . . . . Transmission Control Protocol

TTL . . . . . Time to Live

UDP . . . . . User Datagram Protocol

ULP . . . . . Ultra Low Power

URL . . . . . Uniform Resource Locator

USB . . . . . Universal Serial Bus

UTC . . . . . Coordinated Universal Time

UTF . . . . . Unicode Transformation Format

WLAN . . . . . Wireless Local Area Network

1 Einleitung

1.1 Motivation

Funknetzwerke in Anbindung an Microcomputer, wie bei IoT-Sensoren, werden immer mehr zum Kommunikationsstandard [1–4]. Gleichzeitig stehen immer mehr Softwarebibliotheken für Funkstandards zum Aufbau von Mesh-Netzwerken unter Open-Source-Lizenz frei zur Anwendung in Projekten zur Verfügung. Dazu zählen ESP-WIFI-MESH für den Standard IEEE 802.11, ESP-BLE-MESH für den Standard IEEE 802.15.1 oder Meshtastic® für den Standard LoRa® [5–7]. Es ist davon auszugehen, dass die Nutzung dieser Protokolle in Zukunft auf Grund der Kombination aus geringen Hardwarekosten und frei verfügbarer Softwarebibliotheken stark zunehmen wird. Dies ist der Anlass, die Sicherheit und die Möglichkeiten forensischer Untersuchungen dieser Mesh-Netzwerke zu überprüfen.

1.2 Zielsetzung

In dieser Masterarbeit sollen forensische Analyseansätze für eine Untersuchung eines Meshtastic®-Netzwerkes abgeleitet werden. Insbesondere werden Methoden erforscht, wie ein existierendes Netzwerk aufgedeckt werden kann. Darüber hinaus sollen die einzelnen Knoten identifiziert und lokalisiert werden. Meshtastic® kombiniert verschiedene Technologien, zu denen LoRa®, BLE, WLAN und GPS gehören [8]. Folglich bieten sich diverse Möglichkeiten Netzwerkteilnehmer zu lokalisieren. Hier soll insbesondere auf die unterschiedlichen Herangehensweisen und deren Resultate eingegangen werden.

Meshtastic® unterstützt mit AES eine verschlüsselte Kommunikation [9]. Deshalb soll geprüft werden, inwiefern diese Schwachstellen in der Implementierung aufweisen. Ferner soll die dafür genutzte Schlüsselgenerierung auf Fehler analysiert werden.

Für eine forensische Aufzeichnung von Nachrichten soll eine Möglichkeit geschaffen werden, diese zu protokollieren. Darüber hinaus sollen die somit gewonnen Daten auf forensischen Wert untersucht werden. Die somit gewonnen Informationen fließen in die Auswertung der zuvor genannten Punkte mit ein.

Abschließend wird ein Fazit aus den gewonnen Forschungsergebnissen gezogen. Dabei lassen sich Abwehrmaßnahmen schlussfolgern, mit denen die forensischen Analysen begegnet werden kann. Daraus können Maßnahmen abgeleitet werden, um sowohl die Sicherheit als auch die forensische Analyse zu verbessern.

1.3 Abgrenzung

Meshtastic® deckt durch die gemeinschaftliche Entwicklung ein breites Einsatzfeld ab. Dieses reicht von reiner Textkommunikation über Positionserfassung und Routenplanung bis zu Datenübertragung von Messdaten. Zur Fokussierung dieser Masterarbeit wird insbesondere auf Aspekte eingegangen, die in anderen Mesh-Netzwerken erwartet werden können. Dazu zählt insbesondere die allgemeine Netzwerkfunktionalität und -struktur, sowie die Nutzung der LoRa®-Technologie. Bei dem Nachrichteninhalt wird eine Konzentrierung auf durch die Mesh-Hardware standardmäßig bereitgestellten Funktionen festgelegt. Dazu zählen insbesondere Angaben zu GPS-Koordinaten, Geräteinformationen und Textnachrichten [10–12].

Darüber hinausgehende Funktionen, wie das Auslesen von zusätzlicher Sensoren mittels GPIO oder die externe Anbindung des Mesh-Netzwerkes ans Internet werden nicht betrachtet. Ebenfalls wird auf eingebettete Nachrichten anderer Transportprotokolle, wie MQTT, TCP oder UDP, nicht eingegangen.

1.4 Dokumentaufbau

Der Aufbau dieser Arbeit gliedert sich in die Kapitel Grundlagen, forensische Ansätze, Durchführung und Ergebnisse. Zunächst liefert das anschließende Kapitel 2 eine Aufbereitung von grundlegendem Wissen zum Verständnis dieser Arbeit. Somit erläutert das Kapitel die theoretischen Grundlagen zu Mikrokontrollern, GPS, LoRa®, sowie dem verwendeten Mesh-Netzwerk Meshtastic®. Dabei wird insbesondere auf Hardware und Software eingegangen.

Anschließend stellt das Kapitel 3 mögliche forensische Ansätze für eine Analyse vor. Dabei werden unterschiedliche Maßnahmen für die Einzelschritte Erkennung, Teilnehmeridentifizierung, und Nachrichtenprotokollierung aufgezeigt. Gleichzeitig werden theoretische Limitierungen der einzelnen Methoden erläutert.

Aus der theoretischen Betrachtung können anschließend umsetzbare Vorgehensweisen abgeleitet werden. Deren Umsetzung beschreibt das Kapitel 4. Hierbei werden eingesetzte Programme und Techniken detailliert aufgeführt.

Darauf folgend werden die somit gewonnen Daten präsentiert und ausgewertet. Es findet eine tiefgreifende Analyse der Ergebnisse und deren Auswirkungen statt. So wird fortführend auch auf denkbare Maßnahmen zur Umgehung der vorgestellten Umsetzungen eingegangen.

Abschließend ist das Kapitel 6 angefügt, in dem Perspektiven für zukünftige Forschungsarbeiten präsentiert werden.

2 Grundlagen

Die verwendete Hardware und Software liegt außerhalb des vorausgesetzten allgemeinen Informatikwissens. Zur Verbesserung der Verständlichkeit wird zunächst auf einige Begriffe und Hintergrundinformationen eingegangen.

2.1 Mikrocontroller

Ein Prozessor stellt die Zentrale Verarbeitungseinheit eines Computers dar. Mikrocontroller erweitern den Prozessor durch zusätzlich integrierte Komponenten, wie Arbeitsspeicher, Analog-Digital-Wandler und Kommunikationsmodule. Somit werden auf einem Siliziumchip mehrere Komponenten zu einem Gesammtsystem vereinigt. Folglich werden Mikrocontroller auch als System on a Chip, kurz SoC, bezeichnet. [13]

2.2 Bluetooth Low Engergy

Bluetooth ist ein Protokoll zur Datenübertragung, welches von der Bluetooth SIG, Inc. entwickelt wird. Dabei hat sich Bluetooth zu einem Standard für Punkt-zu-Punkt-Übertragung für mediale Endgeräte entwickelt, ist jedoch nicht darauf beschränkt. Ergänzend existiert Bluetooth Low Energy, kurz BLE, welches insbesondere auf niedrigen Stromverbrauch optimiert wurde. Die beiden Protokolle verwenden jeweils die Frequenz 2,4 GHz. [14]

2.3 Wireless Local Area Network

Die Bezeichnung Wireless Local Area Network, kurz WLAN, ist, wie der Name bereits beschreibt, ein drahtloses Netzwerk in lokaler Umgebung. Dabei handelt es sich um einen übergreifenden Begriff, welcher viele Übertragungsprotokolle umfasst. Von dem Institute of Electrical and Electronics Engineers, kurz IEEE, werden diese Protokolle unter IEEE 802.11 zusammengefasst. [15]

Für diese Arbeit sind insbesondere die Standarderweiterungen IEEE 802.11b, g und n zu betrachten. Diese definieren eine drahtlose Datenübertragung mit der Frequenz 2,4 GHz. Außerdem ermöglichen die Erweiterungen schnellere Übertragungsraten gegenüber des ursprünglichen Standards 802.11. [15]

Darüber hinaus entwickelte die Firma Espressif den Modus LR (Long Range), welcher laut Hersteller eine Reichweite bis zu einem Kilometer überbrückt. Dieser unabhängig vom IEEE entwickelte Übertragungsstandard dient vorrangig der Kommunikation zwischen Mikrocontrollern der Firma Espressif untereinander. [16]

2.4 ESP32

Die Mikrocontroller der ESP32-Familie von der chinesischen Firma Espressif sind leisttungsstärkere Nachfolger des ESP8266. Wie sein Vorgänger, kombinieren auch die ESP32 einen Xtensa-Prozessor mit WLAN im 2,4 GHz Band nach IEEE 802.11 b/g/n. Darüber hinaus unterstützen die ESP32-Mikrocontroller nativ Bluetooth und Bluetooth Low Energy. Weiterhin enthalten die ESP32 einen Tiefschlafmodus, im Englischen Deep-Sleep-Mode, wodurch ein sehr geringer Stromverbrauch von 10-150 μA ermöglicht wird. Dieser wird durch einen Ultra Low Power (ULP)-Coprozessor realisiert. Durch diesen kann der Hauptprozessor sowohl durch PIN-Interrupts, als auch durch die integrierte Realtime-Clock (RTC) geweckt werden. Damit eignen sich die ESP32 besonders gut für batteriegespeiste Projekte. Folglich kommen diese häufig in IoT-Geräten zum Einsatz. [17]

Für Programme stehen 448 KB ROM-Speicherplatz direkt innerhalb des Prozessors zur Verfügung. Speziell für Programmabschnitte die eng mit dem Tiefschlafmodus zusammenarbeiten, existiert ein 16 KB großer SRAM-Speicher. Während der Ausführung können alle Programme auf 520 KB SRAM-Arbeitsspeicher zugreifen. Zum dauerhaften, aber überschreibbaren, speichern von Ergebnissen kann ein externer Flashspeicher über SPI angebunden werden. Dieser kann gleichzeitig auch zum Hinterlegen von Programmcode genutzt werden. Dafür kommen typischer Weise 8 MB PSRAM und 4 MB Flashspeicher zum Einsatz. [17]

2.5 Global Positioning System

Global Positioning System, kurz GPS, ist ein Ortungsverfahren basierend auf Satelliten und Zeitdifferenzen durch die Übertragungsstrecke zum Empfänger. Dabei agiert der Empfänger rein passiv und emittiert selbst keine Signale. Die ermittelte Position wird auf der durch den Ellipsoid WGS-84 modellierten Erdoberfläche bestimmt. Dabei wird die berechnete Position durch die drei Koordinaten Latitude (Breitengrad), Longitude (Längengrad) und Altitude (Höhe) angegeben. Ursprünglich entwickelte das US-Verteidigungsministerium dieses System zur militärischen Nutzung. Darüber hinaus ist das System auch zivil nutzbar und wird besonders für zwei Zwecke genutzt. Einerseits dient GPS zur eigenen Positionsbestimmung und erleichtert die Navigation. Andererseits wird GPS verwendet, um die Position von Gegenständen zeitlich zu protokollieren. So findet GPS Anwendung in der Detektion von Holzdiebstahl [18, 19]. [20]

Neben GPS gibt es noch weitere Systeme zur globalen Positionsbestimmung. Dazu zählen das europäische Galileo, das russische Globalnaja nawigazionnaja sputnikowaja sistema (GLONASS) und das chinesische Beidou (BDS) [21]. Diese sind für diese Arbeit jedoch nicht relevant.

Öffentlich käuflich gibt es viele GPS-Empfänger zu erwerben. Im Bereich der IoT kommt oftmals der GPS-Chip Neo-6M zum Einsatz. So ist insbesondere dieser Chip für diese Arbeit relevant. Der GPS-Chip Neo-6M der Firma u-blox besitzt eine Genauigkeit (CEP) von 2,5 m [22]. Zur erstmaligen Positionsermittlung aus einem Kaltstart benötigt dieser Chip 27 s [22]. Zum Empfang des GPS-Signals kommen typischer Weise Keramikpatchantennen zum Einsatz. Hierbei existieren zwei sehr häufig verwendete Modelle. Diese sind in der Größe eines Quadrates mit 25 mm x 25 mm oder bei der kleineren Antenne im rechteckigen Format mit den Maßen 6 mm x 16 mm. Durch die empfindlichere Signalaufnahme der quadratischen Antenne ermöglicht diese auch in Innenräumen einen zuverlässigen Empfang.

2.6 LoRa®

Bei Long Range, kurz LoRa®, handelt es sich um eine Übertragungstechnik auf Layer 1 des OSI-Modells. Die Übertragung findet mit einer sehr geringen Sendeleistung von 27 mW, jedoch sehr breitbandig auf öffentlichen Frequenzen im 433 Mhz oder 868 Mhz Band inerhalb der EU statt. Es arbeitet mit dem Chirp-Spread-Spektrum-Modulationsverfahren, kurz CSS, und erreicht damit trotz der geringen Sendeleistung Übertragungen auf Sichtentfernung. Insofern Sichtverbindung besteht, können sogar Reichweiten bis 200 km erreicht werden [23]. Unter Nutzung von Meshtastic® wird eine maximal erreichte Funkstrecke von 166 km angegeben [24]. Ein Netzwerk in Größe einer Stadt ist damit durchaus realistisch. Diese großen Reichweiten ermöglichen eine Überbrückung der letzten Meile für IoT-Geräte im Außenbereich. Mit Infrastrukturen, wie LoRaWAN, können diese Geräte an das Internet angebunden werden. [25, 26]

Je nach Konfiguration kann bei LoRa® die Übertragung auf Datendurchsatz oder Reichweite optimiert werden. Für die Umsetzung wird die Bandbreite und der Spread-Faktor angepasst. In der Firmware von Meshtastic® werden sogenannte Presets als vordefinierte Konfiguration bereitgestellt. Somit wird eine Einstellung der Funkkonfiguration erleichtert. [27]

2.7 Advanced Encryption Standard

Ausgestrahlte Funksignale lassen sich von allen Empfängern in der Reichweite des Signals erfassen. Folglich muss der Kommunikationsinhalt verschlüsselt werden, um diesen gegen Zugriff von Dritten zu schützen. Zu diesem Zweck eignet sich die symmetrische Verschlüsselung. Symmetrisch bezeichnet in der Kryptographie, dass für das Verschlüsseln und Entschlüsseln der selbe Schlüssel verwendet wird. Insofern sowohl Sender, als auch Empfänger den identischen, symmetrischen Schlüssel besitzen, können diese untereinander verschlüsselte Nachrichten austauschen. Dieser gemeinsame Schlüssel muss im Vorfeld zur Kommunikation ausgetauscht werden. Dementsprechend wird dieser Schlüssel im englischen als Pre-Shared-Key, kurz PSK, bezeichnet. [28]

Bei Advanced Encryption Standard, kurz AES, handelt es sich um ein symmetrisches Kryptographieverfahren. Genauer spezifiziert stellt AES eine symmetrische Blockverschlüsselung dar. Dabei wird eine Nachricht in Blöcke aufgeteilt, welche anschließend symmetrisch verschlüsselt werden. [29]

2.8 Mesh-Netzwerk

Der englische Begriff Mesh beschreibt ein Netz oder ein Geflecht. Dementsprechend handelt es sich bei einem Mesh-Netzwerk um ein netzartiges oder verflochtenes Netzwerk. Dieses steht einer klaren Struktur, wie Stern, Baum oder Ring gegenüber. Man spricht auch von einer vermashten Topologie. Es existiert also keine wohldefinierte Struktur. Jeder Knoten kann mit jedem anderen, aber auch nur mit einem einzigen anderen Knoten verbunden sein. Häufig werden diese Netzwerke auch als Ad-Hoc-Netzwerk bezeichnet. [30]

2.9 Meshtastic®

Bei Meshtastic® handelt es sich um ein Open-Source-Softwareprojekt zum Aufbau eines Mesh-Netzwerkes zum Versenden von Textnachrichten. Vergleichbar ist dieses mit einem privaten Netzwerk zum Versenden von SMS-Nachrichten. Darüber hinaus können jedoch auch kleine Datenmengen übertragen werden. [8]

Das Wort Meshtastic® ist ein Neologismus aus den kombinierten Wörtern Mesh, zu deutsch verflochten, und fantastic, zu deutsch fantastisch. Meshtastic® setzt sich als Ziel ein nutzerfreundliches System zum Versenden von Nachrichten bereitzustellen [8]. Insbesondere soll die Technik von jedem legal ohne Voraussetzungen, wie Funklizenzen, betrieben werden können [8]. Dabei werden als Zielgruppe vorrangig Sportler im Außenbereich angesprochen. Explizit nennt die Projektwebseite als Sportarten Wandern, Skifahren und Paragleiten [31]. Darüber hinaus eignet sich das System für Katastrophenfälle, bei denen keine Kommunikationsinfrastruktur mehr besteht. Das kann beispielsweise bei Überschwemmungen der Fall sein.

Das Meshtastic®-Projekt setzt sich aus verschiedenen Komponenten zusammen. Als Hauptprojekt kann die Software zum Aufbau des Mesh-Netzwerkes angesehen werden [32]. Diese wird als ”Firmware” bezeichnet. Das Projekt unterstützt verschiedene Hardwarelösungen. Jedoch wurde in Koorperation mit LILYGO und Neil Hao jeweils speziell auf Meshtastic® zugeschnittene Hardware entworfen [33, 34]. Mit diesen und anderen verwendeten Geräten kann sowohl über ein Python-Komandozeilen-Tool, als auch über das Handy mit einer Android-App kommuniziert werden. Folglich werden diese beiden Komponenten als ”Python” und ”Meshtastic-Android” betitelt. Zuletzt kann eine Dokumentation und Anleitungen über alle Projektkomponenten auf einer Webseite nachgelesen werden. Diese existiert ebenfalls als Open-Source-Projekt unter dem Namen ”Meshtastic”. [35]

Mit goTenna® existiert ein kommerzieller Anbieter einer kompletten proprietären Meshtastic®-Lösung [36]. Auf diese wird in dieser Arbeit nicht eingegangen. Im Folgenden wird die verwendete Hardware und Software detaillierter beleuchtet.

2.9.1 Meshtastic®-Hardware

Die für den Aufbau des Mesh-Netzwerkes verwendete Hardware ist nicht auf eine Konfiguration begrenzt. Vielmehr stehen verschiedene Hardwarekombinationen zur Auswahl. Allen gemein ist, dass diese stets einen Mikroprozessor, meist einen ESP32, mit einem LoRa®-Modul kombinieren. Außerdem wird in den meisten Fällen zusätzlich ein Bildschirm integriert. Darüber hinaus verfügen einige Module über GPS- und Akkumodule. Für eine präzisere Bezeichnung wird die Hardware zum Aufbau des Mesh-Netzwerkes in diesem Dokument fortführend als Mesh-Hardware betitelt. Anschließend sind die einzelnen unterstützen Mesh-Hardware-Varianten aufgelistet. [37]

RAK WisBlock Die RAK-WisBlock-Familie setzt auf einen modularen Ansatz, bei dem Zusatzfunktionalität individuell angeschlossen werden kann. Als Grundlage stehen zwei Boards zur Auswahl. Das Board RAK4631 kombiniert einen nRF52840-Mikrokontroller mit einem SX1262-LoRa®-Empfänger. Der Prozessor erlaubt einen sehr geringen Energieverbrauch und eine Verbindung über BLE 5.0. Alternativ wird das Board RAK11200 unterstützt. Dieses verwendet einen ESP32-WROVER

Mikrokontroller und stellt damit sowohl Bluetooth 4.2 als auch WLAN bereit. Eine Anbindung über LoRa® wird durch ein eigenständiges Modul realisiert. Hierfür wird des RAK13300-LPWAN-Moduls empfohlen. [38]

Beide zur Wahl stehenden Boards lassen sich im 433 MHz als auch 868 MHz Band betreiben. Darüber hinaus werden Frequenzen für den US-Markt unterstützt. Ebenso besteht ein Support für sowohl 0,96 Zoll OLED- als auch 2,13 Zoll e-Ink-Displays. GPS-Funktionalität kann mit dem uBlox-MAX-7Q-GPS-Modul bzw. mit dem uBlox-Zoe-M8Q-GNSS-Empfänger bereitgestellt werden. Letzteres unterstützt neben GPS sowohl GLONASS und BDS, als auch die GPS-Ortungsverbesserung QZSS für den japanischen Raum durch zusätzliche Satelliten. Außerdem existieren für die RAK-WisBlock-Familie Erweiterungsmodule für Buzzer, I/O-Erweiterungen und dem Umweltsensor BME680. Zusätzlich hebt sich die Boardfamilie durch eine native Unterstützung von 5V-Solar-Paneelen hervor. Eine Beispielkonfiguration ist in Abbildung 2.1 dargestellt. [38]

Diese Boardgruppe orientiert sich dementsprechend an einem experimentellen Aufbau und für die Realisierung einer stationären zentralen Relaystation mit Solarbetrieb. Durch den modularen Aufbau ist das Gesamtsystem fragil, groß und somit eher unportabel.

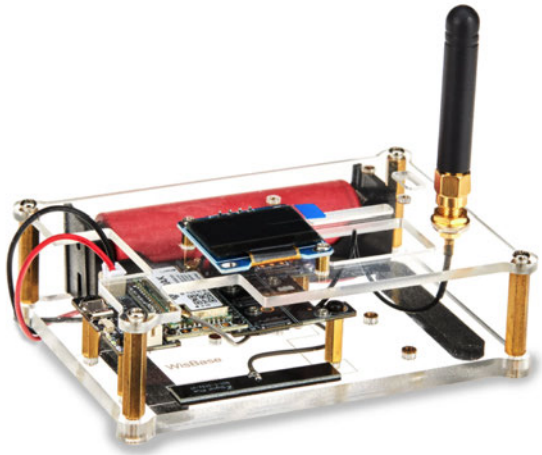


Abbildung 2.1: RAK4631-Board mit OLED-Display [39]

LILYGO® T-Echo Das LILYGO®-T-Echo-Modul kombiniert GPS-Empfänger, Akku, e-Ink-Display, LoRa®-Empfänger und einen sparsamen Mikrokontroller zu einem Gesamtpaket, welches zum Betrieb eines Meshtastic®-Knotens vollständig ausgerüstet ist. Dabei verwendet es den nRF52840-Mikrokontroller und SX1262-LoRa®-Modul. Auf dem 1,54 Zoll großen e-Ink-Display können Nachrichten und andere Informationen dauerhaft angezeigt werden. Folglich ist durch die Verwendung eines e-Ink-Displays der Stromverbrauch gegenüber eines OLED-Displays reduziert. Außerdem ist die Positionsbestimmung durch das L76K-GNSS-Modul durch den Empfang von GPS, GLONASS, BDS und QZSS sehr genau. Für eine zuverlässige Stromversorgung sorgt der eingebaute 850 mAh Akku. Darüber hinaus wird das Gesamtpaket mit einem Gehäuse ausgeliefert. [33, 40]

Derzeit ist das Board in zwei Ausführungen erhältlich. So besitzt die BME820-Variante zusätzlich zu der Normal-Ausführung einen BME820-Sensor [33]. Folglich kann mit diesem Temperatur, Luftfeuchtigkeit und der Luftdruck erfasst werden. Beide Versionen verwenden die gleiche Grundausstattung und das identische Gehäuseformat. Das LILYGO® T-Echo ist in Abbildung 2.2 dargestellt.

Abbildung 2.2: LILYGO® T-Echo mit e-Ink-Display [41]

LILYGO® T-Beam Es existieren innerhalb der T-Beam-Familie verschiedene Boards. Grundlegend ist allen gemein, dass ein ESP32 mit GPS kombiniert wurde. Optional kann ein Display direkt auf das Layout mit Pins gelötet werden. Dadurch behalten die Geräte ihr kompaktes Format. Zusätzlich beinhalten die meisten Geräte auf der Rückseite eine Halterung für einen Akku im 18650 Größenformat, welches auch häufig für E-Zigaretten oder Akkubanks verwendet wird. Darüber hinaus wird für den Empfang von GPS das Modul Neo-6M-GPS oder Neo-M8N-GNSS verbaut. Letzteres ermöglicht durch die Unterstützung von GLONASS, BDS und QZSS eine genauere Positionsbestimmung. Das LoRa®-Modul SX1278 bzw. SX1276 ermöglicht einen Betrieb auf dem 433 Mhz bzw. 868 Mhz, 915 Mhz und 923 Mhz. Folglich ist ein weltweiter Einsatz ohne Funklizenz möglich. [42, 43]

Als besonderes Merkmal des T-Beam lässt sich festhalten, dass es seit Version 1.1 Module mit bereits vorinstallierter Meshtastic®-Firmware ohne signifikanten Aufpreis zu erwerben gibt [44]. Diese soll sich mittels Handy-App über Bluetooth aktualisieren lassen. Folglich können auch technisch unversierte Nutzer ein Netzwerk aufbauen.

In Abbildung 2.3 ist ein solches T-Beam-Gerät abgebildet. Deutlich zu erkennen ist die namensgebende T-Form durch die mittig abstehende LoRa®-Antenne. Der rückseitig montierte Akku ermöglicht ein bequemes Hinstellen des Geräts auf einer glatten Oberfläche. So ist beispielsweise das Display auf einem Schreibtisch stets lesbar. Zum Schutz des Moduls gegen Spritzwasser oder anderen physischen Kräften sind im Internet kostenlos Gehäuseformen downloadbar, welche mittels 3D-Druck hergestellt werden können [45].

(a) Vorderseite [46] **(b)** Rückseite [47]

Abbildung 2.3: LILYGO® T-Beam mit OLED-Display und NEO-6M-GPS

LILYGO® Lora32 Die Board-Familie LILYGO® Lora32 beinhaltet vier Entwicklungsstufen von V1 und V1.3 über V2.0 zur aktuellen Version V2.1-1.6. Diese verbesserten den Aufbau der Platine, um Fehler im Design zu beheben. Übergreifend wird von allen Versionen der Microcontroller ESP32 mit WLAN- und Bluetooth-Unterstützung genutzt. Ab Version V2.0 gibt es auch eine 433 Mhz Variante. Das bereits verbaute 0,96 Zoll große Display liegt flach auf der Platine, wodurch diese ein kleines und vor allem flaches Format behält. So lässt sich in der Abbildung 2.4 die Größe anhand der 50 mm langen Antenne einordnen. Zusätzlich befindet sich auf der Rückseite ein MicroSD-Kartenslot, wodurch Daten durch das Gerät selbst gespeichert werden können. Jedoch besitzt das Gerät keinen GPS-Empfänger und auch von der Nutzung des Akku-Anschlusses wird auf der Meshtastic®-Webseite explizit gewarnt. So gibt es bei allen Versionen unterhalb V2.1-1.6 einen Fehler im Platinendesign, welcher eine Überlastung des Akkus im Ladevorgang zur Folge hat. [48, 49]

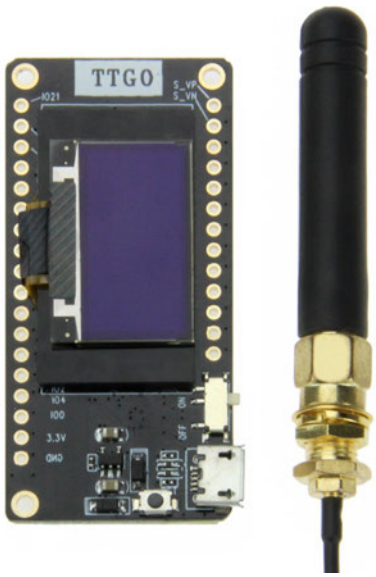


Abbildung 2.4: LILYGO® Lora32 V2.0 mit Antenne [50]

Nano G1 Das Board Nano G1 wurde von Neil Hao speziell für die Verwendung von Meshtastic® entworfen. Dabei sollte das resultierende Design in einem kompakten Formfaktor sein. Dieses ist, wie in Abbildung 2.5 zu sehen, im Kreditkartendesign gehalten. Meshtastic® bietet als Kernfunktion das teilen der eigenen GPS-Position. Folglich wurde für dieses Modul der hoch präzise GPS-Empfänger ATGM336H-5N-71 verwendet. Dieser bestimmt die aktuelle Position mittels GPS, GLONASS und BDS. Obwohl der Nano G1 einen Semtech SX1276-LoRa®-Chip verwendet, kann auf Grund der in der Platine enthaltenen Antenne nur auf der Frequenz 915 MHz gesendet werden. Diese Frequenz ist nur im US-Raum frei nutzbar. Dementsprechend wird dieses Board auch nur im US-Markt zum Kauf angeboten. [34]

Abbildung 2.5: Nano G1 [51]

Heltec® Das Heltec®-Modul ist vergleichbar mit LILYGO®'s Lora32. So verwendet auch dieses Board einen ESP32-Mikrocontroller, verzichtet aber auf Akku und GPS. Es stehen für dieses Modul zwei Varianten zur Verfügung, welche entweder das 433 MHz oder 868 MHz Band benutzen. Dementsprechend ist entweder der LoRa®-Chip SX1278 (433 MHz) oder SX1276 (868 MHz) verbaut. Als Display dient ein 0,96 Zoll großes OLED-Display, welches fest verbaut wurde. Darüber hinaus wird die Platine und die verwendete Antenne mit einem Gehäuse ausgeliefert. Folglich ist diese Version gegen Spritzwasser geschützt. [52, 53]

Jedoch ist nachteilig zu nennen, dass das Board zwar einen Anschluss für einen Akku besitzt, dieser aber nicht im mitgelieferten Gehäuse verwendet werden kann. Außerdem ist der integrierte Hauptprozessor des SoC-ESP32 der leistungsschwächste aller hier vorgestellten Boards. Darüber hinaus läuft das Heltec®-Platinendesign instabil. Aus den zwei zuletzt genannten Gründen wird das Heltec®-Board bereits ab Version 1.3 nicht mehr vom Meshtastic®-Projekt unterstützt [54].

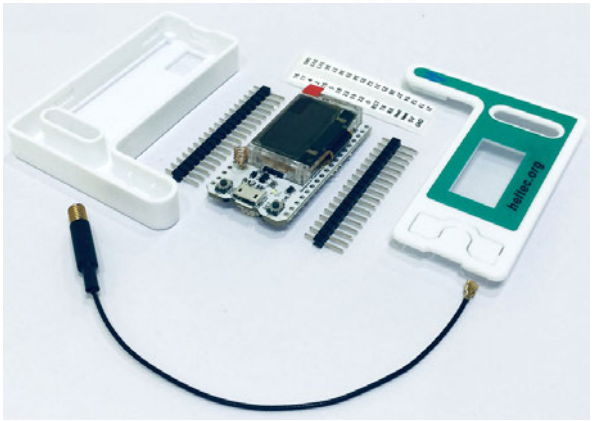


Abbildung 2.6: Heltec® Wifi Lora32 V2.1 [55]

Zusammenfassung Die Tabelle 2.1 fasst die Hardwareausstattung der vorgestellten Boards zusammen.

Tabelle 2.1: Überblick von Meshtastic®-Hardware-Modulen [33, 34, 38, 40, 42, 43, 48, 49, 53]

Board	CPU	LoRa	GPS	Akku	Zulassung
RAK WisBlock	nRF52840 ESP32-WROVER	RAK13300	optional uBlox-MAX-7Q uBlox-Zoe-M8Q	ja	weltweit
LILYGO® T-Echo	nRF52840	SX1262	L76K	ja	weltweit
LILYGO® T-Beam	ESP32	SX1278 SX1276	NEO-6M NEO-M8N	ja	weltweit
LILYGO® Lora32	ESP32	SX1278 SX1276	—	optional nur V2.1-1.6	weltweit
Nano G1	ESP32-WROVER	SX1276	ATGM336H-5N-71	nein	USA
Heltec®	ESP32	SX1278 SX1276	—	nein	weltweit

2.9.2 Meshtastic®-Software

Wie im Kapitel 2.9 bereits angedeutet, setzt sich das Meshtastic®-Projekt aus verschiedenen Software-Komponenten zusammen. Diese werden in diesem Kapitel genauer erläutert. Dem vorangehend sind jedoch die unterschiedlichen Versionen festzuhalten.

Versionen Grundlegend lässt sich festhalten, dass die Hauptversionen 1.2, 1.3 und 2.0 aktuell als relevant zu nennen sind (Stand November 2022). Dabei orientieren sich die Bezeichnungen in den einzelnen Teilprojekten an den Hauptversionen. Jedoch sind die jeweiligen genauen Versionsbezeichnungen für einzelne Änderungen innerhalb der Hauptversionen nicht identisch. So wird beispielsweise der letzte Release der Version 1.2 im Firmware-Teilprojekt als Version 1.2.65.0adc5ce alpha bezeichnet, jedoch die letzte Version im Python-Teilprojekt als Version 1.2.95 betitelt. [56, 57]

Im Juni 2022 war die Version 1.2 die neuste, stabile und damit empfohlene Softwareversion. Darauf folgend wurde mit Version 1.3 an einer grundlegenden Verbesserung der Software gearbeitet. In dieser Version wurde der Quellcode aufgeräumt und Änderungen basierend auf den Erfahrungen des bisherigen Betriebs integriert. Dadurch änderte sich das Routingprotokoll des Netzwerkes und der Aufbau der versendeten Nachrichten umfassend, wodurch diese Version inkompatibel mit der zuvor verwendeten Version 1.2 wird. Version 1.3 ist jedoch als instabiler Prototyp dieser Änderungen zu sehen und dient damit zum Testen der Neuerungen. Mit Version 2.0 wurden die durch Version 1.3 erprobten Änderungen als stabiler Release am 31.10.2022 veröffentlicht. Für den Netzwerkaufbau und der detaillierten Betrachtung in dieser Arbeit wird die Version 2.0.0 verwendet. [56]

Firmware Bei der Firmware handelt es sich um die Software, welche direkt auf der Mesh-Hardware installiert wird. Diese bestimmt die Funktionsweise des Mesh-Netzwerkes und sorgt darüber hinaus für eine Interaktion mit den angeschlossenen Sensoren, Bildschirmen und anderen Hardwarekomponenten. [58]

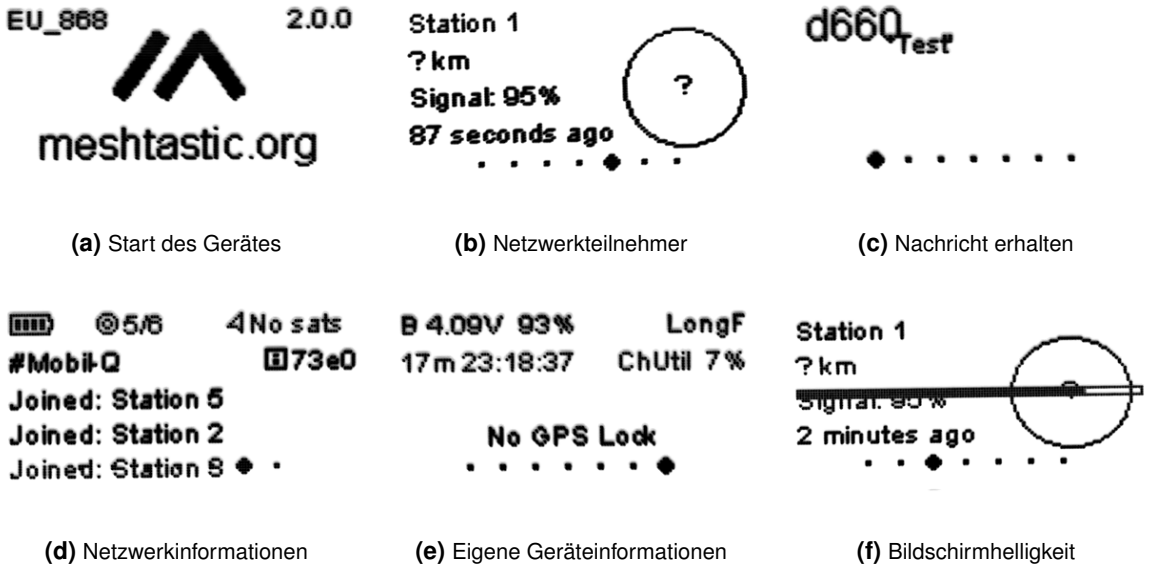
Meshtastic® nutzt die Funktechnik LoRa® zum Aufbau des Mesh-Netzwerkes und folglich dieses zum Austausch von Nachrichten mit Text- und Dateninhalt. Dabei kann ein Gerät auf bis zu acht Kanälen gleichzeitig kommunizieren. Im Gegensatz zu Funkkanälen, wo jeweils andere Frequenzen

benutzt werden, verwenden die Meshtastic®-Kanäle meist den gleichen Funkkanal. Folglich kann die Bezeichnnung Kanal hier eher als Kommunikationskanal verstanden werden. Dieser definiert sich durch einen Index, einen Namen, eine Rolle, einen symmetrischen Schlüssel (PSK) und eine Konfiguration. Dabei definiert der Index die Kanalnummer und der Name eine menschenlesbare Bezeichnung. Für den vergebenen Namen existieren drei reservierte Namensgebungen. Bei diesen handelt es sich um ”admin” für einen administrativen Kanal zur Verwaltung von Mesh-Hardware über das Netzwerk selbst, ”gpio” für einen direkten Zugriff auf GPIO-PINs und ”serial” für serielle Daten eines angeschlossenen Sensors. Weiterführend werden in der Konfiguration Einstellungen für die Funkfrequenz und die LoRa®-Codierung definiert. Als Rolle kann zwischen deaktiviert, primär und sekundär gewählt werden. Wie zu vermuten, kann ein Kanal optional deaktiviert werden, um Senden oder Empfangen von Nachrichten zu verhindern. Dagegen wird stets genau ein Kanal in der primären Rolle benötigt. Auf diesem werden GPS-Positionen oder andere Daten versendet. Darüber hinaus wird dieser, sofern nicht anders angefordert, auch standardmäßig zum versenden von Textnachrichten genutzt. Optional angegebene Kanäle in der Rolle eines sekundären Kanals können, wie der primäre Kanal genutzt werden. Folglich kann mit der Nutzung unterschiedlicher Kanäle unterschiedliche Schlüssel für die Verschlüsselung der Nachrichten genutzt werden. [58]

Für die Verschlüsselung von Nachrichten nutzt Meshtastic® das symmetrische Verschlüsselungsverfahren AES mit einem 128-Bit langen Schlüssel. Hierfür werden die Bibliotheken tiny-AES für Mesh-Hardware mit NRF52-Mikrocontrollern und mbedtls für ESP32-Mikrocontroller verwendet. Grundsätzlich lässt sich festhalten, dass das Verschlüsselungsverfahren AES unter Verwendung einer korrekten Implementierung als ungebrochen angesehen wird. Vorausgesetzt für die sichere Kommunikation wird ein kryptographisch starker, zufällig generierter Schlüssel. Die Generierung dieses Schlüssels erfolgt auf externer, zur Konfiguration genutzter Hardware, wie beispielsweise einem über USB angeschlossenen Computer. Unter Nutzung der Android-App wird der zu verwendende Schlüssel durch ein SecureRandom-Objekt erzeugt. Die Konfiguration der verwendeten Kanäle mit den generierten kryptographischen Schlüsseln kann anschließend über eine generierte URL geteilt werden. Diese URL dient dabei als Codierung und nicht dem Aufruf einer Webseite. Für eine schnelle Übertragung kann die generierte URL auch als QR-Code dargestellt und eingescannt werden. [58]

Zuletzt ist zu nennen, dass die Firmware mit angeschlossenen Sensoren, Bildschirmen und anderen Hardwarekomponenten interagiert. So werden Umweltsensoren, wie der BME820 standardmäßig unterstützt. Weiterführend zeigt ein angeschlossener Bildschirm Informationen des eigenen Gerätes, aber auch erkannter Netzwerkteilnehmer an. Darüber hinaus informiert der Bildschirm den Nutzer über empfangene Nachrichten. Unter Verwendung eines Buzzers wird gleichzeitig ein akustisches Signal abgespielt. Beim Start der Mesh-Hardware zeigt der Bildschirm die verwendete Versionsnummer der installierten Firmware an. In Abbildung 2.7 sind einige mögliche Darstellungen auf dem Bildschirm abgebildet. Hierbei zeigen die Punkte im unteren Bildschirmbereich die Einzelseiten und der einzelne Punkt die derzeit dargestellte Seite an. [58]

Die Abbildung 2.7a zeigt den Startbildschirm beim Anschalten des Gerätes an. In der oberen Zeile lassen sich die Funkeinstellung und die Firmwareversion ablesen. Wie auf der Abbildung zu erkennen, ist diese Mesh-Hardware für einen Einsatz in der EU auf der Frequenz von 868 MHz eingerichtet. Dabei kommt die Firmware 2.0.0 zum Einsatz.



(a) Start des Gerätes **(b)** Netzwerkteilnehmer **(c)** Nachricht erhalten

(d) Netzwerkinformationen **(e)** Eigene Geräteinformationen **(f)** Bildschirmhelligkeit

Abbildung 2.7: Erfassbare Informationen vom OLED-Display (Farben invertiert)

Weiterführend stellt die Abbildung 2.7b detaillierte Informationen von einem speziellen Teilnehmer dar. Es lassen sich Name, Distanz zur eigenen Position, Empfangsstärke und die Zeitdauer seit dem letzten Kontakt ablesen. Insofern sowohl das eigene Gerät, als auch der andere Netzwerkteilnehmer, eine Position ermittelt haben, wird innerhalb des Kreises eine Richtung zum Teilnehmer angezeigt. In diesem Beispiel fehlt eine fremde Positionsangabe.

Ein Beispiel einer empfangenen Textnachricht ist auf der Abbildung 2.7c zu betrachten. Dabei steht in der oberen linken Ecke der Absender der Nachricht. In diesem Fall handelt es sich um den Knoten mit dem Namen d660, was den letzten zwei Bytes seiner Bluetooth Low Energy-MAC-Adresse entspricht.

Eine Übersicht über den aktuellen Status des Mesh-Netzwerkes kann aus der dargestellten Seite in Abbildung 2.7d abgelesen werden. In der ersten Zeile befinden sich Informationen über den eigenen Akkustand, Anzahl der aktiven und bereits gesehenden Netzwerkteilnehmer, sowie die Anzahl der empfangenen GPS-Satelliten. Wie in der Abbildung zu erkennen, besteht das Mesh-Netzwerk derzeit aus 6 Teilnehmern, von denen 5 aktuell aktiv sind. Ferner lässt sich ablesen, dass derzeit kein GPS-Satellit empfangen werden kann. Darunter ist in der zweiten Zeile der primäre Kanalname und die Endung der Hardwareadresse abzulesen. Anschließend folgt in drei Zeilen ein kurzer Log-Bericht, welche Knoten dem Netzwerk beigetreten sind, bzw. dieses verlassen haben.

In der Abbildung 2.7e lassen sich Informationen über das eigene Gerät erfassen. In der ersten Zeile steht erneut der Batteriestand, nur diesmal als Spannungsmessung, sowie auch als prozentualen Zahlenwert. Anschließend folgt eine Angabe über die verwendete Kanaleinstellung, welche in diesem Fall ”LongFast” ist. Daraufhin steht in der zweiten Zeile die Betriebsdauer, die aktuelle Uhrzeit und die Funkkanalauslastung.

Zuletzt zeigt die Abbildung 2.7f eine Änderung der Bildschirmhelligkeit durch langes Drücken der Durchwahltaste.

Protobufs Bei ”protocol buffer”, zusammengefasst protobuf, handelt es sich um einen Standard zum Strukturieren und Serialisieren von Daten. Dabei sind die verwendeten Angaben programmiersprachenübergreifend verwendbar. Dafür stehen die Programmiersprachen Java, Python, Objective-C und C++, sowie ab Version 3 die Sprachen Kotlin, Dart, Go, Ruby und C# zur Auswahl. [59]

Folglich werden im Teilprojekt protobufs Nachrichtenformate von Meshtastic® definiert. Auf die dort festgelegten Definitionen greifen andere Teilprojekte zu, um die Daten korrekt zu verarbeiten. [60]

Web Meshtastic® bietet verschiedene Möglichkeiten mit der Mesh-Hardware zu interagieren. Dieses Teilprojekt stellt eine dieser Möglichkeiten dar. Über einen Webbrowser kann auf die Mesh-Hardware zugegriffen werden. Dafür stehen BLE, WLAN und eine Serielle-USB-Verbindung als Schnittstelle zur Auswahl. Beispielhaft ist unter `https://client.meshtastic.org/` ein Service gehostet. Alternativ lässt sich eine identische Webseite aufrufen, wenn die Mesh-Hardware in den Web-Modus versetzt wird. In diesem Fall lässt sich die Webseite nach dem Aufbau einer WLAN-Verbindung mit dem Gerät über `http://meshtastic.local/` aufrufen. Für die Nutzung der Webseite wird ein chrome-basierter Webbrowser benötigt. Im Webbrowser lässt sich anschließend über das Webinterface die Einstellungen der gekoppelten Mesh-Hardware ändern, sowie Nachrichten versenden und empfangen. Darüber hinaus können auf einer Karte die aktuellen Positionen anderer Teilnehmer angezeigt werden. [61]

Meshtastic-Android Die Android-App ist kostenlos Verfügbar im Android-App-Store erhältlich [62]. Mit dieser kann man ein Handy über BLE mit einem Gerät koppeln. Anschließend können Nachrichten wie in einem Messenger versendet und empfangen werden. Jedoch beschränken sich dabei Direktnachrichten auf den primären Kanal. Darüber hinaus kann ein Softwareupdate auf der Mesh-Hardware über die App eingeleitet werden. Außerdem lassen sich auf einer Karte die Positionen anderer Teilnehmer anzeigen. [63]

Meshtastic-Apple Neben der Android-App bietet Meshtastic® auch eine Softwarelösung für Apple-Geräte an. Diese Anwendung gleicht im Inhalt und Funktionalität der Android-Version. Dementsprechend wird in dieser Arbeit lediglich auf die Android-Version eingegangen. Jedoch lassen sich die getroffenen Aussagen auch auf die iOS-App übertragen. [64]

Python Als dritte Möglichkeit der Interaktion mit der Mesh-Hardware stellt sich ein Konsolenprogramm zur Auswahl. Folglich liefert das Teilprojekt eine auf der Programmiersprache Python basierte Schnittstelle. Unter `https://python.meshtastic.org/` sind alle unterstützten Befehle aufgelistet. Festzuhalten ist, dass über die Konsole mit diesem Programm sowohl die Mesh-Hardware konfiguriert, als auch mit Nachrichten interagiert werden kann. So ist es möglich Nachrichten zu senden, als auch zu empfangen. Insbesondere können über dieses Python-Tool auch Nachrichten über sekundäre Kanäle, als auch administrative Befehle an andere Teilnehmer versendet und empfangen werden. Ebenso kann dieses Werkzeug genutzt werden, um Konfigurationen von einem Gerät auszulesen. Auch über diese Schnittstelle ist es möglich Informationen über andere Netzwerkteilnehmer, wie deren Hardware-MAC-Adresse oder Empfangsqualität zu ermitteln. Darüber hinaus kann mittels des Konsolenprogramms über eine USB-Verbindung ein Firmwareupdate eingespielt werden. [65]

Meshtastic-gui-installer Um technisch unerfahrenen Nutzern das Flashen der Mesh-Hardware zu vereinfachen, wurde ein graphisches Programm erstellt. Dieses ist für diverse Linux-Derivate, aber auch für PCs ab Microsoft Windows 7 verfügbar. Das Programm verwendet den beschreibenden Namen ”Meshtastic Flasher”. [66]

Die Menüführung gestaltet sich durch große Buttons mit Nummerierung nach der Reihenfolge der Aufgabenschritte mit entsprechender Auflistungsreihenfolge sehr intuitiv. Zusätzlich wählt der Benutzer aus Drop-Down-Menüs die gewünschte Firmwareversion und das entsprechende Gerät aus. Durch die automatische Erkennung des angeschlossenen Gerätes, wie auch der bereits installierten Firmwareversion, wird dem Benutzer ein Update erleichtert. Darüber hinaus ist eine Konfiguration der Einstellungen über die graphische Oberfläche möglich. Dagegen verzichtet diese Software auf eine Funktion des Sendens und Empfangens von Nachrichten. [66]

Andere Teilprojekte Der Meshtastic®-Github-Account beinhaltet noch viele weitere Projekte, wie beispielsweise einen ESP32-HTTPS-Server [35]. Diese werden jedoch nicht detaillierter betrachtet, da es sich bei den meisten lediglich um einen Repository-Fork von den ursprünglichen Projekten handelt. Dieses Vorgehen wird üblicherweise für eine Archivierung oder die Referenzierung eines speziellen Projektstandes durchgeführt.

3 Forensische Ansätze

Das Kapitel 2 zusammenfassend ist in Abbildung 3.1 eine Übersicht der Zusammenhänge der von Meshtastic® genutzten Technologien dargestellt. Das darin abgebildete Beispielnetzwerk nutzt die vorgestellten Kommunikationsschnittstellen. Diese aufgreifend kann an allen Übertragungspunkten und -wegen eine forensische Analyse angesetzt werden.

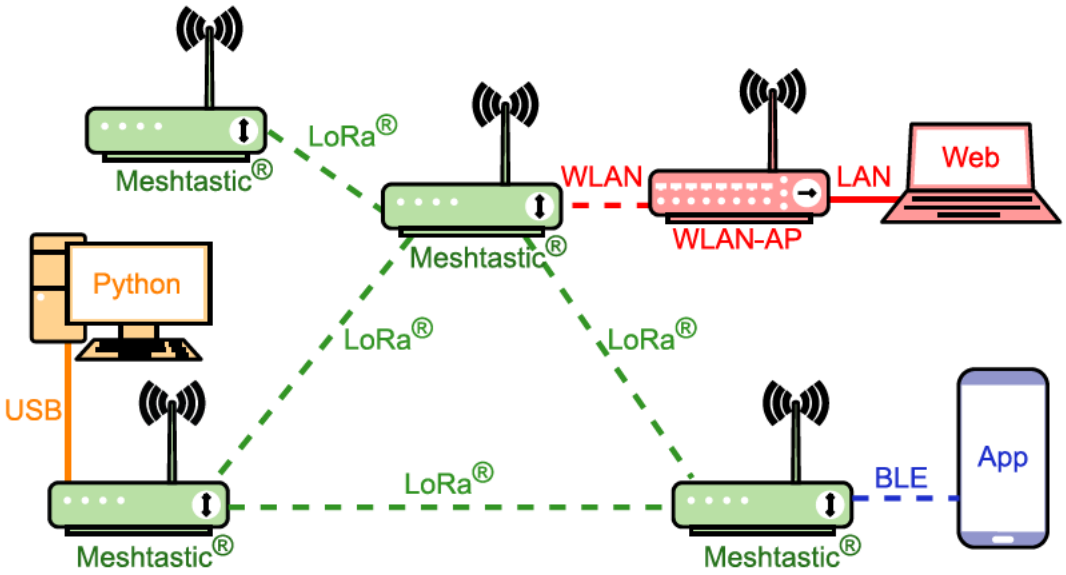


Abbildung 3.1: Meshtastic® Technologie Übersicht

Um die Analyse zu strukturieren, kann die Untersuchung basierend auf unterschiedlichen Kenntnisständen eingegliedert werden. Dabei kann die bestehende Ausgangslage in die folgenden unterschiedliche Ebenen gestaffelt werden.

1. Keine Informationskenntnis
2. Netzwerkexistenz ist bekannt
3. Netzwerkteilnehmer sind bekannt
4. Netzwerknachrichten sind bekannt

Aus dieser Erkenntnis lässt sich die Vorgehensweise einer Analyse schrittweise wie folgt gliedern.

1. Existenz des Netzwerks erkennen
2. Teilnehmer identifizieren
3. Nachrichten protokollieren
4. Nachrichten auswerten

Gleichzeitig kann an die forensische Untersuchung Anforderungen gestellt werden. Ein wichtiges Kriterium ist, dass eine Analyse des Netzwerkes nicht durch dessen Teilnehmern erkannt werden kann. Insofern die Nutzer des Netzwerkes eine Untersuchung bemerken, kann sich deren Verhalten ändern. Ferner kann es dazu führen, dass Beweise bewusst vernichtet werden. So ist es ebenfalls wichtig, dass der Betrieb des Netzwerkes nicht beeinflusst wird. Eine Störung des Netzwerkes kann zu einem Nutzungsende oder einer aufmerksameren Beobachtung hinsichtlich einer Analyse führen. Fortführend ist die Beständigkeit der erfassten Beweise ein bedeutsames Kriterium. Insofern Beweise

flüchtig sind oder manipuliert werden können, ist deren Verwertbarkeit gefährdet. Schließlich sollen unterschiedliche Vorgehensweise hinsichtlich des Aufwandes und der Erfolgswahrscheinlichkeit bewertet werden. Daraus kann eine anzustrebende Reihenfolge von durchgeführten Maßnahmen abgeleitet werden.

Unter der Betrachtung der genannten Anforderungen und aufgelisteten Arbeitsschritte wird in den folgenden Kapiteln auf forensische Ansätze eingegangen.

3.1 Netzwerkerkennung

Bevor weiterfassende forensische Analysen vorgenommen werden, sollte zunächst geklärt werden, ob überhaupt ein Mesh-Netzwerk, in diesem Fall ein Meshtastic®-Netzwerk, vorliegt. Für diesen Zweck sollte ein Zeitraum festgelegt werden, in dem eine Nachricht zwischen den Teilnehmern zu erwarten wäre, insofern ein Mesh-Netzwerk besteht. Anschließend muss für diesen Zeitraum die jeweilige Frequenz auf eine Nachricht des Mesh-Netzwerkes abgehört werden. Dafür schließen sich die drei folgenden Fragen an:

- Welche Frequenz verwendet das Mesh-Netzwerk?
- Welche Übertragungstechnik benutzt das Mesh-Netzwerk?
- Wodurch ist eine Nachricht des Mesh-Netzwerkes zu erkennen?

Frequenz Im Falle von Meshtastic® sind die im europäischen Raum zu überprüfenden Frequenzen 433 MHz und 869 MHz [67, 68]. Das folgt aus der Tatsache, dass Meshtastic® LoRa® als Protokoll für den Aufbau des Mesh-Netzwerkes benutzt [68]. Für diese beiden Frequenzen muss also ein Empfänger vorhanden sein, um in der Lage zu sein das Mesh-Netzwerk zu detektieren.

Übertragungsprotokoll Naiv könnte nun angenommen werden, dass eine Aktivität auf dem beobachteten Funkspektrum eine Existenz eines Mesh-Netzwerkes bedeutet. Jedoch benutzen auch dem Mesh-Netzwerk fremde Geräte diese Frequenz. So wird die Frequenz 868 MHz beispielsweise auch für Türklingeln verwendet [69]. Um die zu analysierenden Signale zu filtern und gleichzeitig auswertbar zu demodulieren, sollte ein Empfänger der gleichen Übertragungstechnik, wie im Mesh-Netzwerk verwendet werden. Das bedeutet im Falle von Meshtastic® die Verwendung eines LoRa®-Empfängers.

LoRa® unterstützt verschiedene Konfigurationen bezüglich Geschwindigkeit und Fehlertoleranz [67]. Diese Einstellung ergibt sich aus den Parametern Spreizfaktor, Codierungsrate und Bandbreite [67]. Damit ein empfangenes Signal korrekt dekodiert wird, müssen die gleichen Einstellungen wie beim Sender verwendet werden. Diese sind jedoch ohne die Identifikation eines Netzwerkteilnehmers unbekannt. Jedoch liefert Meshtastic® für den Benutzer sieben vordefinierte Einstellungen [68]. Die Suche nach der korrekten LoRa®-Konfiguraion kann optimiert werden, indem zuerst diese sieben sogenannten Presets überprüft werden.

Identifizierung Wie in der reinen Frequenzanalyse könnte nun auch erneut angenommen werden, dass der Empfang einer LoRa®-Nachricht die Existenz eines Meshtastic®-Netzwerk belegt. Jedoch wird LoRa® auch von anderen Geräten verwendet. So zum Beispiel bei der Nutzung von LoRaWAN [67].

Ebenso sieht sich auch Meshtastic® bei der Auswertung von empfangenen LoRa®-Nachrichten diesem Problem gegenüber. Um dem zu begegnen, wird jeder Nachricht bei Meshtastic® eine Sequenz vorangestellt [70]. Hierbei spricht man auch häufig von Magic-Bytes, Magic Pattern oder Magic Numbers [71]. Im Falle von Meshtastic® handelt es sich bei diesen um die Bytes 0x94 und 0xc3 [70]. Demnach kann auch eine Analyse von Paketen auf diesen speziellen Nachrichtenanfang beschränkt werden. Insofern eine LoRa®-Nachricht mit den Hexadecimalwerten 0x94 0xc3 beginnt, kann davon ausgegangen werden, dass es sich um eine Nachricht für einen Meshtastic®-Knoten handelt und demzufolge ein Meshtastic®-Netzwerk existiert.

Um die LoRa®-Nachrichten zu registrieren, können unterschiedliche Methoden eingesetzt werden. So können einerseits die Funksignale direkt erfasst werden. Andererseits kann ein LoRa®-Modul mit modifizierter Meshtastic-Firmware oder eine speziell für diesen Zweck entworfene Softwarelösung verwendet werden. Die anschließenden Kapitel beschreiben diese Herangehensweisen detaillierter.

3.1.1 Netzwerkerkennung mit Funkverkehranalyse

Als flexibelste Lösung bietet es sich an, den Funkverkehr im Ganzen aufzuzeichnen. Solch eine Protokollierung kann mittels eines sogenannten Software Defined Radio, kurz SDR, durchgeführt werden. Anschließend kann die Aufzeichnung hinsichtlich verschiedener Einstellungsparameter untersucht werden. Dementsprechend muss der benötigte Zeitraum einer Untersuchung nur einmalig betrachtet werden. Darüber hinaus kann eine Untersuchung im Nachhinein und an einem anderen Ort durchgeführt werden. Ebenso besteht die Möglichkeit eine Analyse mit abweichenden Parametern zu wiederholen.

Jedoch erfordert eine solche Lösung die softwareseitige Implementierung der Dekodierung. Dies kann, insofern noch nicht in einer Softwarebibliothek vorhanden, sehr zeit- und arbeitsaufwendig sein. Dementsprechend sollte im Vorfeld diese Methodik gegenüber der Alternativen genau abgewogen werden.

3.1.2 Netzwerkerkennung mit spezieller Firmware

Insofern Hardware mit einem verbauten LoRa®-Chip vorhanden ist, kann dieser für eine Signaldekodierung genutzt werden. Hierfür wird eine Firmware programmiert, die alle empfangenen LoRa®-Pakete auswertet. Auch hier lässt sich ein Meshtastic®-Paket durch den Anfang mit den Magic-Bytes 0x94 0xc3 identifizieren [70].

Die korrekte Dekodierung hängt von der gewählten LoRa®-Konfiguration ab [68]. Somit muss über möglich Einstellungen durchiteriert werden, um die aktuell verwendeten LoRa®-Einstellungen vom Netzwerk zu erkennen. Dafür muss stets für jede Konfigurationsmöglichkeit die Frequenz über den kompletten Zeitraum abgehört werden, in dem mindestens ein Paket des Mesh-Netzwerkes erwartet wird. Um die Analyse zu beschleunigen, können die Konfigurationen priorisiert nach der zu erwarteten Einstellung durchprobiert werden. So ist es bei Meshtastic® am wahrscheinlichsten, dass die Standardkonfiguration ”LongFast” oder eine andere Einstellung für eine große Reichweite verwendet wird [70]. Ebenso sind LoRa®-Konfigurationen aus den vordefinierten Vorgaben wahrscheinlicher gegenüber einer kompletten individuellen Einstellung. Sofern diese nicht zu erwarten sind, können diese zur Optimierung der Analysezeit auch vollständig ignoriert werden.

Darüber hinaus besteht die Möglichkeit, dass zwei getrennte Mesh-Netzwerke mit unterschiedlichen LoRa®-Einstellungen existieren. Folglich müssen selbst bei einem Fund eines Meshtastic®-Paketes auch die restlichen Möglichkeiten überprüft werden.

Diese Untersuchung kann in einer fortlaufenden Schleife durchgeführt werden, um die Wahrscheinlichkeit einer Detektion zu erhöhen. So kann mit ausreichender Zeit auch ein Netzwerk erkannt werden, welches die LoRa®-Einstellungen zeitlich ändert. Hierbei ist zu beachten, das eine vollständige Rotation durch alle Konfigurationen bei zusätzlicher Betrachtung von individuellen Einstellungen einen großen Zeitraum einnehmen kann. Folglich kann es sinnvoll sein, diese zwischenzeitlich zu unterbrechen und erneut die Standardvorgaben zu betrachten, ehe mit den individuellen Einstellungen fortgefahren wird.

3.1.3 Netzwerkerkennung mit modifizierter Firmware

Insofern der Quellcode vorliegt, kann die Firmware auf dem genutzten Empfänger eigenständig modifiziert werden. Damit kann die Verarbeitung empfangener Netzwerkpakete abgeändert werden, sodass lediglich deren Existenz aufgezeigt wird.

Im Falle von Meshtastic® sind die Funktionalitäten für eine Erkennung bereits implementiert. Damit werden die in Kapitel 3.1.2 aufgezeigten Herausforderungen bereits gelöst. Für eine Nutzung muss lediglich der Debugging-Modus aktiviert werden [11]. Anschließend können diese über eine serielle Verbindung am Computer ausgewertet werden.

Gleichzeitig sollte verhindert werden, dass Teilnehmer des Netzwerkes diese Analyse erkennen. Zur Vermeidung einer Detektion müssen Funktionen zum Senden von Daten deaktiviert werden. Vordefiniert existiert im Quellcode von Meshtastic® die Einstellung ”Transmit Power”, die bestimmt, mit welcher Sendeleistung eine Mesh-Hardware Signale emittieren soll. Der Wert Null wird hier jedoch nicht als Sendeunterdrückung aufgefasst. Stattdessen wird dieser durch den Standardwert der jeweiligen Funkkonfiguration, beispielsweise 27 mW bei EU_868, ersetzt. Dementsprechend muss dieser Teil im Quellcode angepasst werden. Als Alternative eignet sich die Konfigurationsoption ”Transmit Enabled” [68]. Mit dieser kann ein Senden von Nachrichten komplett unterbunden werden. [72]

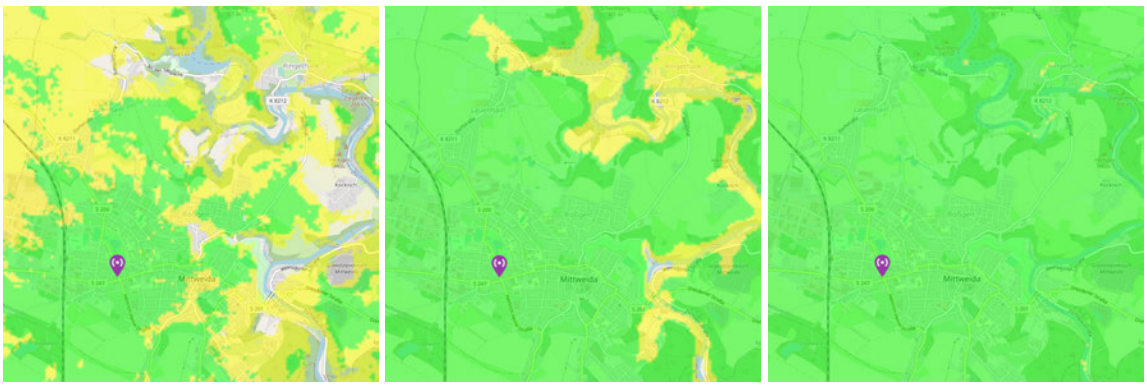
Als weitere Konfiguration ist unter der Geräterolle die Option ”CLIENT_MUTE” zu nennen. Diese deaktivert nur die Routingfunktion, aber nicht das generelle Senden von Daten. Insofern werden Informationen, wie Gerätename, Batteriestand und GPS-Positionen weiterhin an andere Netzwerkteilnehmer ohne Nutzerinteraktion mitgeteilt. Folglich sind Netzwerkknoten anschließend in der Lage diesen für sie neuen unbekannten Netzwerkteilnehmer zu detektieren und Rückschlüsse auf eine Analyse zu ziehen. Dementsprechend ist die Option ”CLIENT_MUTE” für eine stille Observation ungeeignet. [11]

Von den drei vorgestellten Methoden bringt der Debugging-Modus in Kombination mit der Sendeunterdrückung das zielführende Ergebnis. Somit ist dieser Ansatz den anderen vorzuziehen.

3.2 Teilnehmeridentifizierung

Mit dem ausgewählten Verfahren aus Kapitel 3.1 ist bestätigt, dass ein Meshtastic®-Netzwerk besteht. Im Weiteren sollen die einzelnen Teilnehmer des Netzwerkes identifiziert werden. Teilnehmer sind hier die empfangenden und sendenden Geräte im Mesh-Netzwerk. Dabei könnte sowohl eine reale Person mit dem Gerät interagieren, als auch eine autonome Mesh-Hardware, als sogenanntes Relais, am Netzwerk teilnehmen. Letzteres bedeutet, dass dieses Gerät dazu dient, den Empfang für andere Netzwerkteilnehmer zu verbessern.

Das LoRa®-Signal wird durch Hindernisse, wie einem Berg abgeschirmt. Dadurch können optisch voneinander getrennte Teilnehmer nicht direkt miteinander kommunizieren. Folglich benötigen diese einen dritten Netzwerkteilnehmer, welcher eine Funkverbindung zu den getrennten Teilnehmern aufbauen kann. Dieser sendet die LoRa®-Nachrichten beidseitig weiter. Dabei kann es je nach Entfernung und Topologie des Geländes auch nötig sein, mehrere Knoten als Relais zu verwenden. Auf Grund dieser Eigenschaft bieten sich hoch gelegene Positionen mit Sicht auf ein weitläufiges Gebiet als Standort für einen solchen Relais an. Hierfür sind insbesondere Berge, hohe Gebäude und Funkmasten logische Punkte. Ebenso ist es denkbar ein Relais an einem Fesselballon oder Schirmdrachen zu befestigen. So ist in Deutschland eine Flughöhe von 30 m bzw. 100 m gesetzlich erlaubt [73]. Dies kann, wie in der Abbildung 3.2 dargestellt, bereits zu einer starken Verbesserung des abgedeckten Gebietes führen.



(a) Höhe 1m **(b)** Höhe 30m **(c)** Höhe 60m

Abbildung 3.2: Simulation der Signalabdeckung einer Mesh-Hardware auf verschiedenen Höhen
Erstellt mit: `https://www.ve2dbe.com/rmonline_s.asp`

Aus dieser grundlegenden Betrachtung lassen sich durch andere Methoden ermittelte Standorte logisch eingrenzen und priorisiert untersuchen. Handelt es sich bei einem von zwei vermuteten Standorten um eine erhöhte Position, so steigt die Wahrscheinlichkeit, dass sich dort tatsächlich ein Teilnehmer des Netzwerkes befindet.

Insofern ein Person ihren Meshtastic®-Knoten stets bei sich trägt, kann bei längerem Aufenthalt an einem Ort dieser mit in eine Profilbildung eingebunden werden. So sind Orte mit wiederholten Aufenthalt das eigene Zuhause, der Arbeitsort, Vereine, Einkaufsmöglichkeiten und Wohnorte von Freunden sowie Bekannten. Diese lassen sich aus dem Aufenthaltszeitraum, der jeweiligen Uhrzeit und der Regelmäßigkeit ableiten.

Informationen über Teilnehmer beschränken sich nicht nur auf deren Standort. So kann in diesem Schritt auch die verwendete Hardware bedingt ermittelt werden. Beispielsweise ist das LoRa®-Funkmodul SX1278 lediglich in der Lage auf der Frequenz von 433 MHz zu kommunizieren [74]. Dieses kann folglich ausgeschlossen werden, sofern ein Meshtastic®-Netzwerk auf dem Frequenzband von 868 MHz beobachtet wird.

Weiterführend kann der Nutzungszeitraum eines Teilnehmers im Netzwerk Aufschluss über Hardware geben. Insofern ein Knoten stets Nachts ausfällt, liegt die Vermutung nahe, das dieser mittels Solar betrieben wird und zum Ausfallzeitpunkt seinen Akkuspeicher aufgebraucht hat. Im Falle eines wiederholten Ausfalles der Hardware kann versucht werden, eine Korrelation mit anderen Ereignissen herzustellen. Dafür bieten sich Windgeschwindigkeiten für eine Stromerzeugung mit Windkraft, Trockenperioden bei Stromerzeugung durch Wasserkraft aber auch Stromausfälle bei Netzbetrieb ohne Akku an.

Die größte Herausforderung bei der Identifizierung von einzelnen Teilnehmern eines Mesh-Netzwerkes ist jedoch die chaotische Natur eines solchen Netzwerkes. Insbesondere, da diese mobil sind. Daher kann es erforderlich sein, die Erkennung von Netzwerkteilnehmern kontinuierlich durchzuführen. Bei Meshtastic® kommt mit LoRa® eine Funktechnik in einem öffentlichen Spektrum zum Einsatz [8, 75]. Daher erzeugen aus der Perspektive des Analysten alle dem Mesh-Netzwerk fremde Nutzer dieser Frequenz Störsignale. Weiter können dabei gleiche Funktechniken, wie in diesem Fall der Einsatz von LoRaWAN, eine Analyse erschweren [75]. Folglich muss nach einem Empfang eines Signals dieses auf die verwendete Signal-Modulation und den Paketaufbau untersucht werden.

Gleichzeitig benötigt das genutzte Protokoll in einem Mesh-Netzwerk festgelegte Standards, um sich selbst zu organisieren. So wird in Meshtastic® der Header stets im Klartext übertragen. Dieser liefert die folgenden Informationen:

1. ID des Senders (4 Bytes)
2. ID des Ziels (4 Bytes)
3. ID des Paketes (4 Bytes)
4. Flags mit Hop-Limit und Empfangsbestätigung (1 Byte)
5. Channel-Hash (1 Byte) [76]

Daraus lässt sich eine vollständige Liste mit Teilnehmern generieren, sofern diese IDs über einen entsprechend langen Zeitraum aufgezeichnet werden. Bei der ID eines Knotens handelt es sich um die Endung der Hardware-Adresse des verwendeten LoRa®-Moduls. [76, 77]

Im folgenden sollen die Möglichkeiten der Lokalisierung der Teilnehmer genauer beleuchtet werden. Mit den bekannten Positionen der Netzwerkteilnehmern kann aus diesen ein Knoten bestimmt werden, welcher sich besonders zum Auslesen der Netzwerkkonfiguration eignet. Hierbei sollte insbesondere darauf geachtet werden, dass die Erfassung der Konfiguration von den restlichen Netzwerkteilnehmern nicht detektiert wird. In der Konfiguration enthalten sind die für die Kommunikation genutzten symmetrischen Schlüssel. Diese ermöglichen eine vollständige und unverschlüsselte Protokollierung der Nachrichten, auf welche im Kapitel 3.3 eingegangen wird. Dafür werden die Ortungsmöglichkeiten mit LoRa®, WLAN, BLE und GPS betrachtet.

3.2.1 Lokalisierung mit LoRa®

Da Meshtastic® für die Kommunikation zwischen den Mesh-Teilnehmern die Funktechnik LoRa® verwendet, bietet sich die Analyse dieser Signale auf Grund ihrer großen Reichweite an, um eine Ortung vorzunehmen [8]. Funkbasierte Ortung wertet dafür die empfangene Signalstärke an mindestens drei Standorten aus [78, 79]. Die Signalstärke eines empfangenen Signals, wird im englischen Received Signal Strength Iindication oder abgekürzt RSSI genannt. Es ist davon auszugehen, dass das Signal mit einer Signalstärke von 27 mW bei LoRa® ausgestrahlt wird, da dieses das maximal, legal erlaubte Limit darstellt [75]. Bei Funksignalen korreliert die Signalstärke indirekt mit der Entfernung [67]. Folglich kann aus einer empfangenen Signalstärke eine Entfernung zum Sender berechnet werden. Dadurch ergibt sich für jeden Empfänger ein Kreis um den Empfangsort, an dem sich der Sender befinden kann. Die entstandenen Kreise aller Empfänger überschneiden sich an genau der Position, an der sich der Sender befindet, da nur so die jeweils gemessene Entfernung erfüllt werden kann. Dieses Vorgehen wird als Lateration bezeichnet [78, 79]. Insofern exakt drei Messwerte genutzt werden, spricht man von Trilateration. Es wurde bereits gezeigt, dass eine Lokalisierung mittels LoRa® möglich ist [67]. Die in der wissenschaftlichen Veröffentlichung genannte Berechnung kann hier auf die gleiche Weise verwendet werden, da die Entfernung auf dieselbe Art ermittelt wird. Dabei sind lediglich die Rollen von Sender und Empfänger vertauscht. Dieser Perspektivenwechsel ändert die Signalstärke nicht. Die errechnete Entfernung bleibt bestehen.

Auf Grund der Eigenschaft, dass LoRa® nur auf Sichtweite kommunizieren kann, ist theoretisch eine Lokalisierung mittels Abschirmung möglich. Dafür sucht der Analyst gezielt einen Standort auf, von dem er ausgeht, dass sich der Empfang eines Signals abschirmen lässt. Dazu zählt, wie in Abbildung 3.2a beispielhaft gezeigt, eine Bergflanke. Wie auch in Abbildung 3.2b zu erkennen ist der Empfang im Tal der Zschopau deutlich reduziert. Der Empfänger bewegt sich hinter das Hindernis, bis dieses einen Empfang des Signals verhindert. Wiederholt an vielen Positionen lässt sich ein Gebiet ermitteln, bei dem eine Sichtverbindung zu allen Positionen mit vorhandenen Funkkontakt bestand, jedoch eine Abschirmung zu den Orten mit fehlendem Empfang gewährleistet ist. Diese Methode setzt voraus, dass die Positionen der Empfangsmessung überall gleichzeitig geschieht. Alternativ lässt sich diese Methode auch anwenden, wenn sich das Signal stationär verhält und bei wiederholtem Senden eindeutig demselben Sender zugeordnet werden kann. Für jedes Signal lässt sich somit aus allen Empfangsdaten eine Karte mit den Überschneidungen der Orte in Sichtreichweite erstellen. Bei Überschneidungsgebieten von besonders vielen Empfängern handelt es sich um wahrscheinlichere Aufenthaltsorte des Senders. [78]

Zuletzt kann über eine Richtfunkantenne der Empfang eines Signals räumlich eingegrenzt werden. Man spricht hier von Angulation. Unter Verwendung von zwei Empfängern an unterschiedlichen, bekannten Standorten wird eine Triangulierung des Signals ermöglicht. Die Methode ähnelt sehr der Trilateration, nur dass hier mit Winkeln, statt Entfernungen die Position bestimmt wird. Jedoch kann insofern von der bekannten Sendestärke von 27 mW ausgegangen wird, auch mit nur einem Empfänger die Position ermittelt werden [75]. Hier liefert die empfangene Signalstärke eine Abschätzung der Entfernung [67]. Vergleichbar ist dies mit der Erstellung eines Radarbildes im Flugraum durch eine rotierende Antenne. Vorteilhaft ist dabei zu nennen, dass für die komplette Analyse nur ein Standort vonnöten ist. Jedoch setzt diese Methode voraus, dass ein Signal lange genug besteht, bis mit der Antenne alle Richtungen abgetastet wurden. Da im Gegensatz zu Radar kein reflektierendes Signal ausgestrahlt wird, sondern nur empfangen wird, ist damit auch nur der Sendezeitraum für

einen Empfang nutzbar. Erschwerend kommt hinzu, dass die Signale bei Meshtastic® zeitlich kurz und in großen Abständen ausgestrahlt werden. Eine Lokalisierung ist dennoch möglich, indem radial versetzte Antennen genutzt werden, um gleichzeitig mehrere Richtungen abzutasten. [78]

3.2.2 Lokalisierung mit BLE

Auf der Frequenz 2,4 GHz kann eine Verbindung zur Mesh-Hardware über BLE hergestellt werden [63]. Die Signale dieser kabellosen Kommunikation können von allen Empfängern im Umkreis empfangen werden. Jedoch ist die Rechweite auf etwa 60 Meter begrenzt, was eine Detektion erschwert [80, 81]. Dementsprechend kann eine ausschließlich auf BLE basierende Ortung nur mit großem Aufwand realisiert werden. Dafür ist es notwendig mit einem Programm, wie Kismet, das Zielgebiet abzufahren [82]. Dabei fordert die geringe Reichweite ein sehr feines Abtastraster mit einem Punktabstand von wenigen Metern. Das ist auf Grund von privaten Grundstücken praktisch nicht umsetzbar. Dementsprechend sollte eine Ortung mit BLE unterstützend zu einer Ortung mit anderen Techniken, wie beispielsweise LoRa® durchgeführt werden.

Der sichtbare Name des Bluetooth-Gerätes setzt sich aus dem Gerätenamenkürzel, auch Owner-Short genannt, und den letzten zwei Bytes der Hardware-Adresse zusammen. Getrennt werden beide Namensteile durch einen Bindestrich. Insofern der Gerätenamen bekannt ist, lässt sich Mesh-Hardware von anderen Bluetooth-Geräten unterscheiden. [83]

3.2.3 Lokalisierung mit WLAN

Alternativ zu BLE können ESP32-Mikrokontroller auf der gleichen Frequenz von 2,4 GHz über IEEE 802.11 b/g/n, häufig auch als WLAN bezeichnet, kommunizieren [17]. Diese Möglichkeit greift Meshtastic® auf und bietet dementsprechend zu einer Verknüpfung mit einem Endgerät eine WLAN-Verbindung an [84]. Hierbei verbindet sich die Mash-Hardware zu einem Access-Point. Mit diesem verbindet sich anschließend das Endgerät des Nutzers.

Grundsätzlich lässt sich festhalten, dass Meshtastic® für die Konfiguration keinen Namen oder Passwort für den kabellosen Netzwerkzugang festlegt [85]. Es obliegt folglich dem Nutzer eine SSID und einen PSK zu definieren [85]. Somit lassen sich diese WLAN-APs nicht von anderen Access-Points anhand dieser Merkmale differenzieren. Ferner bietet Espressif mit der WiFi-Bibliothek über die Methode `esp_wifi_set_mac` eine Möglichkeit die BSSID im zu verändern [86]. Dementsprechend kann ein ESP32 nicht an der verwendeten MAC-Adresse identifiziert werden. Folglich kann mit der ausschließlichen Analyse von WLAN nicht sichergestellt werden, dass es sich um eine Mesh-Hardware mit der Meshtastic®-Firmware handelt.

ESP-Mikrokontroller unterstützen sehr geringe Sendeleistungen, um den Stromverbrauch zu reduzieren. Dementsprechend ist ein Empfang des Signals nur auf wenigen Metern zu erwarten. Grundsätzlich kann davon ausgegangen werden, dass WLAN eine maximale Reichweite von 100 m erreicht [81]. Somit kann die Ortung eines Mesh-Teilnehmers, wie bei BLE, in einem vermuteten Zielbereich eingegrenzt, aber nicht aus beliebiger Entfernung ermittelt werden.

Als großflächiger Ansatz kann darüber hinaus das Kartographieren von WLAN-APs in einem Gebiet genannt werden. Dabei wird das Zielgebiet systematisch abgetastet und dabei jeder empfangene WLAN-AP mit der entsprechenden aktuellen GPS-Position verknüpft. Umgangssprachlich wird hier von ”Wardriving” gesprochen [82]. Für eine solche Durchführung eignet sich beispielsweise das quelloffene Programm ”kismet” [82, 87]. Unter der Annahme, dass die BSSID unverändert genutzt wird, kann der Sender als ein ESP32-Endgerät identifiziert werden. Somit können die gewonnenen Daten auf ESP-WLAN-Clients gefiltert werden. Diese möglichen Standorte können darauffolgend mit anderen Ortungsergebnissen verknüpft werden. Insofern eine Überschneidung festgestellt wird, handelt es sich mit höherer Wahrscheinlichkeit um einen Standort einer Mesh-Hardware. Ersichtlich ist jedoch, dass bei der Anwendung dieser Methode keine Echtzeitauswertung stattfindet.

Neben IEEE 802.11 b/g/n unterstützen ESP32 das von Espressif entwickelte Zusatzprotokoll LR [16]. Dieses erreicht eine Reichweite von bis zu einem Kilometer, kann jedoch nur von ESP-Mikrokontroller dekodiert werden [16]. In Meshtastic® wird WLAN dazu genutzt die Mesh-Hardware mit einem Endgerät zu verknüpfen. Folglich wird das Zusatzprotokoll LR nicht verwendet und wird fortlaufend nicht betrachtet.

3.2.4 Lokalisierung mit GPS

Ein Endgerät sendet bei der Lokalisierung der eigenen Position mittels GPS keine Signale aus. GPS arbeitet somit passiv [88]. Folglich kann ein Knoten des Mesh-Netzwerkes nicht über Funknachrichten im Zusammenhang mit GPS geortet werden. Jedoch ist eine Kernfunktion von Meshtastic® das Übertragen der eigenen Position an andere Mesh-Teilnehmer [8, 10]. Demzufolge können GPS-Koordinaten in den entsprechenden Nachrichten enthalten sein. Zu beachten ist jedoch, dass Meshtastic® Positionsangaben nur auf dem primären Kanal versendet.

Ein Netzwerkknoten kann seine Position durch verschiedene Methoden ermittelt werden. Im Folgenden sind die Quellen einer Positionsangabe aufgezeigt.

GPS-Modul Insofern ein GPS-Modul auf dem verwendeten Board verbaut wurde, kann dieses genutzt werden, um die eigene Position zu ermitteln. In den Protobuf-Definitionen wird diese Art auch als ”internal” bezeichnet [76]. Die Genauigkeit ergibt sich aus dem verwendeten GPS-Empfänger. Darüber hinaus kann es insbesondere in Gebäuden zu Empfangsschwierigkeiten kommen. Außerdem wird beim Beginn des Betriebs, noch vor der ersten Positionsermittlung, eine Aktualisierung des Almanachs und damit zusätzliche Zeit benötigt [22, 89].

Android-App Meshtastic® kann mit einem Android-Telefon verknüpft werden [83]. Dafür kann aus dem Appstore die Meshtastic®-App installiert werden [90]. Anschließend kann die auf dem Handy ermittelte Position an die Mesh-Hardware übertragen werden. Diese teilt die Position des Telefons mit anderen Netzwerkteilnehmern [91]. Eine mit dieser Methode ermittelte Position wird als Typ ”external” bezeichnet [76].

Die Position wird unter Android nicht ausschließlich basierend auf GPS ermittelt. In die Lokalisation werden ebenfalls WLAN-APs und Mobilfunkmasten einbezogen. Auf diese Weise kann eine genauere Position als mit dem in der Mesh-Hardware integrierten GPS-Modul erzielt werden. Insbesondere wird in Innenräumen eine genauere Position bestimmt. [92]

Statische Position Als dritte Möglichkeit kann in der Konfiguration der Mesh-Hardware eine statische Position eingetragen werden [10]. Diese Art der Positionsangabe ist beispielsweise für Relays mit einem festen Standort geeignet. So muss für diesen keinen Mesh-Hardware mit einem GPS-Empfänger verwendet werden. In den Netzwerknachrichten wird diese Positionsangabe mit ”manuell” markiert [76]. Abseits von der Angabe als ”manuell” lässt sich auch durch die stets exakt gleichen Koordinaten auf eine statische Positionsangabe schließen. Dagegen ist bei einer stationären Mesh-Hardware mit Positionsermittlung über ein GPS-Modul eine ”Bewegung” im Rahmen der Messungenauigkeit zu verzeichnen.

Festzuhalten ist jedoch, dass diese Position nicht der Realität entsprechen muss. Dementsprechend muss diese Angabe durch Beobachtungen vor Ort oder einem Abgleich über andere Lokalisierungsmethoden, wie LoRa® vorgenommen werden.

3.3 Nachrichtenprotokollierung

Nachdem der Betrieb eines Netzwerkes und deren Teilnehmer erkannt wurde, schließt sich die Erfassung der übertragenden Nachrichten an. Auch in diesem Schritt sollte gewährleistet werden, dass durch die Analyse selbst kein Signal ausgestrahlt wird, um unerkannt zu bleiben. Die Erfassung der Nachrichten lässt sich auf verschiedenen Ebenen der Kommunikation realisieren. Diese werden in den folgenden Kapiteln erläutert.

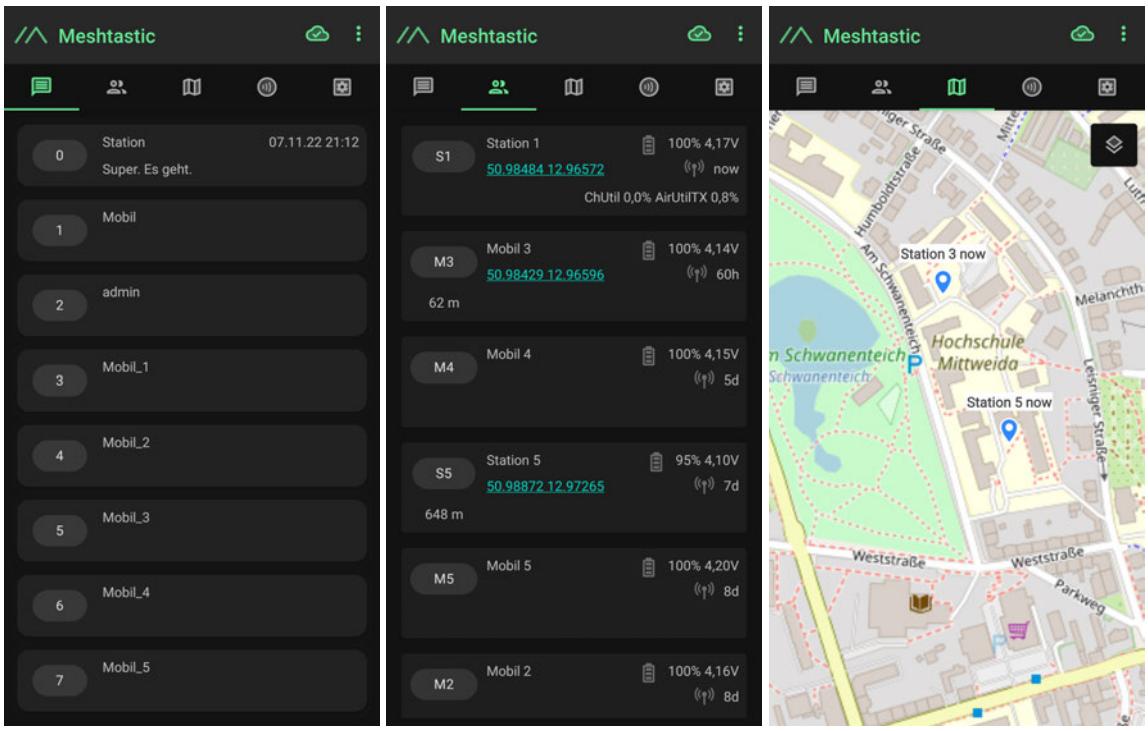
3.3.1 Nachrichten der Android-App

Die Android-App wird genutzt, um benutzerfreundlich mit dem Mesh-Netzwerk zu interagieren. Dafür wird das Mobiltelefon mit der Mesh-Hardware über BLE gekoppelt [83]. Anschließend kann die App genutzt werden, um Informationen über das Netzwerk selbst zu erhalten, in Chatform Nachrichten auszutauschen und Positionen von anderen Mesh-Geräten anzuzeigen. Dabei lassen sich die folgenden Informationen von den Reitern der App auslesen.

Chats Der erste Reiter listet alle bekannten Chats auf. Dabei erhalten standardmäßig alle auf der Mesh-Hardware eingestellten Kanäle in dieser Liste einen Eintrag. Namentlich aufgelistet werden die Chats der eingestellten Kanäle mit dem Index in der Konfiguration. Zusätzlich werden direkte Chats auf dem primären Kanal als eigenständiger Chat aufgelistet. Dieser erhält daraufhin den Namenskürzel des Zielgerätes. So stellt beispielsweise ”0” den primären Kanal und ”M1” den direkten Kanal zu ”Mobil 1” dar. Die Chatnamen werden in einer optisch hervorgehobenen Blase abgebildet. Daneben sind zweizeilig die Informationen über den ausgeschriebenen Kanal bzw. Gerätenamen, ein Zeitstempel der letzten Textnachricht und in der unteren Zeile der Text der letzten Nachricht. Eine Bildschirmaufnahme kann in Abbildung 3.3a nachvollzogen werden. [12]

Nachdem ein Listeneintrag ausgewählt wurde, wird ein Chat im smartphone-üblichen Format angezeigt. Folglich befinden sich rechtsbündig die eigenen Nachrichten und linksbündig die Nachrichten der Gesprächspartner. Ebenso signalisiert ein Haken unter der Nachricht, ob diese erfolgreich versendet wurde. Zusätzlich enthält der Eintrag den minutengenauen Versand- oder Empfangszeitpunkt. [12]

Festzuhalten ist, dass die Textnachrichten im Chatverlauf dauerhaft gespeichert werden. Dementsprechend können hier auch rückwirkend Nachrichten ausgewertet werden. Jedoch lassen sich einzelne Nachrichten, aber auch ganze Chats, jederzeit vollständig löschen. Dabei wird die entsprechende Nachricht in der SQLite-Datenbank als gelöscht markiert. Insofern diese nachfolgend nicht überschrieben wurde, kann mit einer forensischen Analyse der Datenbank die Textnachricht wiederhergestellt werden. [12, 93]



(a) Chatnachrichten **(b)** Teilnehmerübersicht **(c)** Karte mit Positionen

Abbildung 3.3: Screenshots der Android-App-Reiter 1-3

Netzwerkteilnehmer Der zweite Reiter der Android-App gibt Aufschluss über die Teilnehmer im Netzwerk. Dabei bildet jeder Listeneintrag einen Netzwerkteilnehmer ab. Wie auch bei den Chats ist in der hervorgehobenen Blase auf der linken Seite das Namenskürzel eingetragen. Mit einem Klick auf diesen, lässt sich die Mesh-Hardware aus der Ferne Neustarten, Herunterfahren, Zurücksetzen oder eine Speicherleerung anstoßen, sofern ein ”admin”-Kanal zu diesem Gerät eingerichtet wurde [94]. Wie in Abbildung 3.3b zu erkennen, werden daneben in Textform Zusatzinformationen über die Mesh-Hardware angezeigt. Links stehen der vollständige Name und die letzte bekannte Position. Dagegen wird rechtsbündig der aktuelle Akkustand in Prozent und Spannung, die Zeitdauer, seit dem letzten Kontakt, die Empfangsstärke und das Signal-Rausch-Verhältnis angezeigt. Bei mit dem Mobiltelefon gekoppelten Gerät wird statt der Empfangsstärke die Auslastung des Funkkanals angezeigt. [77]

Karte Der mittlere Reiter der Android-App zeigt eine Karte mit den Positionen der Teilnehmer an. Einzelne Teilnehmer werden optisch durch einen tropfenförmigen Pin auf der Karte eingezeichnet. Außerdem wird schriftlich über diesem Marker der vollständige Name der Mesh-Hardware und das Alter der Position vermerkt. In Abbildung 3.3c ist hiervon eine Momentaufnahme dargestellt. Folglich kann aus der Karte nur der letzte bekannte Standort abgeleitet werden. Ein Verlauf und somit eine

Bewegungsrichtung kann nur durch zukünftige Aktualisierungen ermittelt werden. Desweiteren lässt sich der Positionsmarker antippen, wodurch die GPS-Koordinaten und der Batteriestand der Mesh-Hardware angezeigt wird. [91]

Darüber hinaus lässt sich die Darstellung der Karte im Hintergrund verändern. Im aktuellen Stand (29.11.2022) stehen die Kartenanbieter OpenStreetMap, USGS TOPO, Open TOPO, ESRI World TOPO, USGS Satellite und ESRI World Overview zur Auswahl. Dementsprechend kann zwischen schematischen, topologischen und Satellitenkarten gewählt werden. Für eine Änderung der angezeigten Karte wird auf das Ebenen-Symbol in der rechten oberen Ecke geklickt. [91]

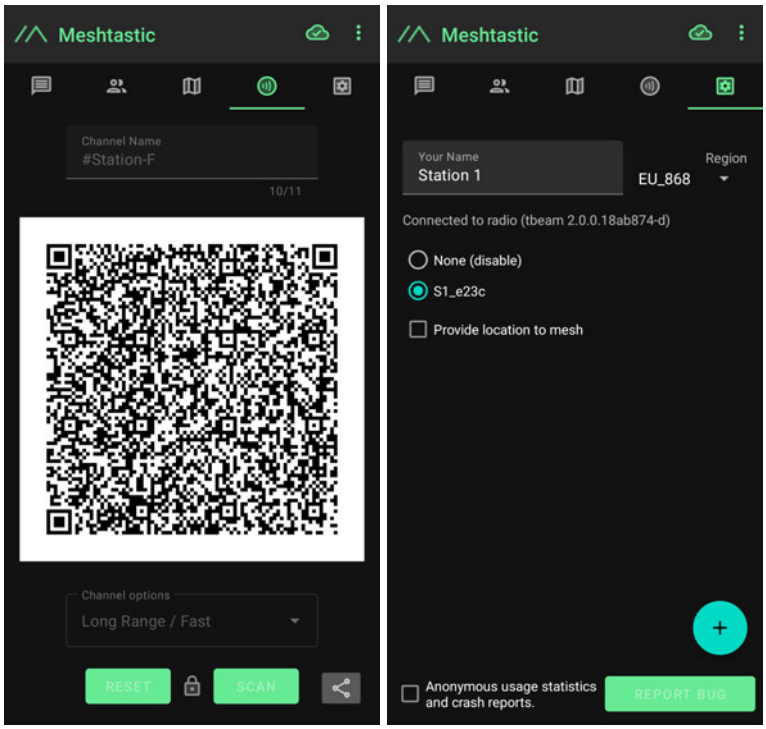
Somit wird für eine kontinuierliche Protokollierung der Position über diese Schnittstelle eine manuelle Erfassung notwendig. Dadurch eignet sich diese Datenerfassung eher nicht für eine langfristige Positionsprotokollierung.

Funkkonfiguration Der vorletzte Reiter ermöglicht es die Kanalkonfiguration der eigenen Mesh-Hardware auf eine andere zu übertragen. Für diesen Informationsaustausch wird ein QR-Code angezeigt. Beispielhaft ist hierfür ein Screenshot in Abbildung 3.4a hinterlegt. In diesen ist eine URL codiert, welche die Kanalkonfiguration enthält. Somit lässt sich die Konfiguration von einem Zweitgerät mit der Kamera scannen und damit auf ein anderes Gerät über die App übertragen. Alternativ kann die URL genutzt werden, um die Einstellungen über einen Meshtastic®-Webserver einzuspielen. [95]

Gleichzeitig kann aus dem QR-Code für Analysezwecke die Kanal-Konfigurationen entnommen werden. Insbesondere kann diese Extraktion auch vorgenommen werden, wenn die App nicht mit einer Mesh-Hardware verbunden ist. Die App speichert die Konfiguration des letzten gekoppelten Gerätes lokal. Dadurch ist es möglich, diese Einstellungen auch nachträglich zu erfassen. Insbesondere zählt dazu der AES-Schlüssel, wodurch die verschlüsselte Kommunikation dechiffrierbar wird. [95]

Gekoppelte Geräte Der letzte Reiter bietet eine Übersicht aller mit dem Android Gerät gekoppelten Mesh-Hardware. Diese werden durch ihre Hardwareadresse repräsentiert. Ferner lassen sich auf dieser Ansicht weitere Geräte mit der App verknüpfen. Jedoch kann stets nur ein Gerät als das aktiv genutzte ausgewählt werden. Eine Koppelung mit einem neuen Gerät erfolgt über die standardmäßige Bluetooth-Koppelung unter Android. Dementsprechend lässt sich unter diesem Reiter nur erfassen, mit welchen Gerät ein Mobiltelefon verknüpft wurde. Abbildung 3.4b zeigt beispielhaft eine Bildschirmaufnahme dieses Reiters. [83]

Ergänzend ist zu erwähnen, dass die drei Punkte in der rechten oberen Ecke von allen Reitern aus auswählbar sind. Darunter lässt sich ein Debug-Log aufrufen, in welchem die letzten 500 empfangen Pakete protokolliert sind. Diese entsprechen je nach Größe des Netzwerks einem unterschiedlichen Zeitrahmen. Im Falle eines Netzwerkes mit 10 Knoten umfassen diese Logeinträge etwa 30-60 Minuten. Je mobiler die Netzwerkteilnehmer sind, desto häufiger wird eine Aktualisierung der Position übertragen. Folglich verringert sich damit der erfasste Zeitraum innerhalb der 500 Log-Einträge. Zusätzlich beinhalten die Nachrichten Informationen über Telemetrie und der Mesh-Hardware anderer Knoten selbst. Diese Informationen lassen sich innerhalb der Android-App nur an dieser Stelle erfassen. [96]



(a) Kanalkonfiguration **(b)** Gekoppelte Geräte

Abbildung 3.4: Screenshots der Android-App-Reiter 4-5

Schließlich sind für eine vollständige Ausführung im Anhang A die Bildschirmaufnahmen der iOS-Anwendung beigefügt.

3.3.2 Nachrichten der Mesh-Hardware

Die Mesh-Hardware selbst empfängt und sendet die Nachrichten des Netzwerks. Hierbei arbeitet Meshtastic® nach dem Prinzip des ”Store and Forward”, zu deutsch Speichern und Weiterleiten [97]. Dabei wird eine Nachricht zunächst vollständig empfangen, ausgewertet und anschließend falls nötig weitergeleitet. Dementsprechend lassen sich direkt von der Hardware Nachrichten des Mesh-Netzwerkes auslesen. Dafür genügt es eine Mesh-Hardware mit einem Mobiltelefon zu verbinden. Dieses erhält anschließend alle intern gespeicherten Nachrichten.

Fortführend kann die Mesh-Hardware optional ein Bildschirm beinhalten. In diesem Fall wird dieser genutzt, um dem Nutzer Informationen anzuzeigen. Die Anzeige teilt sich dabei in unterschiedliche Seiten, die mit der Funktionstaste des Gerätes durchrotiert werden können.

Die erste Seite beinhaltet die letzte empfangene Textnachricht. Wie in Abbildung 2.7c zu erkennen ist, wird der Sender durch sein Namenskürzel und der Nachrichtentext selbst dargestellt. Daraufhin folgt je eine Seite für jeden Netzwerkteilnehmer. Auf dieser wird der vollständige Name, die Entfernung und Richtung des Knotens zum eigenen Standort, die Empfangsqualität und die Zeit seit dem letzten Kontakt präsentiert. Ein Beispiel solch einer Seite ist in Abbildung 2.7b abgebildet. Auf den letzten beiden Seiten, zu sehen in Abbildung 2.7d und 2.7e, lassen sich Informationen über das Netzwerk als Ganzes und das eigene Gerät erfahren. Dazu zählen die verwendete Funkeinstellung von LoRa®,

Teilnehmeranzahl, sichtbare GPS-Satelliten, eigene Koordinaten und Höhe, Batteriestand, Hardwareadressendung und Kanalbelegung. Außerdem werden Teilnehmer aufgelistet, die dem Netzwerk beigetreten sind oder es verlassen haben. [98]

Folglich liefern die Informationen auf der Mesh-Hardware selbst einen kleinen Einblick in das Netzwerk, eignen sich jedoch nicht für eine forensische, fortlaufende Protokollierung von Nachrichten. Jedoch kann mit der Mesh-Hardware über USB eine serielle Verbindung zu einem Computer aufgebaut werden. Über diese kann die vollständige Konfiguration ausgelesen werden. Damit können weitere Geräte in die Lage versetzt werden, ebenfalls Nachrichten zu empfangen und zu entschlüsseln. Außerdem können über die USB-Schnittstelle alle empfangenen Nachrichten am Computer protokolliert werden. Dementsprechend eignet sich diese Methodik hervorragend, um Nachrichten des Netzwerks dauerhaft zu erfassen. [99]

3.3.3 Nachrichten der LoRa®-Übertragung

Eine weitere Möglichkeit der Datenerfassung ist während der Datenübertragung über LoRa® selbst. Die Funksignale können unbemerkt abgegriffen und dekodiert werden. Jedoch beinhaltet die Übertragung zuvor verschlüsselte Nachrichten [9]. Lediglich der Header mit Sender, Empfänger und Hop-Anzahl werden unverschlüsselt übertragen [76]. Dementsprechend wird für eine sinnvolle Verwendung der gewonnenen Informationen der symmetrische AES-Schlüssel benötigt. Meshtastic bietet auch die Möglichkeit die Nachrichten unverschlüsselt auszutauschen [100]. Dafür muss sich jedoch der Nutzer bewusst entscheiden. Selbst in der Standardkonfiguration wird eine Verschlüsselung verwendet [100].

Insofern die Nachrichten entschlüsselt werden können, lassen sich diese in verschiedene Arten kategorisieren. Meshtastic sendet neben den Textnachrichten auch weitere Informationen an andere Teilnehmer. Diese werden durch Kennziffern, genannt Portnummern, unterschieden. Die 512 Portnummern gliedern sich in 0-63 für Kernfunktionen von Meshtastic, 64-127 für registrierte Anwendungen von Drittanbietern und 256-511 für private, individuelle Zwecke [101]. In der Version 2.0 gliedern sich die bekannten Portnummern, wie in Tabelle 3.1 aufgelistet, auf. Jedes Datenpaket beinhaltet darüber hinaus ein Payload mit den entsprechenden Inhalten. Im Folgenden werden die relevantesten dieser Paket-Arten erläutert.

Port	Anwendung	Kurzbeschreibung
0	UNKNOWN_APP	Platzhalter, keine Funktion
1	TEXT_MESSAGE_APP	Textnachricht
2	REMOTE_HARDWARE_APP	Nachricht an GPIO-Pins
3	POSITION_APP	Position
4	NODEINFO_APP	Name und Hardware
5	ROUTING_APP	Routendetektion und -fehler
6	ADMIN_APP	Administration über Mesh-Netzwerk
7	TEXT_MESSAGE_COMPRESSED_APP	Textnachricht komprimiert
8	WAYPOINT_APP	Wegpunkt oder Markierung
9	AUDIO_APP	Audio-Daten im codec2-Format
32	REPLY_APP	Ping über Mesh-Netzwerk
33	IP_TUNNEL_APP	IP-Pakete über das Mesh-Netzwerk
64	SERIAL_APP	Daten von und zu seriellen Ports anderer Knoten
65	STORE_FORWARD_APP	Reserviert, in Entwicklung
66	RANGE_TEST_APP	Entfernungsmessung
67	TELEMETRY_APP	Telemetriedaten
68	ZPS_APP	Ortung durch LoRa®
69	SIMULATOR_APP	Linux-Simulation
257	ATAK_FORWARDER	Anbindung an ATAK-App

Tabelle 3.1: Portnummern der Mesh-Nachrichten [101]

Routing Zur zuverlässigen Datenübertragung in einem Mesh-Netzwerk müssen sich die gewählten Routen dynamisch den mobilen Knoten anpassen. Für diesen Zweck verwendet Meshtastic® eine Implementierung des RFC 4728. Dieses Routingprotokoll spezialisiert sich auf mobile Ad-Hoc-Netzwerke mit einer Teilnehmeranzahl von bis zu 200 [102]. Hierbei wird gegenüber anderen Routingprotokollen, wie beispielsweise RIP oder OSPF, keine Routingtabelle aufgebaut. Vielmehr werden Nachrichten impulsiv weitergeleitet. [97]

Überträgt ein Mesh-Teilnehmer eine Nachricht, so wird diese von allen Knoten in Reichweite empfangen. Insofern die Nachricht bereits empfangen wurde, wird diese ignoriert. Andernfalls bestimmt die Empfangsstärke des Signals, wie lange ein Mesh-Knoten zu warten hat, bis dieser die Nachricht erneut aussendet. Es kann davon ausgegangen werden, dass weiter entfernte Mesh-Knoten eine Nachricht mit geringerer Signalstärke empfangen. Dementsprechend übertragen diese die Nachricht vor den näher liegenden Knoten, welche anschließend auf ein erneutes Senden verzichten. Folglich wird eine Auslastung des Funkkanals reduziert. Ferner verhindert das Erhalten einer Empfangsbestätigung ein Weiterleiten. Die Nachricht hat den Empfänger erreicht. Ein weiteres Routen dieses Pakets ist unnötig. Ergänzen lässt sich die Sendepriorität durch die zugewiesenen Geräterollen beeinflussen. So wird eine Mesh-Hardware in der ”ROUTER”-Geräterolle ein Paket vor einem ”CLIENT” erneut ausstrahlen [27]. [97, 102]

Durch dieses dynamische Routingprotokoll entfällt die Notwendigkeit Routen im Netzwerk mitzuteilen. Somit wird dieser Port genutzt, um Fehlermeldungen im Zusammenhang der Übertragung zu versenden. Beispielsweise, wenn das Hop-Limit erreicht wird. [76]

Text Message Textnachrichten stellen eine Kernfunktion des Meshtastic®-Netzwerkes dar [8]. Diese sind vergleichbar mit SMS-Kurznachrichten im Mobilfunk. Als Kodierung verwendet Meshtastic® UTF-8, was vor allem bei Nutzung von lateinischen Buchstaben geringe Datenmengen erzeugt. Dennoch ermöglicht diese Kodierung die Verwendung von anderen Alphabeten und Emojis. [101]

Darüber hinaus bietet Meshtastic® für längere Nachrichten die Möglichkeit der Kompression an. In diesem Fall werden die dem TEXT_MESSAGE_APP-Format entsprechenden Daten komprimiert und anschließend in einem TEXT_MESSAGE_COMPRESSED_APP-Paket verschachtelt übertragen. Dabei kommt als Kompressionsalgorithmus Unishox2 zur Anwendung. Folglich lassen sich selbst bei geringen Textlängen deutscher Sprache Kompressionsraten von bis zu 50% erzielen [103]. [101]

Meshtastic®-Pakete besitzen eine maximale Länge von 255 Bytes. Dementsprechend ist eine Textnachricht mit lateinischen Buchstaben auf etwa 200 Zeichen begrenzt. Unter Verwendung der Kompression kann die Textlänge einer Nachricht je nach erreichter Kompressionrate erhöht werden. Jedoch beschränken die Anwendungen die Textlänge künstlich. Beispielsweise lässt die Android-App nur Nachrichten mit bis zu 228 Zeichen zu [104].

Position Positionen werden über das Netzwerk mittels des Position-Pakets anderen Netzwerkteilnehmer mitgeteilt. Dabei enthält die Positionsangabe eine GPS-Koordinate mit einer Präzision von 10^{-7}. Ferner können die GPS-Koordinaten durch eine Höhe ergänzt werden. Auch hier unterscheidet Meshtastic® zwischen interner, externer und manueller, aber darüber hinaus auch barometrischer Quelle. Intern bezieht sich dabei auf die Höheninformation, die von dem direkt verbauten GPS-Empfänger ermittelt wurde. Als eine externe Quelle zählt auch hier ein angeschlossenes Gerät, wie ein Laptop oder Handy. Gleichbleibend beschreibt auch hier ein manueller Wert eine fixe, händisch eingetragene Angabe. Weiterführend bietet Meshtastic® die Möglichkeit an, die Höhe mittels eines barometrischen Sensors zu bestimmen. Dieser Messwert wird innerhalb von Meshtastic® als barometrisch markiert. [76, 101]

Zuletzt bietet ein Paket des Typs Position die Möglichkeit eine Geschwindigkeit und Bewegungsrichtung anzugeben. Dabei wird die Geschwindigkeit in m/s und die Bewegungsrichtung in hundertstel Grad erfasst. Dementsprechend bedeutet der Wert Null eine Bewegung gegen Norden und 27.000 eine westliche Ausrichtung. [76]

NodeInfo Informationen über eine Mesh-Hardware selbst werden in einem NodeInfo-Paket versendet. Somit enthält dieses Paket Angaben über den gewählten Namen, das Namenskürzel und die genutzte Hardware. Hierfür wird die entsprechende Hardware-Variante als Zahl kodiert. Folglich lassen sich darüber nur öffentlich bekannte Hardwaremodelle identifizieren. [76]

Telemetry Bis zur stabilen Version 1.2.95 wurde der Batteriestand im Positions-Paket mit übertragen. Durch die Unterstützung weiterer Sensoren wurden diese Messwerte in ein eigenes Paket überführt. Aktuell zählt Meshtastic® den Batteriestand, eine Spannung, eine Temperatur, einen barometrischer Luftdruck, eine relative Luftfeuchtigkeit und eine Gas-Konzentration zu den unterstützten Messwerten. Ergänzt wird die Liste der Messwerte durch die Kanalauslastung der verwendeten LoRa®-Frequenz. [105]

Für die Erfassung der Messwerte wurden in einigen Hardwaremodellen ab Werk bereits Sensoren verbaut. Dazu zählt beispielsweise das Modell LILYGO T-Echo, in welchem der Sensor BME820 Temperatur, Luftdruck und relative Luftfeuchte erfasst [33].

Waypoint Die Option der Teilung von Wegpunkten wurde mit der Version 2.0 Meshtastic® hinzugefügt. Hierbei definiert sich ein Wegpunkt aus einer GPS-Koordinate, einem Namen, einer Beschreibung und einem Ablaufdatum. Mit dem Ablaufdatum wird ein Zeitpunkt in Epoch-Zeit angegeben, zu dem der Wegpunkt automatisiert gelöscht werden kann. Epoch-Zeit gibt eine Sekundendifferenz zum festen Datum 01.01.1970 um 0:00 Uhr in UTC an. Ferner besteht die Möglichkeit einen Wegpunkt zu sperren, wodurch nur der Ersteller des Punktes diesen in Zukunft verändern kann. [76]

Admin Über den ”admin”-Kanal kann Mesh-Hardware durch Nutzung des Mesh-Netzwerkes selbst aus der Ferne konfiguriert werden. Insofern kommen für jene Anweisungen der Konfigurationspakete vom Typ Admin zum Einsatz. Der Inhalt dieser leitet sich folglich aus den konfigurierbaren Einstellungen ab. [106]

3.3.4 Nachrichten der BLE-Übertragung

Während des initialen Verbindungsaufbaus erzeugt das zu koppelnde Gerät eine achtstellige PIN, welche auf dem Display angezeigt wird [98]. Hierbei spricht man auch von einem Temporary-Key. Anschließend tippt der Benutzer die angezeigte Zahl in seinem Endgerät ein. Daraufhin leiten beide Geräte einen symmetrischen Langzeitschlüssel ab, welcher auch als Long-Term-Key bezeichnet wird. Mittels diesem Schlüssel wird darauf folgend die gesamte Kommunikation durch AES-CCM kryptographisch gesichert. Der darin enthaltene Datenstrom ist unter Meshtastic® unverschlüsselt. [107]

Dementsprechend lässt sich die Kommunikation nur erfassen, insofern der Langzeitschlüssel bekannt ist. Bis zur BLE-Version 4.2 existierte eine Schwachstelle, welche es ermöglichte den Verbindungsaufbau zu brechen. Für diesen Zweck kann das von Mike Ryan entwickelte Programm ”crackle” eingesetzt werden [107]. Diese Schwachstelle wurde durch die Einführung von Low-Energy-Secure-Connections in der Version 4.2 geschlossen. Folglich kann aus einem Mitschnitt des Verbindungsaufbaus nur mittels Brute-Force der Langzeitschlüssel abgeleitet werden. [108]

Die von Meshtastic® eingesetzten ESP32 verwenden bereits BLE in der Version 4.2 [17]. Da Nachrichten einer BLE-Verbindung zwar erkannt, aber nicht praktisch entschlüsselt werden können, eignet sich diese Schnittstelle zur Datenerfassung nicht.

3.3.5 Nachrichten der WLAN-Übertragung

Alternativ zu BLE kann WLAN als Kommunikationsmethode zwischen der Mesh-Hardware und dem Endgerät des Benutzers genutzt werden. Hierbei verbindet sich die Mesh-Hardware zu einem festgelegten WLAN-AP und stellt Webserver zur Verfügung. Für eine erfolgreiche Verbindung wird ein PSK benötigt, welcher zuvor während der Konfiguration der Mesh-Hardware festgelegt wurde. Anschließend überträgt die Mesh-Hardware die unverschlüsselten Nachrichten innerhalb des aufgebauten, sicheren Kanals. Dabei kommt ebenfalls eine AES-Verschlüsselung zum Einsatz. [85]

Mittels sogenannten Spoofing von ”DESYNC”-Paketen kann ein erneuter Aufbau der WLAN-Verbindung erzwungen werden. Die anschließenden Übertragungen können mitgeschnitten werden und darauf folgend offline ausgewertet werden. Hierfür können beispielsweise die Programme von ”aircrack-ng” zum Einsatz kommen, welches mittels Brute-Force oder Passwortliste den verwendeten Schlüssel ermittelt. Insofern jedoch ein hinreichend komplexer Schlüssel konfiguriert wurde, ist ein zeitlich akzeptabler Erfolg nicht abzusehen. [109, 110]

Es existiert keine effiziente Möglichkeit die Kommunikation zu entschlüsseln. Folglich eignet sich diese Schnittstelle der Überwachung genauso wenig wie eine BLE-Verbindung.

3.3.6 Verschlüsselte Nachrichten

Insofern eine Live-Überwachung durch das Aufzeichnen von entschlüsselten LoRa®-Nachrichten nicht möglich ist, müssen die verschlüsselten Nachrichten selbst aufgezeichnet werden. Im Netzwerk werden die Nachrichten nur für die Übertragung gespeichert [97]. Ältere Nachrichten werden dementsprechend von Neueren überschrieben. Um alle Nachrichten im vollen Umfang zu erfassen, wird demzufolge eine Aufzeichnung dieser benötigt. Durch eine nachträgliche Analyse der Mesh-Hardware kann der verwendete Schlüssel ausgelesen und die Nachrichten somit im Nachhinein in Klartextform überführt werden [99].

Diese Vorgehensweise lässt sich prinzipiell auch bei BLE und WLAN umsetzen, jedoch sind diese auf eine geringere Reichweite beschränkt. Eine Aufzeichnung von LoRa®-Nachrichten liefert die gleichen Ergebnisse, bei gleichzeitiger Erhöhung des Abstandes zum untersuchten Gerät. Folglich ist eine Aufzeichnung der LoRa®-Signale jener von BLE oder WLAN vorzuziehen.

Darüber hinaus liefert Meshtastic® sogenannte ”Simple Keys”. Dabei handelt es sich nicht um kryptographisch unsichere Schlüssel, jedoch um öffentlich bekannte Schlüsselvorgaben. Hierbei sind im Quellcode von Meshtastic® 255 Schlüssel integriert, wobei der erste Schlüssel als Standardschlüssel verwendet wird [111]. In der Tabelle 3.2 sind die 255 Schlüssel aufgelistet. Folglich sollten diese vor einem Brute-Force-Angriff als Schlüssel getestet werden. [111–113]

Tabelle 3.2: Vordefinierte Standardschlüssel [111, 113]

Simple Key	Schlüsselwert
Default	0xd4f1bb3a20290759f0bcffabcf4e6901
Simple 1	0xd4f1bb3a20290759f0bcffabcf4e6902
Simple 2	0xd4f1bb3a20290759f0bcffabcf4e6903
Simple 3	0xd4f1bb3a20290759f0bcffabcf4e6904
...	...
Simple 253	0xd4f1bb3a20290759f0bcffabcf4e69ff
Simple 254	0xd4f1bb3a20290759f0bcffabcf4e6900

Desweiteren ist festzuhalten, dass nicht die vollständige Nachricht verschlüsselt wird. Wie in Kapitel 3.2 erwähnt ist der Inhalt des Headers auch ohne Wissen über den Schlüssel auszulesen. Aus den Header enthalten sind die Felder ”From”, ”To”, ”PacketID”, ”Flags”, ”ChannelHash”. Aus diesen Feldern lassen sich die folgenden Informationen erfassen. [76]

From Das ”From”-Feld gibt die vier Byte große Hardware-Adresse des LoRa®-Moduls vom Absender an [76]. Folglich lassen sich mit dieser Kennung Teilnehmer im Netzwerk eindeutig identifizieren. Dementsprechend ist es möglich bei einer ausreichend langen Aufzeichnung die Teilnehmeranzahl und deren Aktivität im Netzwerk zu ermitteln.

Darüber hinaus kann aus der Häufigkeit von gesendeten Nachrichten auf die Rolle oder Mobilität geschlossen werden. Die Konfiguration ermöglicht eine dynamische Anpassung der Häufigkeit von Positionsübertragungen. Ferner kann das häufige Senden von Nachrichten auf viele Textmitteilungen oder sonstige individuelle Nachrichten hinweisen. Somit handelt es sich bei solch einem Gerät mit hoher Wahrscheinlichkeit um einen persönlichen Knoten und nicht um einen stationären Router. Diese Vermutung kann durch stark variierende Nachrichtenhäufigkeit im Zeitverlauf erhärtet werden. Hierbei stellen insbesondere erkennbare Schlafrythmen im Aktivitätsverlauf ein weiteres starkes Indiz auf eine mobile Mesh-Hardware dar.

To Die Hardwareadresse des LoRa®-Moduls vom Empfänger ist im ”To”-Feld hinterlegt [76]. Folglich lässt diese ID einen Rückschluss auf die Teilnehmeranzahl im Netzwerk zu. Jedoch ist festzuhalten, dass in diesem Feld sehr häufig der Wert 0xFFFFFFFF steht, welcher als Broadcast-Adresse verstanden wird. Dementsprechend lassen sich in diesem Fall keine Rückschlüsse ziehen.

Jeder abweichende Wert beschreibt einen speziellen Empfänger und stellt damit eine Direktnachricht dar. Grundsätzlich kann von häufigen Direktnachrichten auf eine persönliche Bindung zwischen den Teilnehmern geschlossen werden. Diese kann jedoch auch einseitig, wie beispielsweise das automatisierte Senden, von Sensordaten sein.

PacketID Für jede Nachricht wird eine eindeutige ”PacketID” vergeben [76]. Diese dient der Vermeidung von doppelten Senden von Nachrichten in dem per Definition dynamischen Netzwerk. Darüber hinaus liefert dieses Feld für forensische Zwecke keine relevanten Informationen.

Flags Das ”Flag”-Feld unterstützt die Datenübertragung im Netzwerk. Derzeit kann in diesem Feld das Sprunglimit, auf englisch hop-limit, und eine Aufforderung zur Empfangsbestätigung über das ”want_ack”-Bit gesetzt werden. [76]

Bei einer Weiterleitung eines Paketes spricht man von einem Sprung, auf englisch hop. Nach jedem Sprung wird das Sprunglimit um eins reduziert [97]. Folglich kann eine Nachricht nur begrenzt oft weitergeleitet werden. Eine Limitierung dieser dient der Vermeidung von Paketen, die dauerhaft im Netzwerk weitergeleitet werden. Forensisch kann folglich aus dem aktuellen Sprunglimit die Tiefe des Netzwerkes abgeschätzt werden. Je vermashter das Netzwerk ist, desto eher erhalten alle Teilnehmer eine Nachricht und somit wird keine Weiterleitung darüber hinaus benötigt. Dagegen erfordert ein kettenartiger Netzwerkaufbau eine Weiterleitung jedes einzelnen Knotens in der Kette.

Eine Zustellung einer Nachricht kann mit dem Bit der Empfangsbenachrichtung zugesichert werden. Sofern diese nicht erhalten wird, erfolgt ein erneuter Versand. Dagegen wird bei Broadcast-Nachrichten überprüft, ob die Nachricht im Netzwerk weitergeleitet wird. Ist dies nicht der Fall, so erfolgt auch hier eine erneute Sendung. Folglich bietet das Feld der Empfangsbestätigung einen Rückschluss auf die Wichtigkeit der Nachricht. [76]

ChannelHash Das letzte Feld im Header stellt der ”ChannelHash” dar. Dieser ein Byte große Hashwert dient der Filterung, ob dieses Paket ausgewertet werden muss. Eine Kollision zwischen den Hashwerten zweier unterschiedlicher Kanäle ist möglich. Folglich wird dieser Fall bei der Auswertung des Paketes mitbeachtet. Der Hash dient vielmehr des Stromsparens durch die Verringerung der aufgebrachten Rechenleistung bei der Verarbeitung von empfangenen Paketen. Die Anzahl unterschiedlicher ”ChannelHash”-Werte liefert somit eine Abschätzung der genutzten, unterschiedlichen Kanäle im Netzwerk. [76]

3.4 Nachrichtenauswertung

Als letzten Schritt der forensischen Untersuchung schließt sich die Auswertung der aufgezeichneten Nachrichten an. Hierbei wird der Inhalt der Nachricht selbst ausgewertet. Dementsprechend wird davon ausgegangen, dass sich die Nachrichten im vorherigen Schritt entschlüsseln und protokollieren ließen. Die Auswertung der Nachrichten muss nicht ausschließlich nach einer abgeschlossenen Aufzeichnung dieser erfolgen. Vielmehr ist auch eine Auswertung im Live-Betrieb realisierbar. Damit bietet sich auch die Möglichkeit dynamisch auf Erkenntnisse zu reagieren. Im Folgenden wird auf besonders relevante Punkte einer solchen Auswertung eingegangen.

Textinhalt Der Textinhalt selbst beinhaltet das Gesprächsthema der Teilnehmer. Dementspechend lassen sich daraus individuelle Inhalte extrahieren.

Batteriestand von stationären Teilnehmern Gegenüber der Textnachrichten existieren auch Nachrichteninhalte, welche sich automatisiert auswerten lassen. Hierbei lässt sich der Batteriestand nennen, da von diesem auf die Art und Nutzung der Mesh-Hardware geschlossen werden kann.

Nimmt beispielsweise der Batteriestand kontinuierlich ab, und wird schnell und gleichmäßig geladen, so kann eine mobile Nutzung mit Aufladung am Stromnetz die Ursache sein. Dagegen kann von einem konstant geladenen Gerät auf einen fixen Standort mit festen Stromanschluss geschlossen werden. Weiter kann eine starke Variation des Akkustandes mit Korrelation nach dem Sonnenstand auf eine solarbetriebene Mesh-Hardware hinweisen. Letztere ist insofern relevant, da diese auf einen festen Standort hindeutet. Sofern dieses über einen längeren Zeitraum ausfällt, kann davon ausgegangen werden, dass eine Reparatur oder Akkuaustausch in naher Zukunft vorgenommen wird. Durch die in Kapitel 3.2 vorgenommene Ortung kann eine Observation der genannten Station zeitlich festgelegt werden. Damit ließe sich eine scheinbar besitzlose Mesh-Hardware einem Eigentümer zuordnen.

GPS-Positionen Insofern die übertragenen GPS-Positionen protokolliert werden, ermöglichen diese weitere Rückschlüsse. Die Bewegungsmuster können beispielsweise Wohnort, Arbeitsstätte, Vereine, Wohnorte von Freunden und allgemein den Tagesablauf offenbaren.

3.5 Zusammenfassung der forensischen Ansätze

In diesem Kapitel wurden diverse Möglichkeiten der einzelnen Teilschritte vorgestellt. Jedoch eignen sich einige Vorgehensweisen besser als andere. Somit wird hier zusammenfassend eine anzustrebeswerte Vorgehensweise vorgeschlagen. Sofern die in diesem Abschnitt genannten Einzelschritte nicht realisierbar sind, kann mit den alternativ vorgestellten Ansätzen aus diesem Kapitel abgewichen werden.

1. Existens eines Meshtastic®-Netzwerk erkennen, anhand der Debug-Funktion der Firmware. Insofern ein Netzwerk besteht, ist davon auszugehen, dass nach ausreichend langer Betrachtung ein Packet empfangen wird. Dieses wird im Debug-Modus angezeigt, selbst wenn der Kommunikationsschlüssel nicht bekannt ist. [99]
2. Anschließend können Netzwerkteilnehmer anhand der MAC-Adresse des Senders identifiziert werden [76]. Ferner lassen sich diese über eine Trilateration des Funksignals orten [78, 79].
3. Insofern es eine frei zugängliche, stationäre Mesh-Hardware im Netzwerk gibt, kann von dieser die Konfiguration ausgelesen werden [99]. Folglich ermöglicht der darin enthaltene symmetrische Schlüssel der Kommmunikationsverschlüsselung einen Einblick in den Netzwerkverkehr.
4. Anschließend wird der Nachrichtenaustausch im Netzwerk passend aufgezeichnet. Die Aufzeichnung muss dementsprechend nutzerfreundlich aufbereitet werden.
5. Schließlich lassen sich die protokollierten Mitschnitte forensisch auswerten und für die Untersuchung ergänzende Maßnahmen ergreifen.

4 Umsetzung

In diesem Kapitel wird die praktische Umsetzung der zuvor theoretisch beschriebenen Möglichkeiten der forensischen Analyse erläutert. Hierbei wird insbesondere Wert auf einen realistisch umsetzbaren Aufbau gesetzt. Damit sollen sich insbesondere Hindernisse offenbaren, die unter Laborbedingungen möglicherweise nicht auftreten.

Zunächst wird das aufgebaute Netzwerk im Detail beschrieben. Anschließend wird das Mesh-Netzwerk mittels eigener Firmware, aber auch durch Nutzung der Meshtastic®-Firmware erkannt. Fortführend wird eine Möglichkeit geschaffen, Nachrichten des Netzwerkes zu protokollieren. Abschließend wird in Kapitel 4.4 ein praktischer Versuch mit Hilfe von Freiwilligen durchgeführt. Dabei geht dieses Kapitel insbesondere auf die Ortung von Mesh-Hardware ein.

Die durch die Versuche gewonnenen Daten werden anschließend in Kapitel 5 ausgewertet.

4.1 Netzwerkaufbau

Für alle experiementellen Versuche der forensischen Analyse dieser Arbeit wird zunächst ein Netzwerk benötigt. Dieses soll eine authentische Umsetzung eines Netzwerks auf Kleinstadtgröße repräsentieren. Für ein solches Netzwerk wird zunächst die passende Mesh-Hardware benötigt.

4.1.1 Verwendete Hardware

Wie in Kapitel 2.9.1 beschrieben, unterstützt Meshtastic® diverse Hardwaretypen. Zur Auswahl der passenden Hardware für den Versuchsaufbau werden an diese die folgenden Anforderungen gestellt.

1. Betrieb legal innerhalb der EU
2. Mobiler Einsatz möglich
3. Hoher Funktionsumfang
4. Kompatibel zur verwendeten Firmwareversion (2.0.5)
5. Geringe Kosten

Da der Versuch innerhalb der EU durchgeführt wird und der Funkbetrieb gesetzlich geregelt ist, muss die Nutzung der Hardware legal ohne Funklizenz möglich sein [75]. Somit eignet sich keine Mesh-Hardware, die ausschließlich für den amerikanischen Betrieb entworfen wurde. Die Tabelle 4.1 beinhaltet folglich nur die legal wählbaren Möglichkeiten.

Meshtastic® zeichnet sich vor allem durch den mobilen Betrieb aus [8]. Dementsprechend soll mit der ausgewählten Hardware auch ein mobiler Einsatz ermöglicht werden. Das bedeutet insbesondere, dass für den mobilen Gebrauch keine externe Hardware notwendig sein soll. Folglich soll auf der Hardware ein Akku direkt verbaut sein.

Meshtastic® bietet neben dem reinen Mesh-Netzwerkbetrieb auch Zusatzfunktionen, wie das Teilen von GPS-Positionsangaben und Telemetriedaten [8]. Diese sollen in eine Untersuchung einfließen. Insbesondere, da GPS eine Ortungsmöglichkeit darstellt.

Die Tests werden mit der aktuellen Firmwareversion 2.0.5 durchgeführt. Folglich ist es erforderlich, dass diese die ausgewählte Hardware unterstützt.

Darüber hinaus muss der finanzielle Aspekt berücksichtigt werden. Bei gleichem Funktionsumfang sollte eine günstigere Hardware einer teureren Variante vorgezogen werden. Mit der Verwendung von kostengünstigerer Hardware kann mit dem gleichen monitären Einsatz ein größeres Netzwerk realisiert werden. Dieses ermöglicht eine breiter gestreute Analyse der forensischen Ansätze.

Tabelle 4.1: Mesh-Hardware mit Auswahlkriterien

Hardwaretyp	Mobil	GPS	Telemetriedaten	Firmware 2.0.5	Preis
RAK WisBlock	nein	ja	Funk, Batterie, optional Wetter	ja	65€ - 170€
T-Echo	ja	ja	Funk, Batterie, optional Wetter	ja	75-85€
T-Beam	ja	ja	Funk, Batterie	ja	50€
Lora V2.0	nein	nein	Funk	ja	35€
Heltec	nein	nein	Funk	nein	35€

Auf Grund der gegebenen Faktoren wird für den Versuch die folgende Mesh-Hardware verwendet.

- 10x T-Beam
- 4x Lora V2.0
- 2x T-Echo

Dabei stellen die T-Beam-Module das Hauptnetzwerk dar. Diese sind insbesondere durch ihr hervorragendes Preis-Leistungsverhältnis in der Meshtastic®-Community sehr beliebt [114]. Ergänzt wird das Netzwerk durch die neueren T-Echo-Module. Darüber hinaus bieten die Lora32 Module in der Version V2.0 eine kostengünstige Möglichkeit LoRa®-Signale zu empfangen. Folglich bieten sich letztgenannte Module für eine Auswertung mit individueller Software an. Ferner lässt sich diese Mesh-Hardware auch als stationäre Meshtastic®-Knoten verwenden. Darüber hinaus nutzen die fest positionierten Empfänger eine größere GPS-Antenne von 25 mm x 25 mm, um dem GPS-Empfang in Gebäuden zu verbessern.

4.1.2 Konfiguration der Mesh-Hardware

Meshtastic® erlaubt einen vielfältigen Einsatz der ausgewählten Hardware. Um die unterschiedlichen Verhaltensweisen mit einzubeziehen, werden die diversen Sender mit differenzierter Konfiguration versehen. Jedoch werden auch einige gleichbleibende Einstellungen benötigt, damit alle Geräte untereinander kommunizieren können. Somit gehen die folgenden Absätze auf die vorgenommene Konfiguration der Hardware von Tabelle 4.2 ein.

Für jede Mesh-Hardware wurde eine gemeinsame Konfiguration der Modemkonfiguration gewählt. Hier bietet Meshtastic® vordefinierte LoRa®-Einstellungen an. Standardmäßig nutzt die Mesh-Hardware die Modemkonfiguration ”LongFast”, bei der eine Bandbreite von 250 kHz, eine Chirp-Rate von 8 und ein Spreizfaktor von 11 zum Einsatz kommt. Diese Modemkonfiguration wurde aufgrund des Kompromisses aus Geschwindigkeit und Reichweite auch für dieses Netzwerk gewählt. Darüber hinaus wird für einen legalen Betrieb in Deutschland die Funkkonfiguration EU_868 hinterlegt, wodurch die Frequenz 868 MHz genutzt wird. Gleichzeitig regelt diese Einstellung die maximale Sendeleistung von 27 mW, was einen legalen Betrieb ohne Funklizenz ermöglicht. [27, 75]

Weiterführend lässt sich eine Mesh-Hardware in verschiedene Rollen einteilen. In Kapitel 3.1.2 wurde bereits der Modus ”CLIENT_MUTE” vorgestellt. Ferner existieren die Rollen ”CLIENT”, ”ROUTER” und ”ROUTER_CLIENT”. Um diese abzudecken, werden der Mesh-Hardware, wie in Tabelle 2.1 aufgezeigt, die jeweiligen Rollen zugewiesen. Da sich die Mesh-Hardware im Haus 8 faktisch in einem faradayschen Käfig befindet, wurde dieser mit der Rolle ”CLIENT_MUTE” eine Teilnahme am Routen von Nachrichten untersagt. Den besonders erhöhten Sendern im Haus 6, Haus 42 und dem Wohnhaus an der Adresse Am Sportplatz 10 wurde auf Grund ihrer guten Empfangslage die Rolle als Router zugewiesen. Wobei jedoch die Mesh-Hardware im Haus 6 und Am Sportplatz 10 zusätzlich auch als Client agieren. Die entsprechenden Rollen lauten ”ROUTER” und ”ROUTER_CLIENT”. Durch diese Konfiguration werden diese Knoten beim Routing bevorzugt. Schließlich wird jeder weiteren Mesh-Hardware die Rolle ”CLIENT” zugewiesen. [11]

Tabelle 4.2: Geräteübersicht

Hardwaretyp	GPS-Antenne	ID	Name	Standort	Rolle
T-Beam	25x25 mm	!8487e23c	Station 1	Feldstraße 3	CLIENT
T-Beam	25x25 mm	!8487da68	Station 2	Am Sportplatz 10	ROUTER_CLIENT
T-Beam	25x25 mm	!8487de40	Station 3	Haus 8-107	CLIENT_MUTE
T-Beam	25x25 mm	!8487e1d4	Station 4	Haus 42-101	ROUTER
T-Beam	25x25 mm	!8487df28	Station 5	Haus 6-405/1	ROUTER_CLIENT
T-Beam	6x16 mm	!55c6e158	Mobil 1	Mobil	CLIENT
T-Beam	6x16 mm	!55c573e0	Mobil 2	Mobil	CLIENT
T-Beam	6x16 mm	!55c56310	Mobil 3	Mobil	CLIENT
T-Beam	6x16 mm	!55c6d660	Mobil 4	Mobil	CLIENT
T-Beam	6x16 mm	!55c6db64	Mobil 5	Mobil	CLIENT
Lora V2.0	keine	!9197f264	Station 6	Haus 11 bei TMM	CLIENT
Lora V2.0	keine	defekt	—	—	—
Lora V2.0	keine	!9197a6b4	—	Mobil	—
Lora V2.0	keine	!9198fd04	—	Mobil	CLIENT
T-Echo	6x16 mm	!d2ee44e8	Mobil 6	Mobil	CLIENT
T-Echo	6x16 mm	!587d47d6	Mobil 7	Mobil	CLIENT

Damit alle Knoten untereinander über eine kryptographisch gesicherte Verbindung kommunizieren können, werden gemeinsame Kanäle eingerichtet. Hierfür wird zunächst für jeden Kanal ein neuer AES-Schlüssel über den kryptographischen Zufallszahlengenerator im Betriebssystem Linux Mint erzeugt. Zu diesem Zweck lassen sich Zufallszahlen von `/dev/urandom` auslesen. Anschließend werden daraus Kanäle definiert und der Mesh-Hardware zugewiesen. Hier zeigt die Tabelle 4.3 den jeweils gewählten Kanal auf. Wie in der Tabelle 4.3 dargestellt ist, erhält der primäre Kanal 0, die Kanaldefinition ”Station”, beziehungsweise ”Mobil”, je nachdem, wie der Einsatz der jeweiligen Mesh-Hardware vorbestimmt ist. Der mobilen Mesh-Hardware wird anschließend die Kanaleinstellung ”Station” über den Kanal 2 hinzugefügt. Diese Konfigurationsweise ermöglicht, dass die mobile Mesh-Hardware Nachrichten der Stationen 1-6 empfangen und entschlüsseln kann, aber gleichzeitig die mobile Mesh-Hardware nicht auf den Stationen angezeigt wird. Da die Stationen sich auch bei Drittpersonen befinden, wird diesen ein Einblick in die Kommunikation der mobilen Knoten verwehrt. Das dient vor allem einer minimalen Zusicherung der Privatsphäre der Probanden im Praxistest. Folglich sehen sich mobile Geräte untereinander und die Stationen, aber die Stationen lediglich sich selbst. Der zweite Kanal ist als administrativer Kanal konfiguriert, sodass eine Änderung der Konfiguration aller Geräte ohne einen direkten Zugriff auf diese ermöglicht wird [94]. Schließlich ist festzuhalten, dass jede mobile Mesh-Hardware einen eigenen mobilen Kanal besitzt, über welchen,

unabhängig anderer Kanäle, Direktnachrichten versendet werden können. Eine parallele Aufzeichnung findet über die Station 1 statt, wodurch diese eine gesonderte Konfiguration erhält. Im Anhang C ist die vollständige Konfiguration aller Mesh-Hardwareknoten zu entnehmen.

Tabelle 4.3: Kanalkonfiguration

ID	Kanal 0	Kanal 1	Kanal 2	Kanal 3	Kanal 4	Kanal 5	Kanal 6	Kanal 7
!8487e23c	Station	Mobil	admin	Mobil 1	Mobil 2	Mobil 3	Mobil 4	Mobil 5
!8487da68	Station	admin	—	—	—	—	—	—
!8487de40	Station	admin	—	—	—	—	—	—
!8487e1d4	Station	admin	—	—	—	—	—	—
!8487df28	Station	admin	—	—	—	—	—	—
!55c6e158	Mobil	admin	Station	Mobil 1	—	—	—	—
!55c573e0	Mobil	admin	Station	Mobil 2	—	—	—	—
!55c56310	Mobil	admin	Station	Mobil 3	—	—	—	—
!55c6d660	Mobil	admin	Station	Mobil 4	—	—	—	—
!55c6db64	Mobil	admin	Station	Mobil 5	—	—	—	—
!9197f264	Station	admin	—	—	—	—	—	—
defekt	—	—	—	—	—	—	—	—
!9197a6b4	—	—	—	—	—	—	—	—
!9198fd04	—	—	—	—	—	—	—	—
!d2ee44e8	Mobil	admin	Station	Mobil 6	—	—	—	—
!587d47d6	Mobil	admin	Station	Mobil 7	—	—	—	—

Ferner wurde die Datenübertragung der GPS-Positionen angepasst. Neben den Angaben zu Longitude, Latitude und Altitude werden zusätzlich die sichtbare GPS-Satellitenanzahl und Geschwindigkeit über Grund übertragen. Weiterführend wurde das standardmäßige Sendeintervall dieser Informationen von 15 auf 5 Minuten beschleunigt [10]. Bei einer signifikanten Abweichung der neuen eigenen GPS-Position zur letzten wird ein Update dieser vor Ablauf der 5 Minuten ermöglicht. Letztgenannte Einstellung wurde mittels des sogenannten ”Smart Broadcast” aktiviert [10].

Anschließend wird die konfigurierte Mesh-Hardware in Mittweida räumlich verteilt, um das Netzwerk zu bilden.

4.1.3 Standorte der Mesh-Hardware

Fünf der zehn T-Beam-Module sollen für als eine fest installierte Station genutzt werden. Die Positionen soll so gewählt werden, dass diese bereits ein Netzwerk im Hochschulgebiet von Mittweida aufspannen. Da besonders erhobene Positionen einen guten Empfang für einen großen Bereich ermöglichen (Abbildung 3.2), wurden die Gebäude 6 und 42 der Hochschule Mittweida ausgewählt. Beide Häuser stehen bereits erhöht auf einem Berg. Insbesondere das Haus 6 bietet mit seinen vier Stockwerken einen vollständigen Blick über Mittweida. Ergänzt wird dieses Kernnetzwerk durch ein Wohnhaus in der Nähe des Bahnhofs. Auch dieses steht erhöht auf einem Berg und bietet ebenfalls vier Stockwerke. Die verbleibenden Knoten werden an diversen Standorten zwischen den drei genannten Kernknoten verteilt aufgestellt. Die realen GPS-Positionen wurden mittels `https://www.google.com/maps` in der standardmäßigen, topographischen Ansicht bestimmt.

Station 1 Die erste stationäre Mesh-Hardware befindet sich in einem Privathaushalt der Feldstraße 3. Innerhalb der Wohnung des dritten Stockwerks wurde die Mesh-Hardware in der Tiefe des Raumes erhöht auf 2 m platziert. Die Abbildung 4.1 zeigt die genaue Platzierung der Hardware. Hier weist das rote Kreuz im Bild auf die exakte Position mit den Koordinaten (50.98928, 12.97020) hin.

(a) Außenbereich **(b)** Positionierung im Raum

Abbildung 4.1: Räumliche Positionierung von Station 1

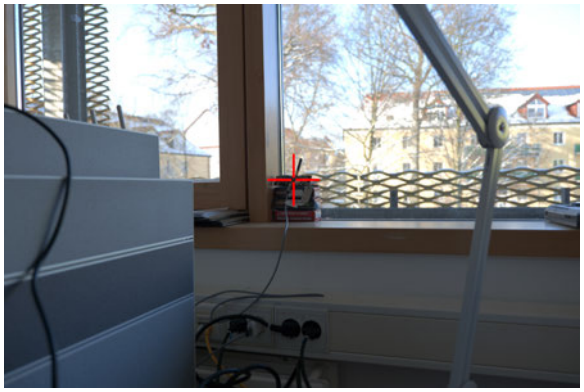
Station 2 Die zweite Station ist ebenfalls in einem privaten Haushalt positioniert. Im Dachgeschoss des Hauses mit der Adresse ”Am Sportplatz 10” kann eine große Fläche Mittweidas überblickt werden. Die GPS-Koordinaten dieser Station lauten (50.98436, 12.96553). Innerhalb der Wohnung steht die Mesh-Hardware auf dem Fensterbrett des Dachfensters in Richtung Süden.

(a) Außenbereich **(b)** Positionierung im Raum

Abbildung 4.2: Räumliche Positionierung von Station 2

Station 3 Wie auf der Abbildung 4.3 zu erkennen ist, befindet sich die dritte Station im Haus 8 der Hochschule. Dieses Gebäude zeichnet sich vor allem durch die metallene Gitterstruktur außerhalb der Fassade aus. Folglich kann es durch den Abschirmungseffekt des faradayschen Käfigs bei diesem Knoten zu Empfangsproblemen führen. Die Positionierung der Hardware im ersten Obergeschoss am Fenster des Raumes 8-107 stellt damit einen Knoten unter erschwerten Umgebungsbedingungen dar. Weiterführend lassen sich für diesen Knoten die GPS-Koordinaten (50.98928, 12.97020) festhalten.

(a) Außenbereich



(b) Positionierung im Raum

Abbildung 4.3: Räumliche Positionierung von Station 3

Station 4 Das Laserinstutut der Hochschule Mittweida ist erhöht auf einem Berg erbaut. Von dort aus überblickt die Mesh-Hardware der Station 4 einen Großteil von Mittweida. Innerhalb des Gebäudes ist der Sender im ersten Obergeschoss auf dem Fenstersims des Sekretariats im Raum 101 aufgestellt. Diese Position wird durch die GPS-Koordinaten (50.98400, 12.97100) beschrieben.

(a) Außenbereich

(b) Positionierung im Raum

Abbildung 4.4: Räumliche Positionierung von Station 4

Station 5 Mit seiner Positionierung im vierten Stockwerk des Hauses 6 der Hochschule Mittweida befindet sich die Mesh-Hardware der Station 5 am höchsten von allen Knoten. Hier steht der Sender auf einem Metallschrank im Lagerraum 405/1 auf der Nordseite des Gebäudes. Somit lassen sich für diesen Sender die GPS-Koordinaten (50.98826, 12.97091) festhalten.

(a) Außenbereich **(b)** Positionierung im Raum

Abbildung 4.5: Räumliche Positionierung von Station 5

Station 6 Zuletzt wurde an den GPS-Koordinaten (50.98543, 12.96839) eine Mesh-Hardware hinterlegt. Dabei handelt es sich um das Gebäude 11 der Hochschule. An dem zur Kreuzung gerichteten Erker hat sich das Motorsportteam TMM niedergelassen. Hier wird der Sender der Station 6 im ersten Obergeschoss direkt am Fenster betrieben. Wie auch für die anderen stationären Mesh-Knoten verdeutlicht die entsprechende Abbildung den genauen Standort.

(a) Außenbereich **(b)** Positionierung im Raum

Abbildung 4.6: Räumliche Positionierung von Station 6

Zusammefassen stellt die Abbildung 4.7 die sechs Stationen und deren Lokalisierung dar.

4.2 Realisierung der Netzwerkerkennung

Um die reine Existenz des Netzwerkes nachzuweisen wurde ein LoRa-V2.0-Mikrocontroller mit der gleichen Meshtastic®-Firmware geflasht, wie bereits die Knoten des Netzwerkes. Zum Empfang der Nachrichten wird zunächst das Funkmodul auf den europäischen Kanal EU_868 und die Nutzung des Modempräfixes ”LongFast” eingerichtet. Desweiteren wurde der Debug-Log über die Option `--set device_log_enabled true` aktiviert [11]. Abseits dessen wurde keine weitere Einstellung abweichend von der Standardkonfiguration vorgenommen. Somit nutzt die Mesh-Hardware den Standardschlüssel ”Default” aus der Tabelle 3.2 [112]. Die für das restliche Netzwerk genutzten AES-Schlüssel sind dieser Mesh-Hardware nicht bekannt.

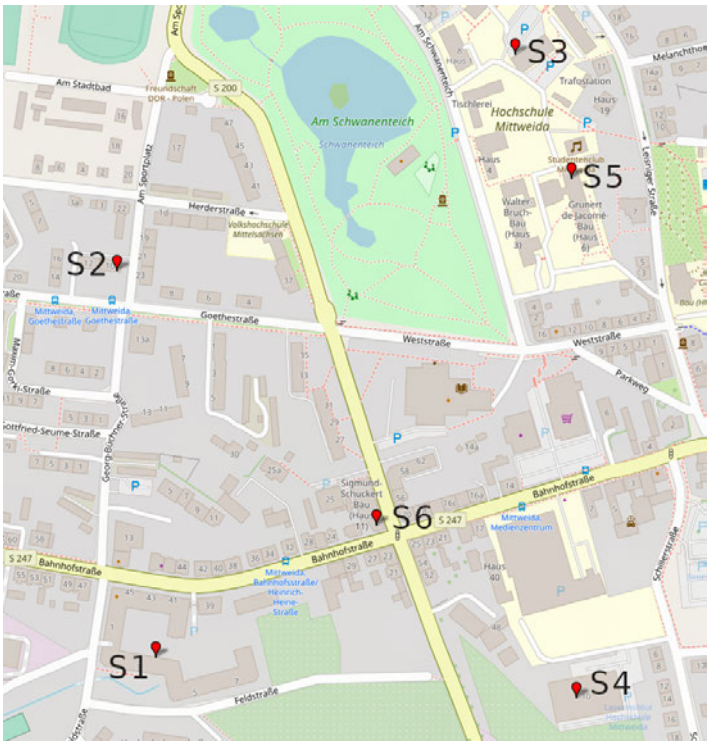


Abbildung 4.7: Karte mit Positionen der stationären Knoten

Anschließend wird das Microcontroller-Board mit Strom über ein USB-Kabel versorgt und einige Minuten betrieben. Folglich ermöglicht dies der Mesh-Hardware Meshtastic®-Nachrichten in diesem Zeitraum zu empfangen. Diese können über die kabelgebundene USB-Schnittstelle mittels des Befehls `meshtastic --debug` ausgelesen werden.

Außerdem wird durch ein eigenständiges Programm der Empfang von LoRa®-Paketen mit gleicher Modemkonfiguration gemessen. Um dies zu erreichen wird das Modem der Lora32-Hardware mit der Meshtastic®-Konfiguration ”LongFast” versehen. Folglich wird die Bandbreite auf 250 kHz, die Kodierungsrate auf 8 und der Spreizfaktor auf 11 gesetzt [27]. Anschließend können alle empfangenen Pakete und jene mit der Meshtastic®-Nachrichtenpreambel, den Magic-Bytes, gezählt werden [70]. Somit lässt sich neben der Detektion des Meshtastic®-Netzwerkes auch abschätzen, wie sehr die Funkfrequenz mit anderen LoRa®-Signalen der gleichen Modemkonfiguration benutzt wird.

4.3 Konfigurationserfassung

Die Konfiguration einer Mesh-Hardware lässt sich auf verschiedene Weisen erfassen [96, 99]. In dieser Umsetzung werden die Einstellung über eine serielle USB-Verbindung ausgelesen. Hierfür wird der Befehl `meshtastic --info` genutzt, welcher eine Übersicht der Konfiguration produziert [115]. Gleichzeitig erhält der Anwender eine Auflistung aller verbundenen Knoten des Netzwerkes.

4.4 Ortung mithilfe von Probanden

Um zu untersuchen, wie präzise sich die Position von Knoten des Mesh-Netzwerkes bestimmen lassen, wurde ein Experiment mit Probanden durchgeführt. Hierfür stellten sich Studenten der Hochschule Mittweida freiwillig zur Verfügung. Diese erhielten die als Mobil 1-7 konfigurierte Mesh-

Hardware und den Auftrag die stationären Teilnehmer des Netzwerkes zu lokalisieren. Zunächst wurde allen Teilnehmern eine Einführung in die Bedienung und Datenquellen der Mesh-Hardware präsentiert. Insbesondere wurde erläutert, wie mit Hilfe der Signalstärke eine Position eines Sender bestimmt wird. Darüber hinaus arbeiteten alle Teilnehmer unabhängig voneinander, sodass eine bessere Einschätzung der erzielten Ergebnisse gewährleistet wird. Jedoch ist darauf hinzuweisen, dass mit sieben Teilnehmern keine statistisch gesicherten Aussagen getroffen werden können.

Die Dauer des Versuchs war auf eine Woche ausgelegt, damit den Probanden genügend Zeit für ihre Suche zur Verfügung gestellt werden konnte. Diese erstreckte sich vom 16.01.2023 bis einschließlich zum 23.01.2023. Innerhalb des Zeitraumes teilten sich die Teilnehmer selbstständig die Zeit für die Suche der Mesh-Hardware ein. Als Orientierung wurde hierfür eine Stunde pro Tag angegeben. Dies wurde nur als Richtwert vorgegeben.

Für die Protokollierung des Versuches wurde den Probanden eine unausgefüllte Unterlage von Anhang B für jede Station zur Verfügung gestellt. Gleichzeitig wurde das in Anhang B dargestellte, ausgefüllte Beispielblatt als Orientierungshilfe angefertigt. Für jede Station wurde eine vermute GPS-Koordinate und die Altitude derer angegeben. Zusätzlich gab der Proband an, wie sicher er sich bei der genannten Position ist. Außerdem wurde genannt, wie groß der Radius um den Standort sein muss, damit die gesuchte Mesh-Hardware mit Sicherheit sich innerhalb des aufgespannten Kreises befindet. Diese Angabe repräsentiert die maximal zu erwartende Abweichung in Metern. Ergänzend konnten optionale Angaben zum Standort ausformuliert werden. Während der Suche notierte der Proband die Messwerte und Erkenntnisse, die zur ermittelten Position führen.

4.5 Messung der Signalstärke von BLE

Für eine Erfassung des Bluetooth Low Energy Signals findet ein Fairphone 4 Anwendung. Auf diesem Endgerät wird die Software ”BLE Scanner” der Firma ”Bluepixel Technologies LLP” in der Version 3.21 genutzt. Diese bildet auf dem Display die empfangene Signalstärke von Bluetooth-Signalen ab. Zusätzlich lassen sich MAC-Adressen und Namensbeschreibungen der einzelnen Sender ablesen. [116]

Mit einem waagerecht und auf Brusthöhe gehaltenen Mobiltelefon wird das entsprechende Gebiet abgelaufen. Zur Durchführung einer Messung an einem Punkt wird an diesem angehalten und mehrere Messwerte des entsprechenden Senders erfasst. Dabei wird der menschliche Körper auf die dem Sender abgewandten Seite positioniert, um die Signalstärke nicht zu beeinflussen. Aus den detektierten Daten wird anschließend ein Mittelwert gebildet.

4.6 Automatisierte Aufzeichnung des Mesh-Netzwerkes

Parallel zur Ortung durch die Probanden findet eine automatisierte Aufzeichnung des Netzwerkes statt. Diese soll statistische Einblicke in GPS-Daten und des generellen Netzwerk-Status liefern. Darüber hinaus wird eine automatisierte, forensische Aufzeichnung von Nachrichten und Telemetriedaten demonstriert.

Zunächst wird eine Schnittstelle zur Mesh-Hardware benötigt. Diese liest empfangene Daten aus und leitet diese zum Speichern weiter. Eine Trennung dieses Auslesens von der speichernden Funktion ermöglicht das gleichzeitige Protokollieren von verschiedenen Standorten. Dennoch steht anschließend der gesamte Datensatz für eine Auswertung zur Verfügung. Im Kern der Aufzeichnung befindet sich eine Datenbank, in der alle Informationen aggregiert werden. Für diese wird sowohl eine maschinelle, als auch eine menschliche Schnittstelle benötigt. Somit können sowohl automatische Auswertungen erstellt, als auch manuelle Analysen an den gewonnenen Daten durchgeführt werden. In den folgenden Unterkapiteln wird die Umsetzung dieser Schritte detailliert beschrieben.

4.6.1 Extraktionsskript

Als eine Schnittstelle für empfangene Daten bietet Meshtastic® bereits eine Bibliothek in der Programmiersprache Python an [117]. Diese wird als Anbindung genutzt. Hierfür werden Klassen für jeden Pakettyp definiert, sodass explizit auf die unterschiedlichen Inhalte derer eingegangen werden kann. Jedoch beinhalten die Pakete unterschiedlicher Typen einige gemeinsame Datenfelder, wie beispielsweise Headerinformationen mit Sender, Empfänger und Hop-Limit, aber auch Empfangsstärke und -zeit [76]. Um die Lesbarkeit des Quellcodes zu verbessern und dessen Modularität zu erhöhen, werden die gemeinsamen Informationen durch Vererbung gebündelt. Anschließend können die relevanten Daten extrahiert werden.

Für die Meshtastic®-Anwendungen ”SERIAL_APP” und ”ROUTING_APP” konnten keine übertragenden Pakete mit der Bibliothek aufgelöst werden. Insofern diese mit zukünftigen Versionen ausgewertet werden, gibt diese das Extraktor-Skript auf der Konsole aus. Darüber hinaus bietet die Python-Bibliothek eine Schnittstelle für die Meshtastic®-Anwendungen ”ADMIN_APP”, ”REMOTE_HARDWARE_APP”, ”SIMULATOR_APP” und ”TRACEROUTE_APP” an [118]. Da unter diesen Ports keine Nachrichten erwartet werden, sind diese in dem Skript nicht integriert. Jedoch ermöglicht der modulare Aufbau bei Bedarf eine schnelle Integration dieser Meshtastic®-Daten in den Protokollierungsprozess.

Abbildung 4.8 verdeutlicht den Aufbau einer solchen Extraktionsklasse. Die Python-Biliothek von Meshtastic® übergibt mit der Variable `packet` ein Dictionary [119]. Dieses beinhaltet sowohl die Rohdaten, als auch die interpretierten Angaben [119]. Hieraus werden mittels der entworfenen Extraktorklasse die relevanten Daten erhoben. Folglich entfallen redundante Angaben. Durch optionale Felder wird es erforderlich, das Datenpaket auf deren Existens zu prüfen, bevor auf diese zugegriffen wird. Zu diesem Zweck enthält jede Extraktorklasse die Methoden `return_if_contains` und `save_if_contains`. Aus allen gewonnenen Daten wird ein Dictionary erzeugt, welches anschließend an die API des Webservers übertragen wird. Dort findet eine Speicherung statt.

4.6.2 Webserver mit Datenbank und REST-API

Für eine strukturierte Hinterlegung der gesammelten Daten werden diese in einer Datenbank gespeichert. Für diese wird zunächst eine Schnittstelle zu dem im vorherigen Kapitel 4.6.1 beschriebenen Skript benötigt. Die Datenbank benötigt eine Möglichkeit der Datenaufnahme. Gleichzeitig soll es Nutzern ermöglicht werden, die erfassten Daten in Echtzeit zu begutachten. Ebenso wird eine Möglichkeit benötigt, maschinelle Auswertungen an den Daten vornehmen zu können. Für all diese Anforderungen bietet sich ein Webserverframework an.

```
class MeshtasticTextExtractor(MeshtasticPacketExtractor):
  tag = "meshtastic.receive.text"
  url = f"{server}/message_logger/text/create/api"

  def extractData(self, packet):
    if not super().extractData(packet):
      return False

    self.save_if_contains("channel", "channel", packet)
    decoded = self.return_if_contains("decoded", packet)
    if decoded:
      if self.save_if_contains("payload", "payload", decoded):

        # decode payload as base64
        self.data_dict["payload"] =
    b64encode(self.data_dict["payload"]).decode()

      # payload interpreted as UTF8
      self.save_if_contains("text", "text", decoded)
    return True
```

Abbildung 4.8: Klasse zum extrahieren von Textnachrichten

Auf Grund der der bereits in Python vorhandenen Bibliothek wurde für das Programm ebenfalls die Programmiersprache Python ausgewählt. Als Webframework wurde das der Firma Django Software Foundation vertriebene Produkt django genutzt, da für dieses Webframework keine Einarbeitungszeit benötigt wurde [120].

Zur Erfassung der Daten werden zunächst Modelle für die einzelnen Meshtastic®-Anwendungen benötigt [121]. Wie bereits im Extraktionsskript wird auch hier Vererbung genutzt, um gemeinsame Datenfelder einmalig an einer zentralen Stelle zu definieren. Außerdem wird für jedes gespeicherte Paket die Möglichkeit geschaffen, dieses zu kommentieren. Somit können durch eine Auswertung Daten direkt mit Notizen versehen werden. Ergänzt wird jedes gespeicherte Paket durch einen Zeitstempel, welcher die Zeit des Speichern selbst widerspiegelt.

Ferner können aus den empfangenen Daten einzelne Teilnehmer des Netzwerkes anhand ihrer ID-Nummer erkannt werden. Für diese Knoten wird ein eigenes Modell angelegt. Damit ergibt sich die Gelegenheit, eine zusammenfassende Übersicht für jeden Knoten zu erstellen. Konkret handelt es sich dabei um die letzte bekannte Position, den letzte Zeitpunkt einer Aktivität im Netzwerk, den Akku-Stand, die aktuelle Namensgebung und schließlich das Hardwaremodell. Ebenso wird auch hier die Möglichkeit hinzugefügt, den Netzwerkknoten selbst mit Kommentaren zu versehen. Ergänzend lässt sich festhalten, dass Datenfelder der erhaltenen Daten optional sind. Somit werden diese aus forensischen Gründen nicht mit Standardwerten gefüllt. Andernfalls ließen sich übermittelte Werte, die dem Standardwert entsprechen, nicht von diesem unterscheiden.

```
class ReceivedNodeInfo(ReceivedPackage):
    """
    One received data package of the node_info type.
    """
    long_name = models.TextField(null=True, blank=True,
      verbose_name="Name")
    short_name = models.TextField(null=True, blank=True,
      verbose_name="Kürzel")
    hardware_model = models.TextField(null=True, blank=True,
      verbose_name="Hardware")
```

Abbildung 4.9: Klasse zum Hinterlegen einer Node-Info-Nachricht

```
class ReceiveNodeInfoSerializer(ReceiveMessageSerializer):
    """
    Serializer to create a node info entry via the api.
    """
    class Meta(ReceiveMessageSerializer.Meta):
        model = ReceivedNodeInfo
```

Abbildung 4.10: Klasse zum Übertragen der Daten in Modellobjekt

In der Abbildung 4.9 wird beispielhaft das Modell der Node-Info präsentiert. Dieses kann typenspezifisch Namensangaben zu einem Netzwerkknoten speichern. Darüber hinaus wird die Hardwarebezeichnung ebenfalls mit hinterlegt. Diese kann beispielsweise ”TBEAM”, ”T_ECHO” oder ”NANO_G1” sein [122]. Eine Liste aller bekannten Hardwaremodelle kann der entsprechenden Protobuf-Liste entnommen werden [76].

Django ermöglicht das Speichern der Daten in diversen Datenbanken. Dazu zählen die Programme SQLite, PostgreSQL MariaDB, MySQL und Oracle [123]. Innerhalb des Prototypen für diese Arbeit wird SQLite als Datenbank genutzt, da hierbei die Daten innerhalb einer einzelnen Datei gespeichert werden können [124]. Folglich können Aufzeichnungen durch das Kopieren der Datenbankdatei übertragen werden. SQLite wirbt außerdem mit einem geringen Ressourcenverbrauch, crossplatformkompartibilität und einem einfachen Dateiaufbau [124]. Diese Faktoren sind für eine forensische Aufzeichnung von Vorteil. Dennoch ist SQLite auf kleinere Anwendungen mit eher geringem Datendurchsatz orientiert [124]. Diese Einschränkung stellt jedoch auf Grund des geringen Datendurchsatzes von LoRa® keine Beeinträchtigung dar.

Um eine einheitliche Schnittstelle für externe Anwendungen bereitzustellen, wird eine REST-API eingerichtet. Hierfür kommt das Django-Rest-Framework als Erweiterung von django zum Einsatz. Das Framework ermöglicht eine Übertragung der Daten im JSON-Dateiformat zum und von dem Server [125]. Da die Python Software Foundation selbst mit der JSON-Bibliothek eine effektive Möglichkeit bietet ein Dictionary umzuwandeln, kann eine Datenübertragung mit dieser Werkzeugkombination unmittelbar umgesetzt werden [126].

```
class ReceivedNodeInfoCreateAPI(CreateAPIView):
    """
    API to create a node info message entry.
    """
    serializer_class = ReceiveNodeInfoSerializer
```

Abbildung 4.11: Klasse zum Bereitstellen der Webschnittstelle

```
1  class ReceivedNodeInfoOverviewView(PaginatedListView):
2      """
3      View all received node info messages.
4      """
5      model = ReceivedNodeInfo
6      template_name: str = 'message_logger/node_info/overview.html'
7
8      def get_queryset(self):
9          """
10         Order list to show the last received first.
11         """
12         return ReceivedNodeInfo.objects.filter(from_id=
           self.kwargs["node_id"]).order_by("-timestamp")
```

Abbildung 4.12: Klasse zum Anzeigen einer Übersicht

Für die Bereitstellung der Schnittstelle werden ferner ein Serializer und eine entsprechende Ansicht benötigt [125]. Ein Serializer ermöglicht eine Übertragung der Daten in das Format des Modells [127]. In der Abbildung 4.10 ist der entsprechende Serializer für das Modell aus Abbildung 4.9 dargestellt. Hier wird das zuvor erstellte Modell in der Meta-Subklasse referenziert. Der schlanke Aufbau des Serializers wird durch die Vererbung realisiert. Folglich sind innerhalb der Basisklasse die übergreifenden Methoden implementiert. So wird beispielsweise bei dem Empfang jeder Nachricht die Daten des entsprechenden Senders aktualisiert. Dazu gehört unter anderem der Zeitpunkt des letzten Kontakts zu dem jeweiligen Knoten.

Anschließend wird eine Ansicht, im englischen View, benötigt, sodass extern Daten hinzugefügt werden können. Im Falle des Node-Info-Modells verdeutlicht die Abbildung 4.11 eine solche Ansicht. Damit die Daten erfolgreich aufgenommen werden können, wird innerhalb der Ansicht der entsprechende Serializer angegeben. Durch eine Vererbung von der Klasse ”CreateAPIView” wird darüber hinaus sichergestellt, dass lediglich Daten hinzugefügt werden können [128]. Ein Löschen oder Verändern der Daten soll aufgrund der späteren forensischen Auswertung nicht möglich sein.

Fortführend wird dem Benutzer eine Webseite erstellt, auf derer er die Daten aufgelistet bekommt. Hierfür ist die in Abbildung 4.12 dargestellte Klasse in den Ansichten implementiert. Zum Erstellen der Webseite wird das entsprechende Modell und HTML-Template angegeben. Letzteres wird mit den Daten aus der Datenbankabfrage gefüllt. Dazu kommt die Methode `get_queryset` zum Tragen. Wie in Zeile 12 des Quellcodeausschnittes zu erkennen ist, findet eine Filterung nach Knoten-ID statt. Ferner werden die Daten nach Zeitstempel sortiert, sodass neuere Einträge zuerst gelistet

```
class ReceivedNodeInfoCommentForm(forms.ModelForm):
    """
    Form to write a comment to a node info.
    """
    class Meta:
        model = ReceivedNodeInfo
        fields = ["comment"]
```

Abbildung 4.13: Klasse zum Bereitstellen eines Formulars

werden. Schließlich findet eine Aufteilung der Daten auf Seiten statt, sodass eine einzelne Ansicht stets übersichtlich und auf Grund der reduzierten Datenmenge schnell geladen werden kann. Hierfür wurde eine übergreifende Basisklasse `PaginatedListView` erstellt, von welcher diese Ansicht erbt.

Eine Veränderung der Daten durch den Nutzer ist unerwünscht. Jedoch soll dem Anwender die Möglichkeit geschaffen werden, die Daten zu kommentieren. Für diesen Zweck wird eine Formularklasse implementiert, mit welcher eine Bearbeitung des Kommentars ermöglicht wird. Alle weiteren Datenfelder bleiben darüber hinaus unberührt. Die Abbildung 4.13 stellt ein solches Formular dar. Es handelt sich dabei um die Kommentierung eines Node-Info-Datenobjektes aus Abbildung 4.9. Folglich wird in der Meta-Klasse das entsprechende Modell und das Datenfeld spezifiziert. Durch Nutzung dieses Formulars kann anschließend eine Ansicht für die Kommentarfunktion implementiert werden.

4.6.3 Skripte zur maschinellen Auswertung

Schließlich können die gewonnenen Daten maschinell ausgewertet werden. Hierfür bietet das Framework django die Möglichkeit, eigene Kommandos zu definieren [129]. Innerhalb dieser Skripte kann auf die Modelle, wie innerhalb der Präsentationsklassen, im englischen als View bezeichnet, zugegriffen werden [129]. Das Filtern und Aggregieren von Daten wird somit erleichtert.

Zunächst kann die Genauigkeit der GPS-Angaben bestimmt werden. Zu diesem Zweck wird die Datenmenge auf stationäre Knoten des Mesh-Netzwerkes beschränkt. Da diese sich stets an bekannten Positionen befanden, kann eine Abweichung der übertragenden Position zur realen Lage gemessen werden. Durch die Betrachtung über einen längeren Zeitraum hinweg kann somit eine statistische Aussage über die mittlere absolute Abweichung in der Positionsgenauigkeit getroffen werden.

Ebenso lassen sich Aussagen über die Signalstärke des empfangenden Signals treffen. Es können RSSI- mit den SNR-Messwerten gegenübergestellt werden. Hierfür können alle empfangenen Nachrichten für eine Auswertung hinzugezogen werden.

Fortführend kann die gemessene Signalstärke mit der realen Distanz zwischen den Knoten in Verbindung gesetzt werden. Auch hier kann die Aufzeichnung von einem stationären Mesh-Teilnehmer genutzt werden. Bei Signalen mit einem Hop-Limit von vier fand noch keine Weiterleitung im Mesh-Netzwerk statt, da das ursprüngliche Hop-Limit noch nicht durch das Routing reduziert wurde. Dementsprechend stimmt der ausstrahlende Sender mit dem Absender der Nachricht überein. Eine Filterung je Absender und dem maximalen Hop-Limit liefert eine Liste von Nachrichten mit direktem Empfang des Senders.

5 Ergebnisauswertung

Mit der Umsetzung der vorgestellten forensischen Ansätze konnten Daten gewonnen werden, welche in diesem Kapitel vorgestellt und ausgewertet werden. Anschließend werden diese Daten beurteilt.

Zunächst wird in Kapitel 5.1 auf die Ergebnisse zur Erkennung des vorhandenen Mesh-Netzwerkes eingegangen. Anschließend werden verschiedene Aspekte bezüglich der Teilnehmerortung ausgewertet. Daraufhin setzt sich das Kapitel 5.3 mit den Erkenntnissen aus der Protokollierung der Nachrichten auseinander. Weiterführend werden aus den soeben genannten Kapiteln Maßnahmen zur Erschwerung der Analyse geschlussfolgert. Somit können einerseits mögliche Handlungsweisen von Kriminellen, aber auch andererseits Ansätze zur Verbesserung der Sicherheit aufgezeigt werden. Schließlich wird nach dem Offenlegen technischer Hindernisse und Fehlerquellen ein Fazit gezogen.

5.1 Ergebnis der Netzwerkerkennung

Durch den Befehl `meshtastic --debug` wird eine Log-Ausgabe erzeugt. In der Abbildung 5.1 wird einen Ausschnitt einer solchen Ausgabe dargestellt. Deutlich abzulesen sind die Sender- und Empfänger-ID mit den Werten `2442722564` und `4294967295`. Umgerechnet in die Hexadezimalschreibweise betragen die Werte `0x9198FD04` und `0xFFFFFFFF`. Daraus können die Knoten-IDs `!0x9198FD04` und `!0xFFFFFFFF` unter Meshtastic® abgeleitet werden. Daraus lässt sich schließen, dass dieser Knoten einen Broadcast durchgeführt. Da der AES-Schlüssel der Kommunikation nicht bekannt ist, kann dementsprechend der direkte Inhalt der Nachrichten, welcher im Datenfeld ”payload” hinterlegt ist, nicht entziffert werden. Da es sich um ein Packet vom Typ ”TELEMETRY_APP” handelt, können in den Daten Netzwerkauslastung, Batteriestand und andere Telemetrieangaben erwartet werden [105].

Es lässt sich folglich festhalten, dass ein Meshtastic®-Netzwerk vor Ort besteht. Darüber hinaus kann, wie beschrieben, ein Teilnehmer erkannt werden. Durch die weiterführende Auswertung können alle Teilnehmer identifiziert und somit die Größe des Netzwerkes bestimmt werden. So konnten bereits nach 5 Minuten alle Meshtastic®-Knoten erfasst werden.

```
1  DEBUG file:mesh_interface.py _handleFromRadio line:492 Received from
       radio: packet {
2    from: 2442722564
3    to: 4294967295
4    decoded {
5      portnum: TELEMETRY_APP
6      payload: "\rj\n\310c\022\005\035\350\264\201?"
7    }
8    id: 1167301354
9    hop_limit: 3
10   priority: MIN
11 }
```

Abbildung 5.1: Debug-Eintrag der Konsolenausgabe

In einer Befragung nach dem praktischen Versuch wurde explizit nach einem weiteren, bisher ungenannten Mesh-Knoten gefragt. Hier ist keinem Probanden eine solche Mesh-Hardware aufgefallen. Die Detektion des Mesh-Netzwerkes von außen blieb folglich den Netzwerkteilnehmern verborgen.

Außerdem wurde eine Netzwerkerkennung mit einer neu programmierten Software durchgeführt. Diese sollte Meshtastic®-Pakete und andere empfangene Pakete zählen. Trotz Nutzung der identischen LoRa®-Konfiguration wie in der Meshtastic®-Firmware, wurde keine Nachricht empfangen. Weder Pakete von Meshtastic® noch andere LoRa®-Pakete konnten erfasst werden. Über den auch zuvor genutzten Meshtastic®-Knoten ist jedoch sichergestellt, dass in dem gleichen Zeitraum Meshtastic®-Pakete übertragen wurden. Es ist ungeklärt, warum trotz der korrekten Konfiguration keine Nachrichten erfasst werden konnten.

5.2 Ergebnis der Teilnehmerortung

Im Kapitel 3.2 wurden diverse Möglichkeiten erläutert, wie sich Teilnehmer identifizieren und lokalisieren lassen. In den anschließenden Kapiteln werden die Ergebnisse aufbereitet, die mit der Durchführung erfasst wurden.

5.2.1 Ergebnis der empfangenen Signalstärke von LoRa®

Mit der Aufzeichnung der Daten an der Station 1 konnten die Station 2, 4 und 6 direkt empfangen werden. Hierbei sind innerhalb des Aufzeichnungszeitraumes zwischen dem 16.01.2023 und dem 24.01.2023 984 Nachrichten von Station 2 und 1583 Nachrichten von Station 6 empfangen worden. Station 4 sendete mit der Rolle als ”ROUTER” nur zweimalig am Tag eine Nachricht direkt ins Netzwerk. Hiervon konnten drei Nachrichten mit einer Signalstärke von je -144 dBm erfasst werden.

In Tabelle 5.1 ist eine Übersicht der Werte hinterlegt. Deutlich zu erkennen ist die geringe Standardabweichung unter allen drei Knoten. Daraus lässt sich schließen, dass sich ein empfangenes Signal sehr zuverlässig bzgl. der empfangenen Signalstärke an einem Ort verhält. Insofern Sender, als auch Empfänger stationär sind, genügen demnach wenige Messungen zur Bestimmung der korrekten Empfangsstärke.

Tabelle 5.1: RSSI-Statistiken

Station	Distanz	Messwerte	RSSI im Mittel	RSSI im Median	Standardabweichung
2	353.14 m	984	-140.20 dBm	-140.0 dBm	3.0876 dBm
4	384.93 m	3	-144.00 dBm	-144.0 dBm	1.0000 dBm
6	234.02 m	1583	-118.43 dBm	-118.0 dBm	2.4839 dBm

5.2.2 Ergebnis der Ortung mittels BLE

Wie in Kapitel 4.5 beschrieben, fand eine Messung des Bluetooth-Signals mit einem Fairphone 4 statt. Hierfür wurde die Station 5 für die Messung ausgewählt, welche sich auf der vierten und damit obersten Etage des Hauses 6 der Hochschule befindet. Innerhalb dieser Etage konnte am gegenüberliegenden Treppenhaus das Signal mit -98 dBm gemessen werden. Mit dem Erfassungslimit von -100 dBm innerhalb der App ist das in diesem Fall gemessene Signal mit einer Entfernung

von 50 m zum Sender am Rand des Erfassbaren. Direkt im Flur neben der Station 5 konnte mit einer Entfernung von 1 m der maximale Wert von -72 dBm erfasst werden. Außerdem fand eine Messung des Signals in der dritten Etage direkt darunter statt. Hier konnte das Bluetooth-Signal mit -83 dBm empfangen werden. Generell wurde der Empfang des Signals durch die Decke oder Boden zwischen der dritten und vierten Etage gedämpft. So ließ sich das Signal bis zur Mitte des Gebäudes, also einer Entfernung von 30 m zum Sender, empfangen. Folglich lässt sich das Bluetooth-Signal im südlichen Treppenhaus auf der dritten Etage gar nicht erfassen. Darüber hinaus ist im zweiten Obergeschoss auf der kompletten Etage kein Empfang möglich.

Insofern eine ungefähre Position des Senders bekannt ist, kann mittels BLE dessen Position auf wenige Quadratmeter beschränkt werden. Im Falle der Station 5 kann auf eine Position im nördlichen Gebäudeabschnittes in der vierten Etage geschlussfolgert werden.

5.2.3 Ergebnis der Ortung mittels GPS

Innerhalb des Zeitraumes vom 16.01.2023 bis zum 24.01.2023 fand eine gesamtheitliche Aufzeichnung aller Nachrichten des Netzwerkes statt. Während dieses Zeitraumes haben nur die Stationen 3 und 5 eine GPS-Position ermitteln und übertragen können. Trotz der größeren 25 mm x 25 mm Keramikantenne konnten die Stationen 1, 2 und 4 kein GPS-Signal erfassen. Die Station 6 besitzt kein GPS-Modul.

Als einen Auszug der erfassten Daten zeigt Abbildung 5.2 alle empfangenen GPS-Positionen der Station 3 am 16.01.2023 an. Dabei wurden die einzelnen Markierungen farblich nach der Anzahl der Satelliten eingefärbt. Hierbei stellen neun Satelliten das Maximum und drei Satelliten das Minimum dar. Deutlich zu erkennen ist eine Konzentration der Koordinaten im und um das Haus 8 der Hochschule. Dennoch gibt es viele Positionsangaben mit signifikanten Abweichungen von mehreren hundert Metern von der tatsächlichen Lokalisierung. Deutlich zu erkennen ist eine Streuung der Positionen entlang einer Achse von Südwest nach Nordost. Diese folgt der Längsseite des Hauses 8. Jedoch ist unklar, ob diese Verteilung der GPS-Positionen auf die Gebäudewände zurückzuführen ist.

Unter der Auswertung aller Positionen innerhalb der acht Tage können die in der Tabelle 5.2 festgehaltenen statistischen Aussagen getroffen werden. So gibt es insgesamt 3134 Positionsangaben, wobei mehrheitlich fünf GPS-Statelliten zur Positionsbestimmung beigetragen haben. Deutlich zu erkennen ist eine Verbesserung der Positionsgenauigkeit mit Zunahme der empfangen Satelliten. Insbesondere mit neun Satelliten ermittelten Koordinaten ermöglichen bereits eine Zuordnung der Position zu einem Raum im Gebäude.

Weiterführend lassen sich aus allen GPS-Positionen ein Mittel und Median bilden. Hierfür wird getrennt ein Mittel bzw. Median von Longitude und Latitude ermittelt. Alle diese gemittelten Positionsangaben liegen innerhalb weniger Meter Abweichung zur realen Position. Hierbei deuten alle Koordinaten auf eine Position in der nordwestlichen Ecke des Hauses 8 hin. Folglich kann für die Station 3 durch die ausschließliche Nutzung der GPS-Angaben eine präzise Lokalisierung durchgeführt werden. Dennoch kann festgehalten werden, dass für diese Angabe entsprechend ausreichend viele Angaben benötigt werden. Insbesondere führt der Empfang von wenigen Satelliten zu ungenauen Angaben, welche jedoch durch eine hohe Anzahl von Koordinaten ausgeglichen werden kann.

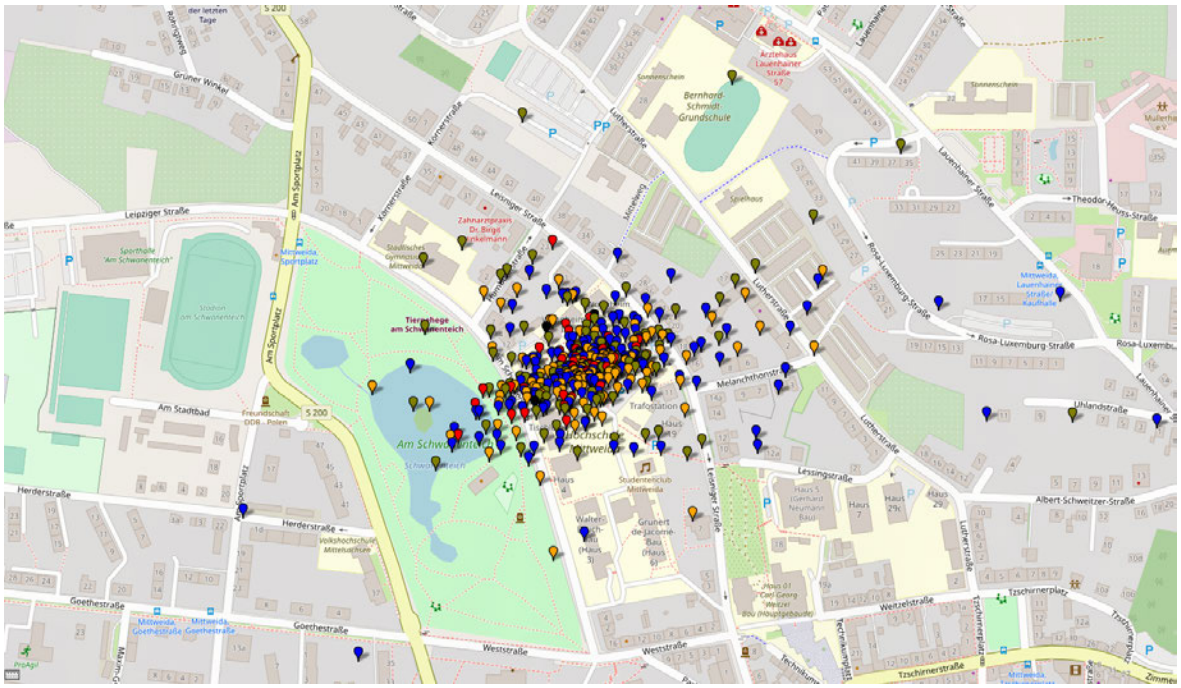


Abbildung 5.2: Übertragende GPS-Positionen der Station 3 am 16.01.2023, Satellitenanzahl: rot ≥ 6, orange $= 5$, blau $= 4$, oliv $= 3$, schwarz $=$ keine Angabe

Tabelle 5.2: GPS-Statistiken

Station	Satelliten-anzahl	Anzahl der Messwerte	Mittel der Abweichung	Median der Abweichung	Standardabweichung zur realen Position
3	3-9	3134	58.25 m	40.52 m	79.62 m
3	3	450	81.46 m	54.55 m	106.00 m
3	4	934	68.18 m	47.53 m	90.26 m
3	5	1028	51.08 m	37.50 m	48.30 m
3	6	505	38.90 m	29.42 m	34.18 m
3	7	158	33.40 m	23.97 m	28.21 m
3	8	35	29.52 m	20.00 m	28.60 m
3	9	6	7.60 m	7.75 m	2.87 m
5	3-7	2364	65.58 m	45.01 m	95.5 m
5	3	596	87.68 m	51.69 m	156.72 m
5	4	957	63.92 m	47.18 m	66.31 m
5	5	588	52.57 m	39.44 m	52.27 m
5	6	177	42.25 m	30.62 m	43.84 m
5	7	21	28.90 m	17.43 m	26.54 m

Tabelle 5.3: GPS-Position im Mittel und Median für Station 3

Satellitenanzahl	Position im Mittel	Position im Median
3-9	(50.9893223, 12.9701412)	(50.98929740, 12.9700702)
3	(50.9893724, 12.9703048)	(50.98932505, 12.9701473)
4	(50.9893340, 12.9701724)	(50.98930625, 12.9700335)
5	(50.9892869, 12.9700749)	(50.98928125, 12.9700541)
6	(50.9893138, 12.9700131)	(50.98929645, 12.9700257)
7	(50.9893138, 12.9700131)	(50.98929645, 12.9700257)
8	(50.9892785, 12.9700429)	(50.98924930, 12.9701241)
9	(50.9892916, 12.9701454)	(50.98927370, 12.9701233)

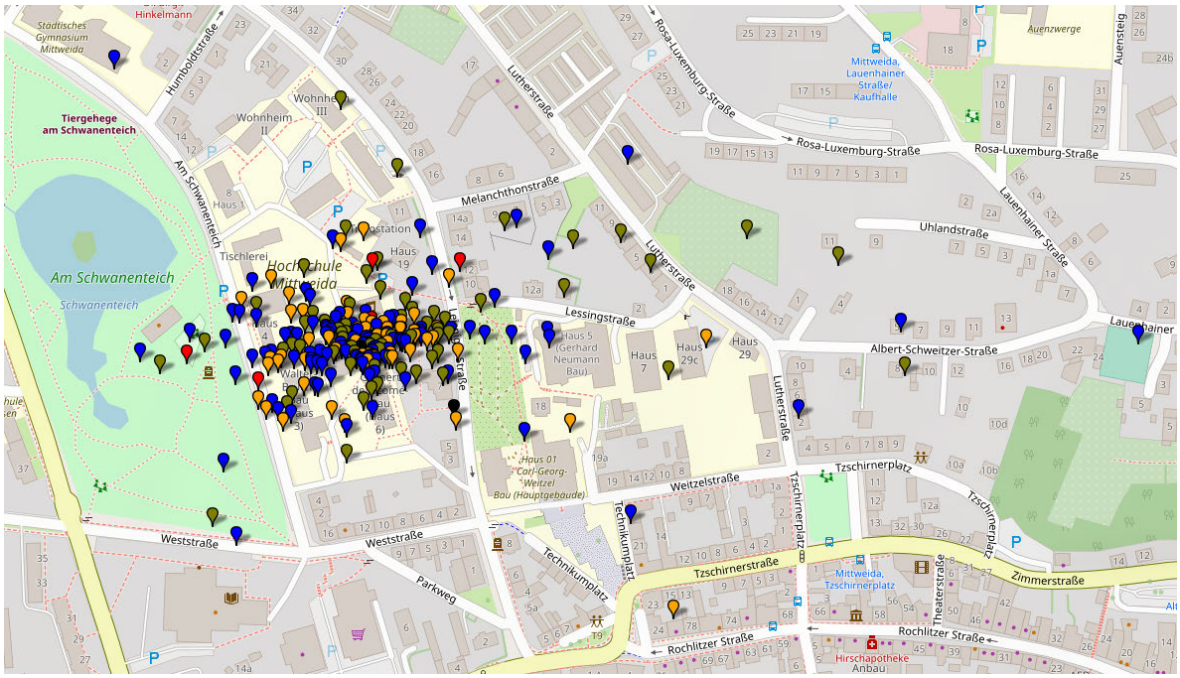


Abbildung 5.3: Übertragende GPS-Positionen der Station 5 am 16.01.2023,
Satellitenanzahl: rot ≥ 6, orange $= 5$, blau $= 4$, oliv $= 3$, schwarz $=$ keine Angabe

Analog zu Station 3 zeigt Abbildung 5.3 alle empfangenen GPS-Positionen der Station 5 am 16.01.2023 auf. Auch hier ist eine Konzentration der Positionen an einem Ort zu erkennen. Dieses Gebiet umfasst die Nordseite von Haus 6, dessen goldenen Anbau Studio B und der Studentenclub im Norden von Haus 6. Übereinstimmend mit Station 3 weichen auch bei Station 5 einige Koordinaten um hunderte Meter von der realen Position ab. Wie bereits bei Station 3 kann auch bei Station 5 eine größere Verteilung entlang der Ost-West-Angaben gegenüber Nord-Süd-Positionen festgestellt werden. Jedoch ist diese stärker in latitudinaler Richtung ausgeprägt als bei Station 3 im Haus 8.

Station 5 hat für die Positionsbestimmung drei bis sieben Satelliten empfangen. In Tabelle 5.2 sind die entsprechenden Statistiken aufgelistet. Dabei lässt sich feststellen, dass gleichfalls zu Station 3, ein Empfang von zusätzlichen GPS-Satelliten die Genauigkeit erhöht.

Eine Positionsbestimmung aus dem Mittelwert bzw. des Medians aller Koordinaten führt auch beim stationären Sender 5 zu sehr genauen Lokalisierungen. Diese sind in Tabelle 5.4 aufgelistet. Insbesondere der Median erreicht eine Abweichung von wenigen Metern. Dennoch wird in allen Fällen deutlich, dass sich der Sender im nördlichen Gebäudeabschnitts von Haus 6 nahe des Fahrstuhls befindet.

Tabelle 5.4: GPS-Position im Mittel und Median für Station 5

Satellitenanzahl	Position im Mittel	Position im Median
3-7	(50.9882375, 12.9709537)	(50.98822495, 12.97093240)
3	(50.9882845, 12.9709603)	(50.98822675, 12.97096515)
4	(50.9882264, 12.9709608)	(50.98822580, 12.97091150)
5	(50.9882095, 12.9709313)	(50.98821670, 12.97094655)
6	(50.9881982, 12.9709296)	(50.98821450, 12.97089880)
7	(50.9881982, 12.9709296)	(50.98821450, 12.97089880)

Für eine schnelle Reaktionsfähigkeit auf eine Ortsänderung beinhaltet die erstellte automatisierte Protokollierung der Netzwerknachrichten eine Überwachung des Standortes. Hier kann der Benutzer geographische Kreise festlegen. Sobald ein definierter Mesh-Teilnehmer diesen Kreis verlässt, wird der Nutzer der Protokollierungssoftware informiert. Alternativ kann auch informiert werden, sobald der geographische Kreis betreten wird. Folglich lassen sich Szenarien, wie ”eine Person betritt ein bestimmtes Haus” oder ”ein stationärer Knoten verlässt seine Position”, abbilden.

Jedoch konnte bereits im Vorfeld festgestellt werden, dass es auf Grund der geringen Genauigkeit einzelner Positionsmeldungen vermehrt zu Fehlalarmen kommt. So zeigt die Statistik in Tabelle 5.2, mit welchen Abweichungen zu rechnen ist. Ein Radius um eine zu beobachtende Position wäre so groß, dass deren Nutzen unbrauchbar wird. Beispielsweise werden bei Station 3 und 5 mehr als jede zweite Position mit einer Entfernung von über 40 Meter von der tatsächlichen Lage angegeben. Im Falle von Station 5 liegt jeder zweite Knoten sogar außerhalb eines Kreises mit dem Radius von 45 m. Eine Reduzierung des erlaubten Radius auf 40 Meter bedeutet also eine Auslösung eines Alarms bei jeder zweiten Positionsmeldung. Übertragen auf die in diesem Fall genutzte Konfiguration, würde diese Tatsache einen Alarm alle 10 Minuten auslösen. Selbst unter der Nutzung eines Radius von 500 m entstünden in den acht Tagen 12 Fehlalarme für Station 5 und 14 für Station 3. Folglich ist die Nutzung eines solchen Systems unbrauchbar.

5.2.4 Ergebnis der Ortung durch Probanden

Innerhalb der acht Tage investierten die Probanden drei bis zehn Stunden für die Lokalisierung der stationären Mesh-Hardware. In Tabelle 5.5 ist zu erfassen, mit welcher Genauigkeit sie die Position der einzelnen Stationen ermitteln konnten.

Deutlich ist eine bessere Lokalisierung der Stationen zu erkennen, welche eine GPS-Position im Netzwerk übertragen haben. So konnten die Station 3 und 5 auf ein Gebäude beschränkt werden. Darüber hinaus konnten einige Probanden die Positionsangabe auf wenige Meter eingrenzen.

Dagegen schwanken die Positionsangaben zu Station 4 so sehr, dass von geraten Angaben ausgegangen werden kann. Die Einstellung dieser Mesh-Hardware als ”ROUTER”-Rolle verringerte die Sendehäufigkeit auf zwei Nachrichten am Tag. Mit einer solchen geringen Informationsmenge war es den Teilnehmern nicht möglich, eine genaue Position zu erfassen.

Tabelle 5.5: Abweichung zur realen Position der durch Probanden erfassten Lokalisierung

Proband	Zeit	Station 1	Station 2	Station 3	Station 4	Station 5	Station 6
1	3 h	—	326.9 m	9.5 m	—	8.7 m	—
2	9 h	315.7 m	219.4 m	18.3 m	415.0 m	23.0 m	657.5 m
3	4,5 h	57.8 m	182.8 m	1.9 m	667.0 m	28.8 m	833.6 m
4	10 h	93.3 m	445.2 m	60.4 m	239.8 m	5.6 m	3.2 m
5	8,5 h	420.0 m	500.0 m	27.0 m	391.1 m	60.5 m	375.7 m
6	4 h	46.3m	418.1 m	11.2 m	372.7 m	15.4 m	327.7 m
7	8 h	29.0 m	420.6 m	8.1 m	398.8 m	29.2 m	90.7 m
Mittelwert	6.7 h	160.35 m	359 m	19.5 m	414.1 m	24.5 m	381.4 m

In einem an dem Versuch anschließenden, persönlichen Gespräch berichteten die Teilnehmer über ihr Vorgehen. Dabei war ein grundlegender Konsens, dass die Korrelation zwischen Signalstärke und Entfernung unzureichend für eine Trilateration sei. Insbesondere fehlten Vergleichswerte, auf welche Entfernung bei welcher Signalstärke zu schließen sei. Darauf adaptierend suchten die Probanden stets nach einem verbesserten Signalempfang, bis schließlich ein Maximum erfasst wurde. Folglich fand eine Lokalisierung entlang des Gradienten der Signalstärke statt. Die Startposition des Probanden wurde durch eine übermittelte GPS-Position oder willkürlich festgelegt. Somit identifizierten die Teilnehmer ein lokales Maximum des Signals in ihrer Umgebung. Ob dieses auch dem globale Maximum entsprach, hing markant von der gewählten Ausgangsposition ab.

In diesem Prozess nahmen die Probanden die bereitgestellten Informationen auf dem Display der Mesh-Hardware äquivalent mit jenen in der mobilen App angezeigten wahr. Jedoch sind laut Aussage der Teilnehmer die Informationen innerhalb der Android- oder iOS-App übersichtlicher aufbereitet. Insbesondere die Karte mit den eingezeichneten GPS-Markierungen wurde als sehr hilfreich wahrgenommen.

Im Folgenden wird auf die erzielten Ergebnisse der einzelnen Stationen und deren zugehöriges Ergebnis bei der Lokalisierung genauer eingegangen.

Station 1 Die Teilnehmer 3, 4, 6 und 7 konnten die Position erfolgreich auf weniger als 100 m eingrenzen. Jedoch vermuteten die Probanden 3, 6 und 7 die Mesh-Hardware auf der gegenüberliegenden Seite des Gebäudes und Teilnehmer 4 spekulierte auf eine Positionierung innerhalb der südlich gelegenen Turnhalle. Ersichtlich wird eine Ausrichtung dieser ermittelten Positionen in paralleler Weise zu den Gebäudewänden. So ist ein Empfang mit voller Signalstärke auf der gegenüberliegenden Seite des Innenhofes möglich.

Abweichend dazu lokalisierte der Proband 2 ein lokales Maximum, welches in der Nähe der Verkehrskreuzung Bahnhofstraße Ecke Heinrich-Heine-Straße auftritt. Der verbesserte Empfang an dieser Stelle wird durch die östliche Lücke im Gebäudekomplex verursacht. Gleichfalls bildet sich durch jene Unterbrechnung in der Gebäudestruktur ein lokales Maximum in der Nähe der Kleingartenanlage im Südosten, welches vom Teilnehmer 5 identifiziert wurde.

Auf der Abbildung 5.4 sind die von den Probanden erwarteten Positionen und deren eingeschätzter Radius abzulesen.

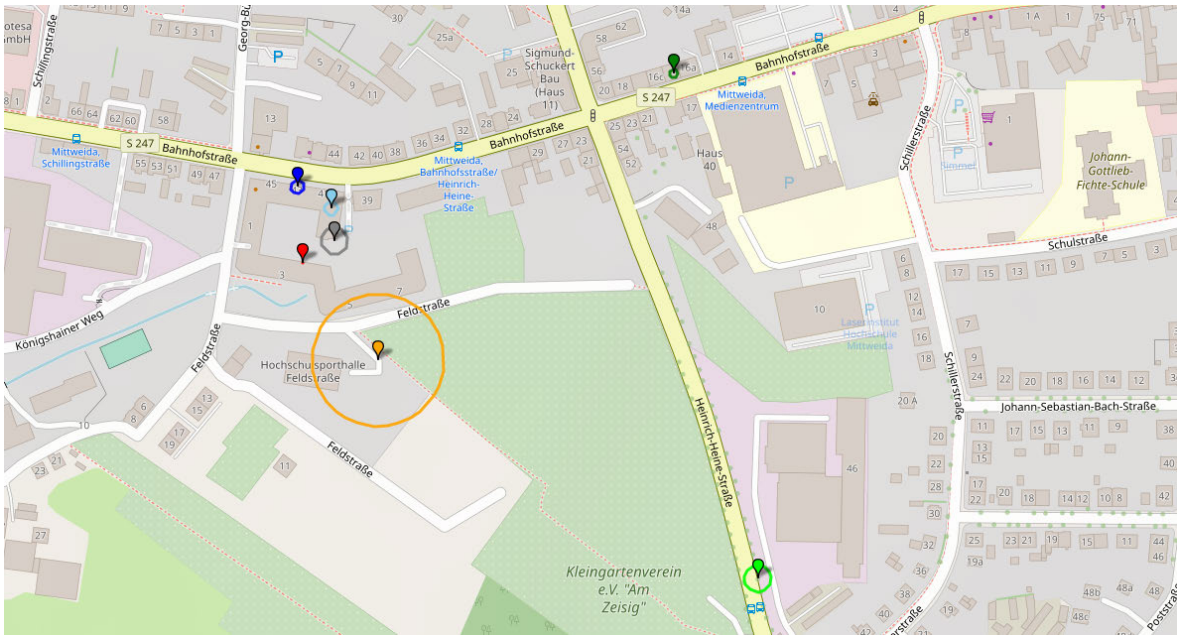


Abbildung 5.4: Ermittelte Position von Station 1, rot entspricht der realen Lage

Station 2 Station 2 ermöglichte durch ihre erhöhte Position einen guten Empfang innerhalb Mittweidas. Dementsprechend ermittelten die Teilnehmer vor allem die lokalen Maxima auf erhöhten Positionen. Folglich konzentrieren sich die Lokalisierungen auf Gebäude 6 der Hochschule Mittweida und das Medienzentrum in der Bahnhofstraße 42.

Dem gegenüber steht der Teilnehmer 3, welcher mit einer Suche am Bahnhof begann. Hier konnte ein lokales Maximum des Empfangs nahe der Goethestraße 25a erfasst werden.

Teilnehmer 1 konnte nicht ausreichend viel Zeit in die Suche dieser Station investieren. Folglich gab dieser lediglich eine grobe Schätzung im Zentrum der Hochschule an. Damit konnte von keinem Probanden die genaue Position der Station 2 identifiziert werden.

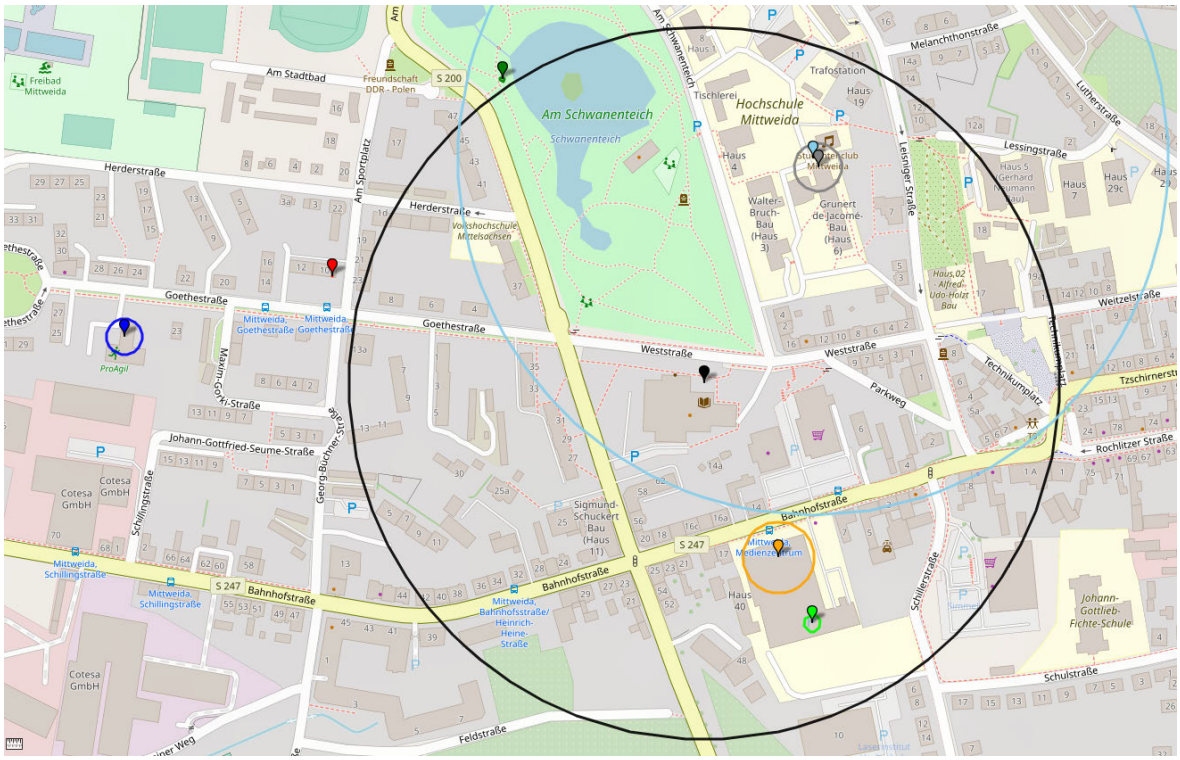


Abbildung 5.5: Ermittelte Position von Station 2, rot entspricht der realen Lage

Station 3 Station 3 teilte ihre GPS-Position, wie bereits in Abbildung 5.2 gezeigt, regelmäßig dem Netzwerk mit. Folglich konnten fast alle Teilnehmer sie in Haus 8 verorten. Darüber hinaus gelang es fünf der sieben Probanden eine korrekte Zuordnung der Mesh-Hardware zum Raum 107 zu erfassen. Ferner konnten zwei Teilnehmer das Endgerät am Fenster identifizieren. In Abbildung 5.6 sind die entsprechenden Angaben der Probanden visualisiert. Jedoch weißt die Positionsangabe des Teilnehmers 5 eine ungenaue Angabe auf. Obwohl hier der Raum 104 im Haus 8 als Standort angegeben wurde, befinden sich die genannten GPS-Koordinaten außerhalb des Gebäudes. Dennoch konnte auch dieser die Mesh-Hardware im Haus 8 der Hochschule korrekt lokalisieren.

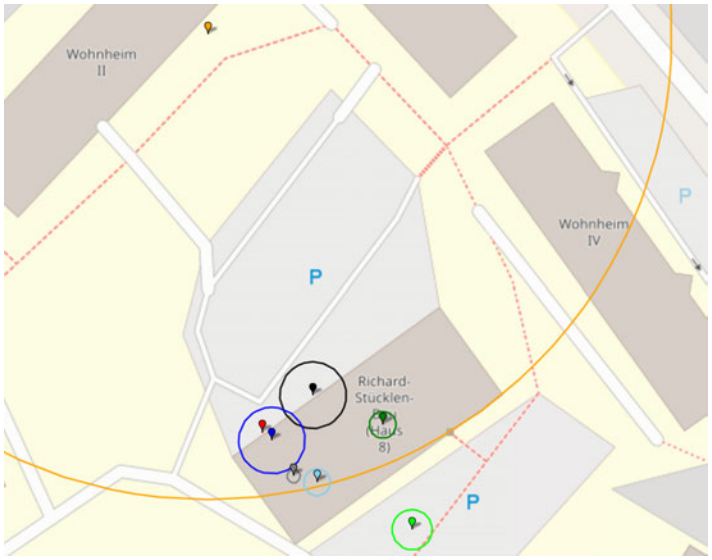


Abbildung 5.6: Ermittelte Position von Station 3, rot entspricht der realen Lage

Station 4 Deutlich in Abbildung 5.7 zu erkennen sind die willkürlichen Schätzungen der Position von Station 4. Auf Grund des geringen Sendeintervalls konnte keine zuverlässige Position durch das Verfahren der Teilnehmer erreicht werden.

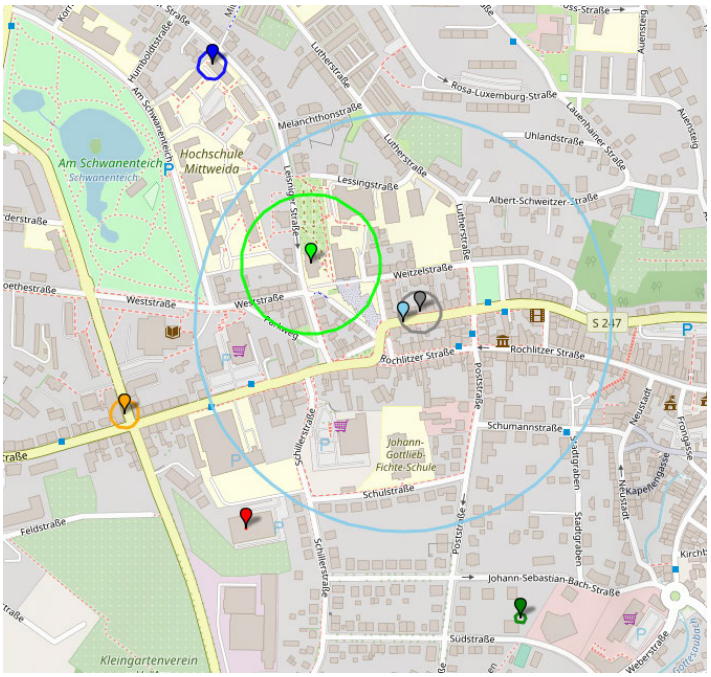


Abbildung 5.7: Ermittelte Position von Station 4, rot entspricht der realen Lage

Station 5 Dahingegen sendete Station 5 in regelmäßigen Abständen die per GPS errechnete Position. Mit dieser Information konnten alle Teilnehmer das korrekte Gebäude 6 der Hochschule als Standort identifizieren. Jedoch erfassten zwei Teilnehmer ein lokales Maximum in Studio B, dem goldenen Anbau im Westen des Hauses. Drei Teilnehmern gelang es mittels Bluetooth die Mesh-Hardware auf die nordöstliche Ecke des Gebäudes einzuschränken. Jedoch vermuteten diese Teilnehmer das Endgerät im dritten statt vierten Obergeschoss des Gebäudes.

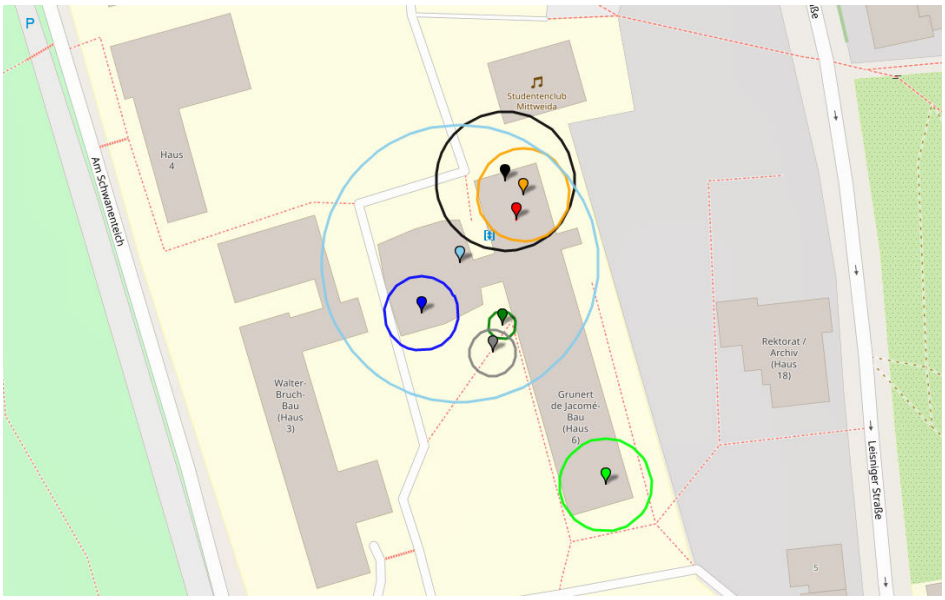


Abbildung 5.8: Ermittelte Position von Station 5, rot entspricht der realen Lage

Station 6 Schließlich kann in Abbildung 5.9 die ermittelten Positionen der Teilnehmer für Station 6 festgehalten werden. Für diese Station investierten einige Teilnehmer weniger Zeit. Dementsprechend schlecht sind die erzielten Ergebnisse. Dennoch konnten zwei Probanden das Gebäude an der Straßenecke als mögliche Lage eingrenzen. Jedoch entschied sich der Teilnehmer 7 schließlich zu Gunsten der Adresse Bahnhofstraße 32.

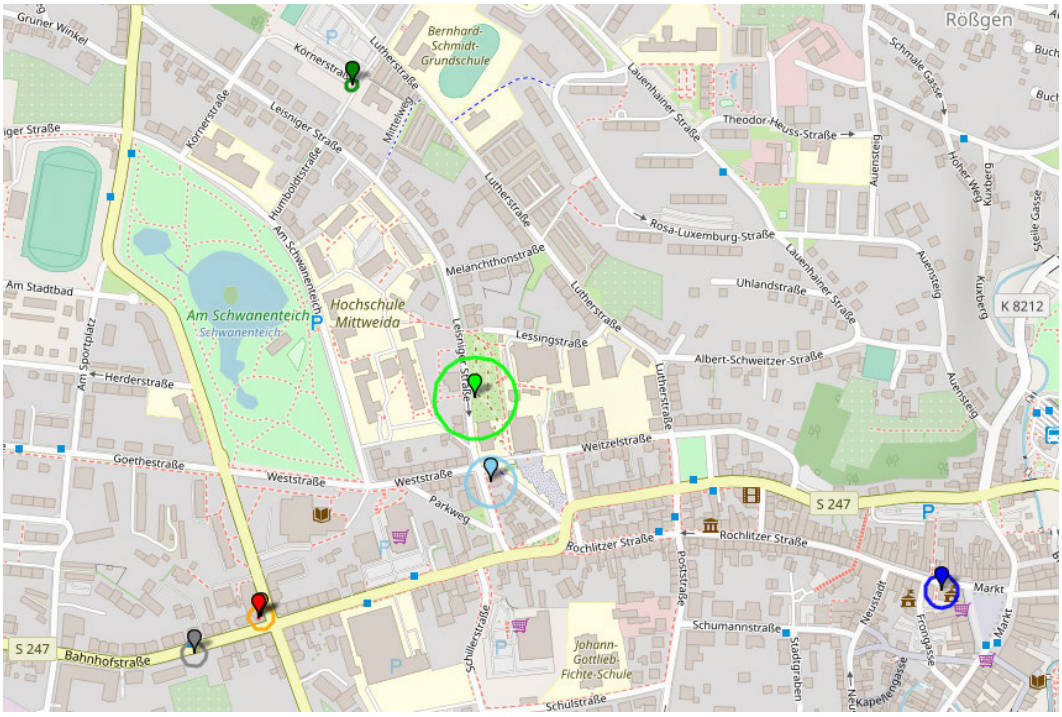


Abbildung 5.9: Ermittelte Position von Station 6, rot entspricht der realen Lage

5.3 Ergebnis der Nachrichtenprotokollierung

Mit Hilfe der automatisierten Nachrichtenerfassung konnten im Zeitraum der Aufzeichnung insgesamt 31.893 Nachrichten aufgezeichnet werden. In Tabelle 5.6 ist aufgeschlüsselt, wie sich diese Anzahl auf die einzelnen Netzwerkteilnehmer aufteilen.

Tabelle 5.6: Gesendete Nachrichtenanzahl

Mesh-Hardware	NodeInfo	Position	Telemetrie	Text	Summe
Mobil 1	332	180	315	1	828
Mobil 2	154	101	182	0	437
Mobil 3	246	230	228	0	704
Mobil 4	708	91	699	1	1.499
Mobil 5	638	671	659	1	1.969
Mobil 6	480	0	432	0	912
Mobil 7	528	0	500	28	1.056
Station 1	0	0	12.464	0	12.464
Station 2	779	0	807	0	1.586
Station 3	716	3.474	729	0	4.919
Station 4	18	9	22	0	49
Station 5	648	2.501	702	0	3.851
Station 6	808	0	811	0	1.619

Es fand eine zweimalige Unterbrechung der Aufzeichnung durch Verlust der USB-Verbindung statt. Diese waren am 18. Januar 2023 von 08:32 Uhr bis 20:17 Uhr für 11 Stunden und 45 Minuten und am 22. Januar 2023 von 16:32 Uhr bis zum 23. Januar 2023 um 10:40 Uhr für 18 Stunden und 8 Minuten. Dadurch ergeben sich einige Lücken in den Datensätzen. Jedoch genügen die aufgezeichneten Daten für die folgenden Analysen.

Deutlich zu erkennen ist das erhöhte Sendeverhalten von Teilnehmer 7 mit 28 Textnachrichten. Diese lassen sich auf dessen Arbeitsweise in der Ortung zurückführen. So erkannte der Proband, dass die gesendete Empfangsbestätigung einer Textnachricht einen Messwert bezüglich der Signalstärke liefert. Folglich war dieser Teilnehmer nicht ausschließlich auf das Sendeintervall der Stationen angewiesen.

5.3.1 Rekonstruktion des Netzwerkaufbaus

Da Meshtastic® stets den originalen Absender als Sender der Nachricht angibt, kann durch diesen nicht auf den ausstrahlenden Knoten geschlossen werden [130]. Jedoch kann unter Nutzung der bekannten Empfangsstärke und des Hop-Limits der Aufbau des Netzwerkes nachvollzogen werden. Dafür werden im ersten Schritt, wie bereits vollzogen die Empfangsstärken der Knoten gemessen, bei denen eine Weiterleitung ausgeschlossen werden kann. In diesem Netzwerk handelt es sich um die Angaben der Tabelle 5.1.

Anschließend können die Nachrichten mit einem um eins verringerten Hop-Wert untersucht werden. Im Bezug auf das vorliegende Netzwerk handelt es sich um einen Hop-Wert von drei. Die dabei erfassten RSSI-Messwerte können mit jenen aus Tabelle 5.1 abgeglichen werden. So bedeutet ein Empfang mit einer Signalstärke von -144 dBm, dass dieses von der Station 4 weitergeleitet wurde. Dementsprechend lassen sich die Daten zusätzlich nach Empfangsstärke filtern. Hierfür kann die Standardabweichung als Intervalllimit genutzt werden. Deutlich ist in Tabelle 5.7 zu erkennen, dass die so gebildeten Intervalle disjunkt sind.

Somit kann festgehalten werden, dass Station 1 die Nachrichten von Station 2 vorrangig über Station 6 empfangen hat, sofern diese nicht direkt erhalten wurden. Eine Ursache kann eine Nachrichtenkollision bei der Nachrichtenübertragung durch Station 2 sein. Weiterführend kann ermittelt werden, dass die Station 3 vorrangig über Station 2, aber auch 4 die Nachrichten im Netzwerk zur Station 1 versendet. Ferner konnte Station 4 einige Nachrichten nicht direkt an Station 1 übermitteln. Diese wurden zu 75% über die Station 6 und zu 25% über Station 2 weitergeleitet. Desweiteren konnte Station 5 Nachrichten nie direkt an Station 1 übertragen. Hier leiteten vorrangig die Stationen 2 und 4 die Nachrichten an Station 1 weiter. Und schließlich existieren einige wenige Nachrichten von Station 6, welche nur indirekt über Station 2 und 4 die Station 1 erreichten.

Aus den in Tabelle 5.7 beschriebenen Daten lässt sich eine Skizze des Netzwerkes anfertigen. Folglich stellt die Abbildung 5.10 das Mesh-Netzwerk dar. Dabei ist jedoch zu beachten, dass es sich um das Netzwerk aus der Sicht von Station 1 handelt. Es bestehen mit Sicherheit weitere Verbindungen zwischen Netzwerkknoten. Außerdem ist zu beachten, dass lediglich die stationären Knoten in der Betrachtung berücksichtigt wurden. Dennoch besteht die Möglichkeit, dass mobile Netzwerkteilnehmer ein Signal gleicher Empfangsstärke erzeugt haben. Ein solcher Fall würde zu einer falschen Zuordnung in der Auswertung führen.

Tabelle 5.7: Routing mit einmaliger Weiterleitung

Station	Routing über Station	Anzahl	RSSI Minimum	RSSI Maximum
2	4	4	-145.0 dBm	-143.1 dBm
2	6	215	-121.0 dBm	-116.0 dBm
3	2	886	-143.0 dBm	-137.0 dBm
3	4	182	-145.0 dBm	-143.1 dBm
3	6	3	-121.0 dBm	-116.0 dBm
4	2	5	-143.0 dBm	-137.0 dBm
4	6	14	-121.0 dBm	-116.0 dBm
5	2	679	-143.0 dBm	-137.0 dBm
5	4	218	-145.0 dBm	-143.1 dBm
5	6	4	-121.0 dBm	-116.0 dBm
6	2	9	-143.0 dBm	-137.0 dBm
6	4	2	-145.0 dBm	-143.1 dBm

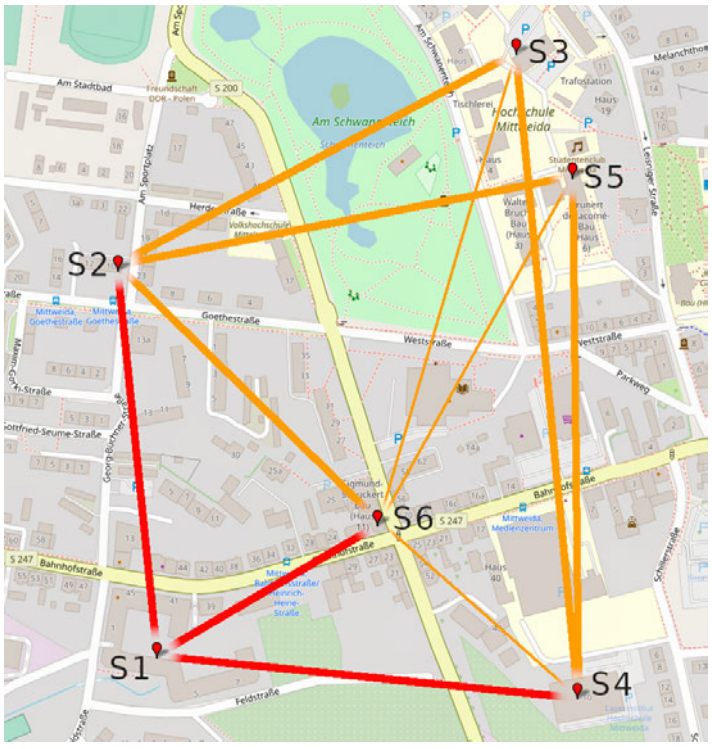


Abbildung 5.10: Rekonstruierter Aufbau des Mesh-Netzwerkes aus Sicht von Station 1, rot: direkte Verbindung, orange: indirekte Verbindung zu Station 1

Eine weiterführende Untersuchung mit einem reduzierten Hop-Wert ist nicht sinnvoll. Hierbei kann nicht nachvollzogen werden, über welchen Knoten die Nachricht zuerst geroutet wurde. Als einzige Ausnahme sind Knoten zu nennen, die in den vorherigen Schritten gar nicht erfasst wurden. Hier ist eine Nutzung der zuletzt erfassten Knoten als Router wahrscheinlich. Jedoch konnten in dem in Mittweida aufgebauten Netzwerk bereits nach einer Weiterleitung alle Knoten des Netzwerkes erfasst werden.

Wichtig zu erwähnen ist, dass die hier vollzogene Schlussfolgerung auch ohne Kenntnis des AES-Schlüssels durchgeführt werden kann. Für die Rekonstruktion des Netzwerkaufbaus wurden lediglich die unverschlüsselten Header-Informationen genutzt.

Schließlich können noch die Befehlsoptionen `--sendping` und `--traceroute` genannt werden. Mittels diesen wäre eine Rekonstruktion des Netzwerks, wie mit dem ICMP-Äquivalenten `ping` und `traceroute` möglich. Dabei wird ausgehend von einem Time to Live Wert, kurz TTL, von Eins ein Netzwerkteilnehmer versucht zu erreichen. In jedem Schritt wird der TTL-Wert um Eins erhöht, bis der gewünschte Knoten antwortet. Somit kann die Tiefe des Zielknotens von der eigenen Position im Netzwerk ermittelt werden. Unter Meshtastic® wird von einem Hop-Limit statt von TTL gesprochen. Dennoch ist die Funktionsweise gleich. Jedoch ist diese Vorgehensweise mit der Meshtastic®-Pythonschnittstelle nicht umsetzbar, da in allen Fällen eine Befehlsausführung auf einen Timeout hinaus lief. Jeder Teilnehmer im Netzwerk verweigerte eine Antwort auf den angeforderten Ping.

5.3.2 Aufzeichung des Batteriestands

Am Beispiel des Ladezustandes des Akkus wird demonstriert, wie eine Auswertung der erfassten Daten visuell aufbereitet werden kann. So wurde der komplette Zeitraum der Aufzeichnung in Abbildung 5.11 bzgl. des Ladezustands in einem Graphen dargestellt. Zunächst sind deutlich die Lücken am 18.01.2023 und 22.01.2023 in der Datenerfassung zu erkennen. Darüber hinaus sind Unterbrechungen in der Datenaufzeichnung am 17.01.2023 und am Abend des 22.01.2023 zu beobachten. Hierbei kann auf eine Ausschaltung der Mesh-Hardware aufgrund des geringen Ladezustandes geschlossen werden. Weiterhin ist ersichtlich, dass der genutzte Akku je nach Nutzungsintensität einen Betrieb von etwa 20-30 Stunden ermöglicht. Ferner kann ein Laden der Mesh-Hardware am Abend abgelesen werden, welche 2-3 Stunden für eine vollständig Aufladung benötigt.

Jedoch kann festgehalten werden, dass mit der Aufzeichung zur Mesh-Hardware Mobil 4 ein nahezu vollständiges Bild ermöglicht wird. Dem gegenüber steht beispielsweise Mobil 3. In diesem Fall bestehen große Lücken in der Datenerfassung, sodass für diesen Knoten keine Schlüsse gezogen werden können. In der Abbildung 5.12 wird der entsprechende Graph für die Mesh-Hardware Mobil 3 dargestellt. Durch ein Gespräch mit dem entsprechenden Teilnehmer konnte der Grund für die Unterbrechungen ermittelt werden. So wohnt der Teilnehmer in einer Senke nahe des Busbahnhofes, welches funktechnisch nur begrenzt durch andere Netzwerkknoten abgedeckt wird. Folglich konnten viele der ausgesendeten Nachrichten nicht im Netzwerk verbreitet und somit auch nicht durch Station 1 aufgezeichnet werden.

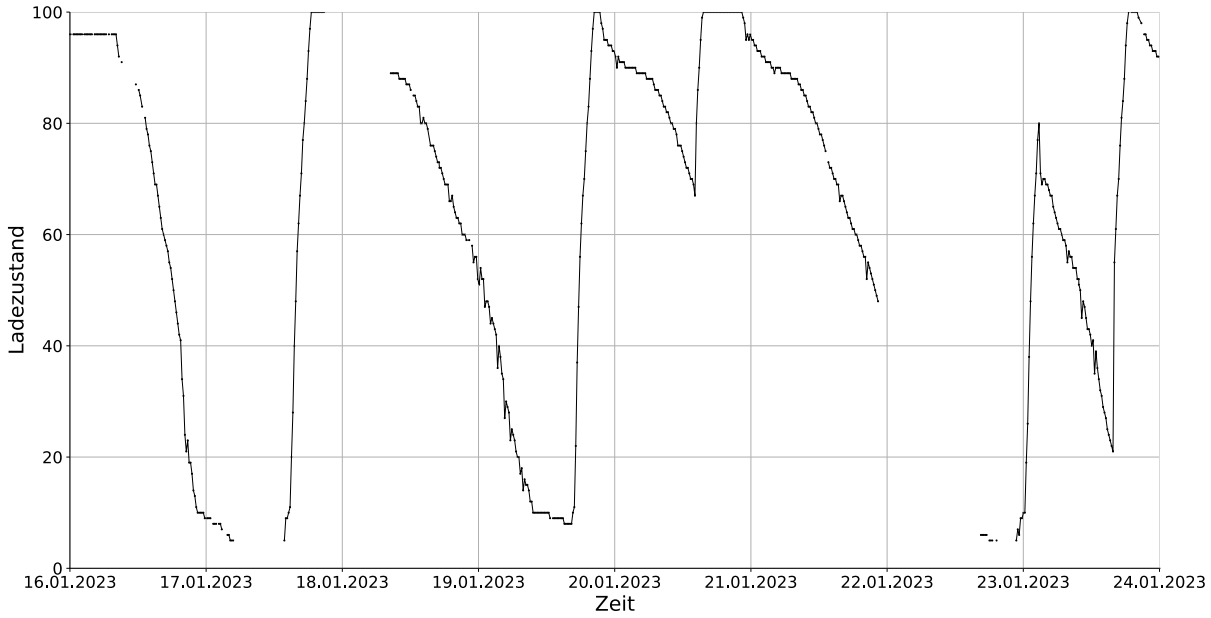


Abbildung 5.11: Ladezustand von Mobil 4

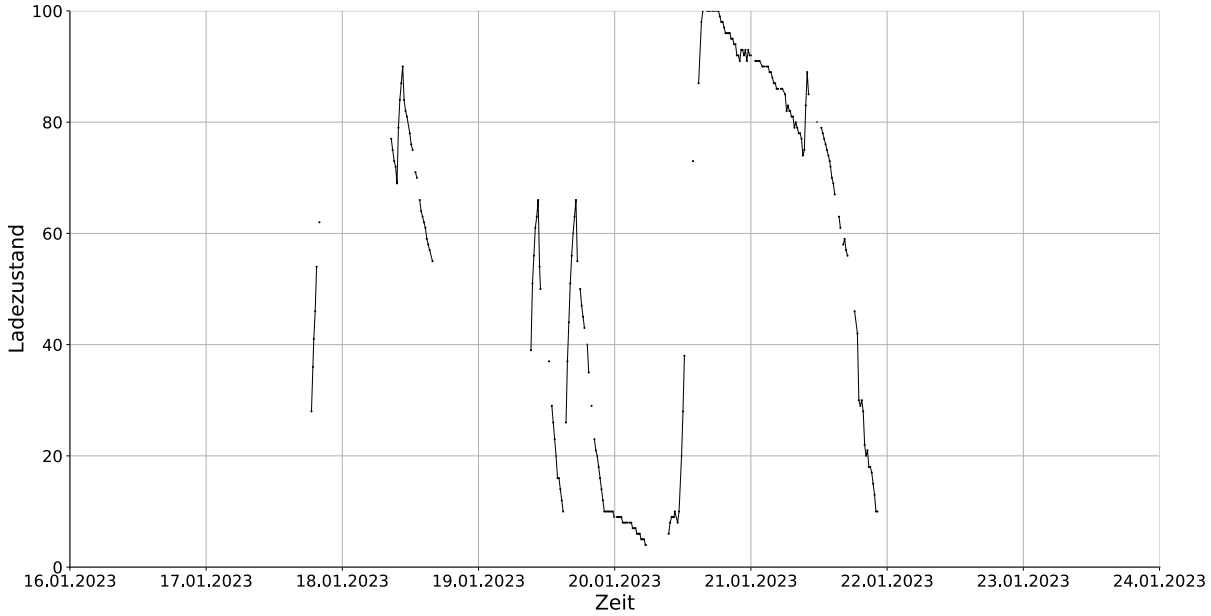


Abbildung 5.12: Ladezustand von Mobil 3

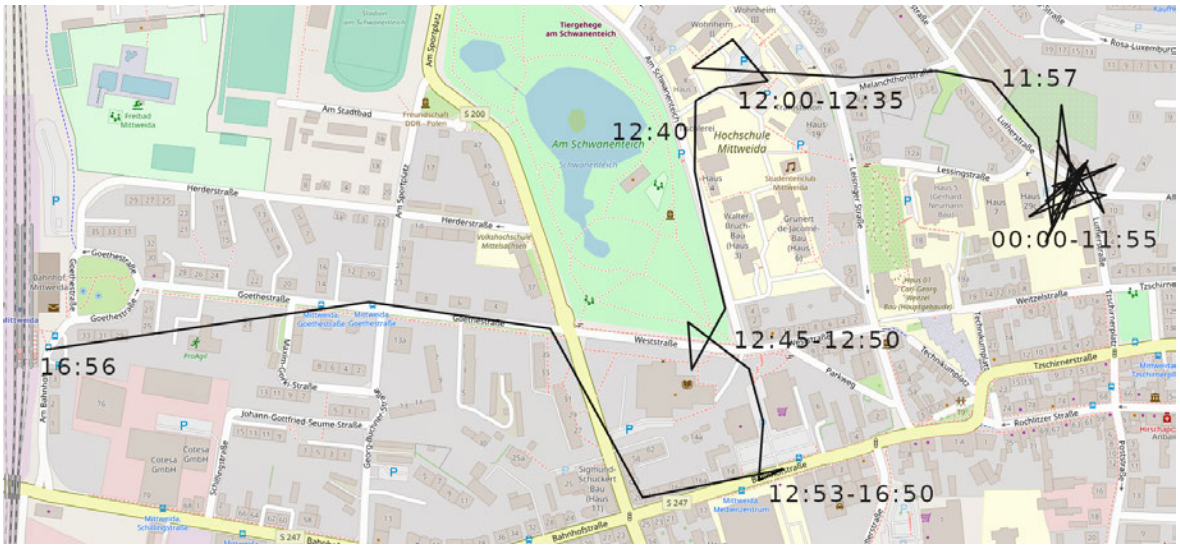


Abbildung 5.13: GPS-Positionsverlauf von Mobil 3 am 18.01.2023

5.3.3 Aufzeichnung der GPS-Koordinaten

In Kapitel 5.2.3 wurde bereits auf die Erfassung der GPS-Koordinaten über mit dem intern verbauten GPS-Empfänger eingegangen. Darüber hinaus kann Meshtastic® die GPS-Position auch mittels des verknüpften Mobiltelefons ermitteln [76]. Als Beispiel einer solchen Datenerfassung soll hier der Teilnehmer mit dem mobilen Mesh-Hardware 3 dienen. Hier zeigt die Abbildung 5.13 den Tagesverlauf am 18.01.2023 auf.

Deutlich zu erkennen ist der Wohnort in der Lutherstraße, welchen der Proband um 11:55 Uhr verließ. Von dort steuerte der Teilnehmer zunächst das Haus 8 der Hochschule an, in welchem er die Station 3 suchte. Anschließend traf sich der Proband vor dem Bibliotheksgebäude mit Kommilitonen, ehe er sich nach fünf Minuten weiter Richtung Haus 42 der Hochschule bewegte. Innerhalb dieses Gebäudes hielt sich der Proband bis 16:50 Uhr auf. So kann in Abbildung 5.12 erfasst werden, dass innerhalb dieses Zeitraumes das Endgerät geladen wurde. Schließlich erreichte er innerhalb von fünf Minuten den Bahnhof Mittweida. Dabei wurden die Straßen Bahnhofstraße, Heinrich-Heine-Straße und Goethestraße genutzt. Es kann davon ausgegangen werden, dass die Busverbindung A vom Medienzentrum zum Bahnhof genutzt wurde, da die geographische Aufzeichnung mit dessen Fahrplan übereinstimmt [131]. Anschließend kann der Teilnehmer die City Bahn 14 Richtung Chemnitz um 17:00 Uhr oder die Regional Bahn 45 Richtung Elsterwerda um 17:23 Uhr genommen haben [132, 133]. Da nach 16:56 Uhr keine weitere Position übertragen wurde ist es wahrscheinlicher, dass der Teilnehmer in Richtung Süden nach Chemnitz gefahren ist. In einem persönlichen Gespräch konnte dieser Ablauf verifiziert werden.

Es ist deutlich ersichtlich, dass die durch ein Mobiltelefon lokalisierte Positionsangaben wesentlich präziser sind, als durch den internen GPS-Empfänger. Einschränkend sind dabei jedoch Lokalisierungen innerhalb von Gebäuden. So konnte in diesem Beispiel keine Position innerhalb von Gebäude 42, aber auch nur ungenaue Positionen beim Wohnort übertragen werden.

5.3.4 Einschränkung der empfangenen Nachrichten

Durch einen Abgleich der im Netzwerk durch Station 1 empfangenen Textnachrichten mit den über die Mobiltelefone versendeten Nachrichten konnte festgestellt werden, dass nicht alle Nachrichten des Mesh-Netzwerkes erfasst werden. Ursache hierfür kann die Tatsache sein, dass ein Knoten eine Nachricht nicht weiterleitet, sofern für diese bereits eine Empfangsbestätigung erfasst wurde. Insbesondere tritt dieser Fall ein, wenn sich der Sender und Empfänger einer Nachricht unmittelbar in Funkreichweite befinden. In diesem Fall kann die direkt empfangene Nachricht augenblicklich durch den Zielknoten bestätigt werden. Eine anschließende Weiterleitung im Netzwerk wird dadurch unterbunden.

Dementsprechend werden Empfänger im gesamten Gebiet des Mesh-Netzwerkes benötigt, um alle übertragenden Nachrichten vollständig zu erfassen. Zum Erreichen dieses Ziels kann insbesondere Mesh-Hardware an besonders erhöhten Punkten zum Einsatz kommen. Damit kann eine Abdeckung des kompletten Gebietes mit einer geringeren Zahl an Meshtastic®-Knoten realisiert werden.

5.4 Maßnahmen der Ermittlungserschwerung

Die in dieser Arbeit durchgeführte forensische Untersuchung orientierte sich an der Standardeinstellung der Meshtastic®-Firmware. Jedoch kann davon ausgegangen werden, dass Personen mit kriminellen Absichten eine Analyse des Netzwerkes bewusst erschweren werden. Im Folgenden sind einige Maßnahmen aufgelistet, wie dies erreicht werden könnte.

Sendehäufigkeit verringern Mit dem Senden jeder Nachricht werden Informationen preisgegeben. Durch die Verringerung der gesendeten Nachrichten, wird nicht nur die Zeit erhöht, die benötigt wird dieses zu erkennen. Außerdem wird eine Ortung einer Mesh-Hardware erschwert, insbesondere, wenn diese mobil sind. Deutlich belegen lässt sich der Effekt der verringerten Sendehäufigkeit an der Station 4 im Versuchsaufbau. Diese sendete lediglich zwei Nachrichten am Tag im Netzwerk aus. Folglich konnte diese Mesh-Hardware von allen Teilnehmern nicht lokalisiert werden.

WLAN oder BLE deaktivieren Nicht nur das Ausstrahlen von LoRa®-Signalen lässt sich verringern. So können WLAN oder BLE deaktiviert werden, falls diese nicht benötigt werden. Das ist insbesondere der Fall, wenn eine Mesh-Hardware nur als Empfänger, Router oder über eine kabelgebundene USB-Verbindung genutzt wird. Durch diese Maßnahme wird eine Ortung über die Funksignale im 2.4 GHz Spektrum verwehrt. Folglich ist eine Lokalisierung im Nahbereich erschwert.

GPS-Position nicht senden Eine übertragende GPS-Koordinate bietet einen guten Ansatzpunkt für eine Lokalisierung des Senders. Folglich sollte auf eine Übertragung dessen verzichtet werden, insofern dieser nicht benötigt wird.

Falsche GPS-Position senden Ergänzend zum vorangegangenen Punkt können falsche GPS-Angaben bewusst genutzt werden. Damit können plausible, aber falsche Koordinaten die Ermittlungen erschweren oder zumindest verzögern. Jedoch sollten diese zeitlich variiert werden, da in Kapitel 5.2.3 gezeigt wurde, dass die GPS-Koordinaten in gewissen Rahmen ungenau sind. Es ist jedoch abzuwägen, inwiefern diese Maßnahme einer Reduzierung der Sendehäufigkeit vorzuziehen ist, da sich diese grundlegend widersprechen.

Neue oder unbekannte Teilnehmer erkennen Insofern bekannt ist, welche Knoten im Netzwerk aktiv sind, kann sich eine Kommunikation auf diese beschränkt werden. Hierfür ließe sich in die Routing-Tabelle der Mesh-Hardware eine Liste von bekannten Teilnehmern integrieren. In dem Augenblick, in dem ein neuer Teilnehmer im Netzwerk aktiv ist, können Knoten über diese Änderung informiert werden. Anschließend können Nutzer entsprechende Handlungen vornehmen.

Änderung des Routing erkennen Unter der Annahme, dass jede Mesh-Hardware im Netzwerk stationär positioniert ist, kann davon ausgegangen werden, dass das Routingverhalten im Netzwerk unverändert bleibt. Folglich kann eine Änderung im Routenverhalten der Knoten auf einen zusätzlichen, ungewollten Mesh-Teilnehmer hinweisen. Diese Art der Detektion ist jedoch im praktischen sehr fehleranfällig. So kann beispielsweise eine Änderung der Luftfeuchtigkeit oder Störsender bereits den Empfang der LoRa®-Signale verändern. Hierdurch können sowohl neue Routen entstehen oder bestehende Verbindungen unterbrochen werden.

Anonyme Netzwerkknoten betreiben Insofern eine Mesh-Hardware an einem erhöten Punkt autonom betrieben werden soll, um dem Empfang der anderen Knoten untereinander zu verbessern, kann dieser anonym betrieben werden. So benötigt eine Mesh-Hardware nicht den AES-Schlüssel eines Kommunikationskanals, um Nachrichten im Netzwerk weiter zu leiten [130]. Folglich kann eine Mesh-Hardware ohne eine spezielle Kanaleinstellung die Flächenabdeckung eines Netzwerks verbessern. Jedoch liefert dieser aus der Ferne betriebene Knoten keine Rückschlüsse auf das restliche Netzwerk. Es besteht eine plausible Abstreitbarkeit, im englischen als plausible deniability bezeichnet. Ferner kann diese Mesh-Hardware von weiteren Meshtastic®-Netzwerken genutzt werden. Dementsprechend wird dieser Knoten einmalig eingerichtet und anschließend vollkommen autonom betrieben. Hierfür benötigt die Mesh-Hardware eine passende Stromversorgung, wie beispielsweise durch eine Solarzelle. Ein so einmalig installierter Netzwerkknoten wird anschließend selbst bei auftretenden Problemen ignoriert.

Diese Idee aufgreifend veröffentlichte Meshtastic® bereits mit Version 2.0.15 am 28.01.2023 die Geräterolle ”REPEATER” [134]. Diese dient ausschließlich der Weiterleitung von Meshtastic®-Nachrichten. Jedoch versendet die Mesh-Hardware selbst keine Nachrichten. Dementsprechend wird dieser Knoten im Netzwerk nicht gelistet und ist ausschließlich über die weitergeleiteten LoRa®-Signale zu orten. [11]

Individuelle LoRa®-Konfiguration verwenden Meshtastic® ermöglicht die Konfiguration selbst gewählter Parameter für Spreizfaktor, Codierungsrate und Bandbreite [135]. Insofern diese abweichend zu den durch Meshtastic® vorgegebenen Kanälen gewählt werden, wird eine Erkennung des Netzwerkes erschwert. So würde das Netzwerk mit einer Untersuchung der bekannten Einstellungen nicht erfasst werden.

Zutrittssensorik Der physische Zutritt zu der Hardware kann mittels Sensoren detektiert werden. Hierfür eignen sich Bewegungssensoren im Umfeld oder Kontakt- bzw. Magnetschalter im Gehäuse der jeweiligen Mesh-Hardware. Sofern ein Zutritt erkannt wird, können andere Netzwerkteilnehmer informiert werden. Alternativ ist es auch denkbar, dass die Konfiguration des Netzwerkknotens sich automatisiert auf Standardeinstellungen zurücksetzt. Letzteres kann jedoch bei einer falschen Detektion zu Empfangsproblemen im Netzwerk führen.

USB-Verbindung erkennen Gleichfalls wie der Zutritt kann auch eine Verbindung über USB detektiert werden [9]. Wie bereits bei einer Zutrittserkennung, können auch in diesem Fall andere Knoten über den Vorgang informiert werden.

Passwortsicherung des USB-Zugangs Ferner lässt sich ein Zugriff über die USB-Schnittstelle absichern. So ist eine Passwortabfrage vor der Nutzung dieser denkbar. Weiterführend kann bei einer falschen Passworteingabe die Konfiguration zurückgesetzt und andere Netzwerkteilnehmer über den Vorgang informiert werden. [9]

Nutzung des ”admin” Kanals detektieren Für eine Konfigurierung aus der Ferne unterstützt Meshtastic® die Angabe von Änderungen über einen ”admin”-Kanal. Insofern dessen AES-Schlüssel bekannt ist, kann folglich auch eine Änderung der Konfiguration durch Dritte vorgenommen werden. Jene Änderung kann von Netzwerkknoten erkannt und eine Benachrichtungung an andere Teilnehmer übertragen werden. Demzufolge können diese eine ungewollte Konfigurationsänderung der Mesh-Hardware detektieren.

Signal direktional abschirmen Jede in Kapitel 2.9.1 vorgestellte Mesh-Hardware verwendet eine omnidirektionale Antenne. Folglich kann deren Signal aus jeder Richtung empfangen werden. Insofern ein Empfang des Signals nur in einem bestimmten Bereich benötigt wird, kann mittels Abschirmung des Signals die Verbreitung auf diesen beschränkt werden. Hierfür genügt eine Aluminiumfolie zur effektiven Verhinderung der Signalausbreitung.

Header verschlüsseln Die Headerinformationen werden bei Meshtastic® stets unverschlüsselt übertragen. Dementsprechend könnte geprüft werden, wie diese ebenfalls verschlüsselt werden können. Dadurch beschränkt sich die Fähigkeit des Routens auf Knoten, die den Schlüssel zu dieser Kommunikation besitzen. Folglich können anonyme Knoten nicht mehr zur Vergrößerung des Netzwerkes genutzt werden. Gleichzeitig erfordert eine Analyse der empfangenen Nachrichten zwangsweise eine Entschlüsselung dieser. Dadurch erfordert der Betrieb einer Mesh-Hardware einen erhöhten Stromverbrauch. Jedoch lässt sich festhalten, dass eine Rekonstruktion des Netzwerkaufbaus, wie in Kapitel 5.3.1 beschrieben, durch einen verschlüsselten Header dem Analysten verwehrt bliebe.

AES-Schlüssel austauschen Zum Entschlüsseln der Nachrichten wird der entsprechende AES-Schlüssel benötigt. Sobald dieser erlangt wurde, kann jede anschließend versendete Nachricht dechiffriert werden. Außerdem ermöglicht dieser die Entschlüsselung alle aufgezeichneten Nachrichten, welche diesen Schlüssel benutzten. Daher empfiehlt es sich den AES-Schlüssel regelmäßig zu ändern. Hierfür sollte bestenfalls ein getrennter Kommunikationskanal genutzt werden, um zu verhindern, dass die Schlüsseländerung ebenfalls aufgezeichnet wird. Folglich wird mit der Nutzung eines weiteren Kommunikationskanals verhindert, dass ein erneuerter AES-Schlüssel bekannt wird. Alternativ besteht die Möglichkeit den ”admin”-Kanal selbst zu nutzen, um einen Schlüssel aus der Ferne zu tauschen. Ein Mitschnitt dieses Vorgangs sollte jedoch ausgeschlossen werden können.

Ferner bringt Meshtastic® die Idee eines zeitlich rotierenden Schlüssels auf. So wird nach Ablauf einer bestimmten Zeitspanne ein neuer AES-Schlüssel abgeleitet, welcher anschließend für die Kommunikation genutzt wird. [9]

Individuelle AES-Schlüssel für den ”admin”-Kanal Über den ”admin”-Kommunikationskanal lässt sich die Mesh-Hardware über das Mesh-Netzwerk konfigurieren. Somit bietet es sich an, für jedes Endgerät einen eigenen AES-Schlüssel festzulegen. Darüber hinaus sollte dieser von der verwaltenden Instanz nur gesichert hinterlegt sein. Demzufolge besteht die Möglichkeit jede Mesh-Hardware ferngesteuert umzukonfigurieren, jedoch kann dies nur durch die verwaltende Instanz geschehen. Eine Änderung der Konfiguration durch andere Mesh-Teilnehmer kann ausgeschlossen werden.

Netzwerk ohne Routing Insofern alle Knoten mit der Rolle ”CLIENT_MUTE” konfiguriert werden, findet keine Weiterleitung von Nachrichten im Netzwerk statt. Folglich können Nachrichten nur noch direkt übertragen werden. Somit findet keine Reduzierung des Hop-Wertes statt, wodurch dessen Reduzierung auf einen externen Meshtastic®-Knoten hinweisen würde. Jedoch reduziert sich die Nutzbarkeit des Netzwerkes auf eine geringe Reichweite. Funknetzwerke mit größerer Ausdehnung auch über oder um Hindernisse herum sind somit nicht mehr zu realisieren.

Änderung der Magic-Bytes Schließlich können die sogenannten Magic-Bytes innerhalb der Meshtastic®-Firmware geändert werden. Sofern von dem Standardwert `0x94 0xc3` abgewichen wird, können andere Meshtastic®-Knoten das Netzwerk nicht detektieren. Darüber hinaus sind forensische Analysten gezwungen, durch eine Auswertung der Funksignale diese Änderung zunächst zu detektieren. Folglich hilft diese Änderung nur gegen eine oberflächliche Analyse der LoRa®-Signale in der Umgebung. Außerdem können anonyme Meshtastic®-Stationen mit dieser Änderung erneut direkt einem Netzwerk zugeordnet werden. Eine plausible Abstreitbarkeit dieser Mesh-Hardware ist nicht mehr gegeben.

5.5 Technische Schwierigkeiten

Während der Durchführung des praktischen Versuches haben sich einige technische Probleme aufgezeigt. Im Folgenden sind diese aufgelistet.

Zurücksetzen der Konfiguration Insbesondere mit der Firmwareversion 1.2.95 erwies sich die Konfiguration der Mesh-Hardware als unzuverlässig. Mehrmals setzte sich die Konfiguration in einigen Einstellungsfeldern zurück. Dementsprechend musste nach einer Konfiguration stets deren Änderung bestätigt werden. In den aufgetretenen Fällen musste die Konfiguration erneut vorgenommen werden. Jedoch konnte nicht beobachtet werden, dass sich die Konfiguration ohne die Durchführung einer Konfigurationsanpassung veränderte. Die Rücksetzung fand nur während der Einstellungsänderung statt.

Hauptversionssprung Am 01. November 2022 veröffentlichte Meshtastic® die Firmware-Version 2.0. Mit diesem Versionssprung wurden vorrangig Nachrichtenübertragung hinsichtlich größerer Netzwerke optimiert. Darüber hinaus passt die Version 2.0 das Routingverhalten der Mesh-Hardware an. Gleichzeitig wurden umfassende Änderungen in der Nachrichtenstrukturierung über die Anpassung der Protobuf-Definitionen vorgenommen. Durch diese Veränderungen ist die Version 2.0 grundlegend inkompatibel zur vorherigen stabilen Version 1.2.95. Diese Änderungen erforderten eine erneute Betrachtung in dieser Arbeit. [136]

Stromversorgung Moderne Netzteile erfassen den Stromfluss des angeschlossenen Verbrauchers [137]. Insofern dieser zu viel oder zu wenig Strom verbraucht, greift eine Sicherheitsfunktion und deaktiviert den entsprechenden Anschluss. Insbesondere die auf geringen Stromverbrauch optimierte Mesh-Hardware führte zu unerwarteten Ausfällen der Netzwerkknoten. Bei Verwendung eines Akkus konnte die ausgefallene Stromversorgung überbrückt werden. Durch den gesunkenen Ladezustand wird mehr Strom angefordert, wodurch eine Stromversorgung erneut hergestellt wird. Jedoch war eine Nutzung des Akkus an Station 4 nicht genehmigt. Um dennoch einen dauerhaften Betrieb zu gewährleisten, wurde der Stromverbrauch dieses Knotens durch die Konfiguration angehoben. Hierfür wird eine Abschaltung des Displays verhindert und dieses mit voller Helligkeit betrieben. Darüber hinaus wird der Deep-Sleep-Modus des Moduls deaktiviert, indem die Betriebszeit nach dem Empfang einer Nachricht auf einen Tag eingestellt wird. Somit wechselt diese Mesh-Hardware nicht in einen energiesparenden Modus, solange mindestens eine Nachricht pro Tag versendet wird.

Instabile USB-Verbindung Für die automatisierte Erfassung der Nachrichten werden diese über eine USB-Verbindung übertragen. Sobald diese unterbrochen wird, findet anschließend keine Datenübertragung mehr statt. Hier konnte insbesondere im Zusammenhang mit der Lora32-V2.0-Hardware eine Instabilität erfasst werden. So eignet sich diese Mesh-Hardware für eine automatisierte Aufzeichnung nicht. Auch mit dem als Station 1 verwendeten T-Beam-Modul fand zweimalig eine Unterbrechung der USB-Verbindung innerhalb des Acht-Tagezeitraums statt. Dementsprechend fehlen am 18.01.2023 und 22.01.2023 zusammengefasst fast 30 Stunden an Daten.

5.6 Fehlerquellen

Neben den technischen Schwierigkeiten existieren auch weitere Ursachen für Fehler in der Datenerfassung. Diese werden im Folgenden aufgelistet.

GPS-Position Die reale GPS-Position ist mittels `https://www.google.de/maps` in der topographischen Ansicht ermittelt. Die Lage der Hardware ist hier manuell durch die Positionierung des Cursors erfasst worden. Folglich ist davon auszugehen, dass es sich bei diesen Koordinaten nicht um die exakt tatsächliche Position handelt. Es kann eine Ungenauigkeit von wenigen Metern erwartet werden.

Gleichfalls nutzten die Probanden einen Kartendienst ihrer Wahl um die Position zu erfassen. Hierbei können Abweichungen zwischen den Diensten existieren. So sind beispielsweise die topologische und satelliten-basierte Ansicht bei `https://www.google.de/maps` nicht deckungsgleich. Teilweise wurde auch die durch das eigene Mobiltelefon ermittelte Position angegeben. Somit kann in beiden Fällen eine signifikante Abweichung zur tatsächlichen Position entstanden sein.

Unvollständige Aufzeichnung Die Erfassung der Nachrichten fand ausschließlich durch die Station 1 statt. Dementsprechend handelt es sich bei der Aufzeichnung um die Perspektive dieser Station auf das Mesh-Netzwerk. Folglich sind Nachrichten des Netzwerkes, die von dieser Station nicht empfangen wurden auch nicht in der Aufzeichnung enthalten.

Abweichende Softwarekonfiguration Wie bereits im vorangehenden Kapitel 5.5 beschrieben, besteht die Möglichkeit, dass die Konfiguration nicht mit der festgelegten Konfiguration übereinstimmt. Die im Anhang C hinterlegten Konfigurationen wurden aus der jeweiligen Mesh-Hardware ausgelesen. Dennoch kann nicht ausgeschlossen werden, dass diese Ausgabe fehlerfrei ist. Also könnte die Hardware eine abweichende Konfiguration von der Angezeigten verwenden.

5.7 Fazit

Abschließend kann ein Fazit zu den in dieser Arbeit betrachteten Aspekten gezogen werden. Hierzu wird nachstehend auf die Kernpunkte dieser Arbeit eingegangen.

Netzwerkexistenz nachweisen Insofern Nachrichten im Netzwerk versendet werden, können diese erfasst und somit das Bestehen eines Netzwerkes detektiert werden. Insbesondere unter der Verwendung von üblichen oder standardmäßigen Einstellungen der Meshtastic®-Firmware ist ein Nachweis innerhalb weniger Minuten realisierbar. Zum Erreichen dieses Ziels kann auf die Firmware der Hardware selbst zurückgegriffen werden.

Teilnehmer identifizieren Während der Erfassung des Netzwerkes an sich können gleichzeitig deren Teilnehmer identifiziert werden. Hierfür unterstützen die unverschlüsselten Header-Daten der übertragenen Nachrichten eine schnelle Detektion der individuellen Sender.

Teilnehmer orten Weiterführend konnte gezeigt werden, dass die jeweiligen Netzwerkknoten genau lokalisiert werden können. Insbesondere, wenn GPS-Koordinaten eine grobe Einschränkung ermöglichen, kann anschließend mittels LoRa® oder BLE eine feinere Ortung erfolgen. Unter der ausschließlichen Betrachtung von LoRa®-Funksignalen sollte jedoch darauf geachtet werden, dass es sich bei der deduzierten Lokalisierung nicht um ein lokales Maximum des Funksignals handelt.

Nachrichten protokollieren Eine Protokollierung der Nachrichten über die kabelgebundene USB-Schnittstelle erzielte eine breite Aufzeichnung der Nachrichten im Netzwerk. Jedoch konnte festgestellt werden, dass diese unvollständig sind, sofern eine Weiterleitung der Nachrichten im Netzwerk nicht erfolgt. Das ist insbesondere der Fall, falls Nachrichten direkt zugestellt werden können. In diesem Fall verhindert ein Erhalt der Empfangsbestätigung weiteres Routen im Netzwerk. Somit erfordert eine vollständige Aufzeichnung aller Nachrichten eine Aufzeichnung im gesamten aufgespannten Bereich des Mesh-Netzwerkes.

Dennoch lassen sich aus den protokollierten Daten forensische Rückschlüsse auf das Mesh-Netzwerk ziehen. Dazu zählen insbesondere der Netzwerkaufbau und Infomationen durch aggregieren von mehreren Nachrichten, wie beispielsweise ein Tagesablauf des Besitzers.

Verschlüsselung Bei der verwendeten AES-Verschlüsselung konnten keine Schwachstellen festgestellt werden. Insbesondere das Erzeugen von Schlüsseln wurde von Meshtastic®-Programmierern sicher implementiert. Jedoch existieren vordefinierte Schlüssel, auf welche für eine sichere Kommunikation verzichtet werden sollte. Insbesondere bei technisch unversierten oder ungeduldigen Nutzern kann eine Prüfung dieser Schlüssel zu einer erfolgreichen Entschlüsselung führen.

6 Ausblick

Fortführend an diese Arbeit lassen sich weitere Themen nennen, in denen die Forschung fortgesetzt werden kann. Im Folgenden wird auf einige Möglichkeiten der Weiterführung eingegangen.

Auswirkung der Gegenmaßnahmen auf die forensische Analyse Zunächst können die im Kapitel 5.4 vorgestellten Maßnahmen auf Effektivität getestet werden. Darüber hinaus kann festgestellt werden, wie sich diese auf die forensischen Ansätze aus Kapitel 3 auswirken.

Asymmetrische Verschlüsselung Ferner lässt sich festhalten, dass in der Meshtastic®-Community Pläne für eine asymmetrische Verschlüsselung existieren. Insbesondere die Verwendung der elliptischen Kurve Ed25519 ist hier angedacht [9]. Hier könnte einerseits aus der Sicht der Cybersicherheit analysiert werden, wie sich diese umsetzen lässt. Andererseits ließe sich anschließend untersuchen, wie die Verwendung von asymmetrischer Verschlüsselung die forensischen Ansätze beeinflussen.

Zukünftige Versionen von Meshtastic® Meshtastic® befindet sich in stetiger Weiterentwicklung. So wurden im Jahr 2023 bis zum 19. Februar bereits zehn neue Software-Versionen für die Firmware veröffentlicht [138]. Dementsprechend kann davon ausgegangen werden, dass auch in Zukunft neue Features veröffentlicht werden, die sowohl die Sicherheit, als auch die in dieser Arbeit vorgestellten forensischen Ansätze beeinflussen. Gleichfalls lassen sich diese Neuerungen hinsichtlich Sicherheit und Forensik untersuchen.

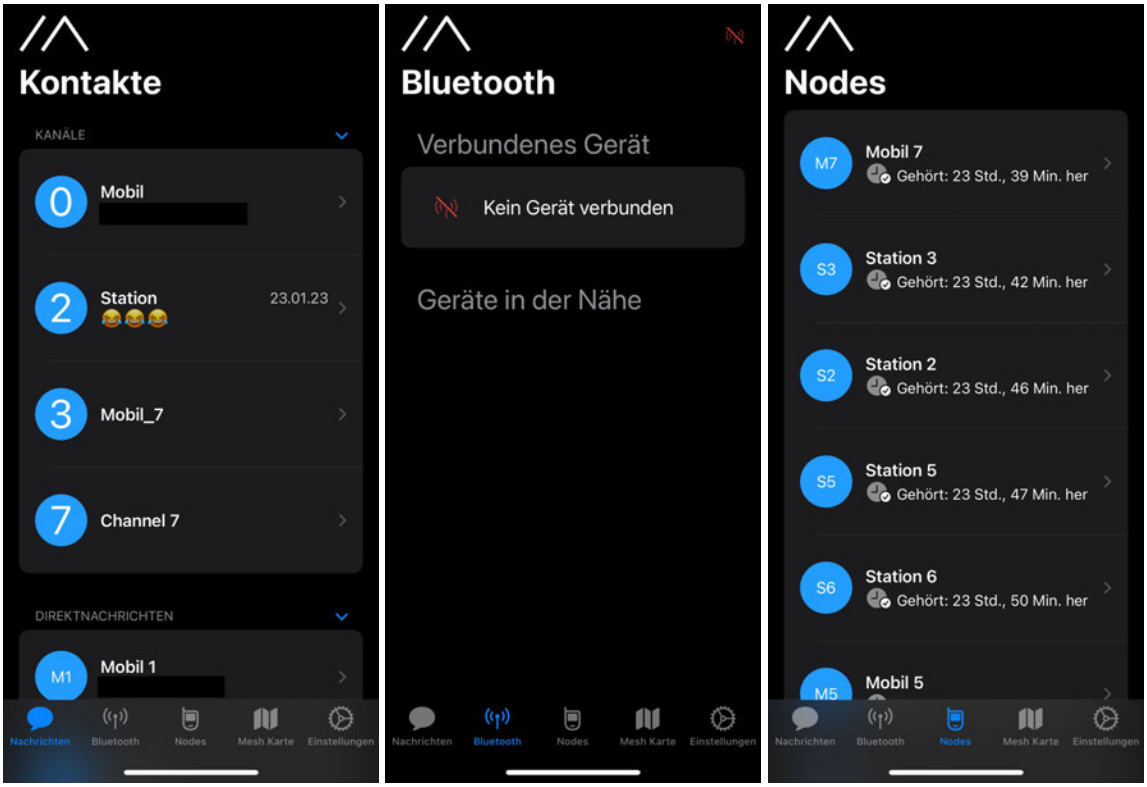
Verknüpfung von Messengern Eine weite Möglichkeit der Fortführung des Meshtastic®-Netzwerkes besteht darin, dieses als reines Transportnetzwerk anzusehen. Darauf aufbauend ließen sich externe Anwendungen anbinden. So ist denkbar, dass die Verschlüsselung und Darstellung von Textnachrichten vollständig von mobilen Apps vorgenommen wird. Demzufolge ließ sich prüfen, ob sich beispielsweise Chatanwendungen, wie Signal, mit der Infrastruktur von Meshtastic® vereinen ließen.

Reichweitenerweiterung durch Satelliten Die Firma ”FOSSA Systems S.L.” betreibt unter dem Namen ”FOSSASat-2 Evolved” Satelliten, welche die LoRa®-Technologie nutzen [139]. Insofern sich diese für Meshtastic® nutzen ließen, könnte die Netzwerkgröße signifikant erhöht werden. Somit wären Meshtastic®-Netzwerke bundesländerübergreifend denkbar. Hier könnte geprüft werden, inwiefern sich ein solches Netzwerk realisieren ließe.

Verzicht auf Magic-Bytes Mit Hilfe der Magic-Bytes lassen sich Meshtastic®-Netzwerke eindeutig identifizieren. Hier könnte geprüft werden, ob die Möglichkeit besteht, ein Mesh-Netzwerk ohne diese zu betreiben.

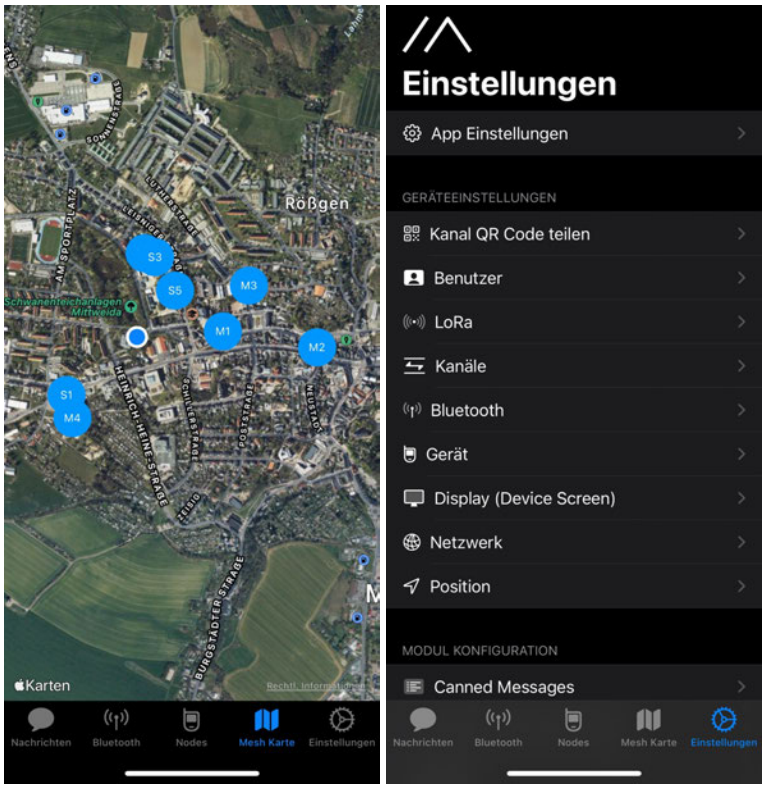
Weitere Mesh-Protokolle Schließlich wurden bereits im Kapitel 1.1 weitere Funkstandards wie ESP-WIFI-MESH, ESP-BLE-MESH und ESP-NOW genannt. Folglich können diese ebenfalls zum Aufbau von Mesh-Netzwerken genutzt werden. Dementsprechend kann auch bei diesen Technologien geprüft werden, wie sich diese forensisch analysieren lassen.

Anhang A: Screenshots der iOS-App



(a) Chatnachrichten **(b)** Gekoppelte Geräte **(c)** Teilnehmerübersicht

Abbildung A.1: Screenshots der iOS-App Reiter 1-3



(a) Karte mit Positionen **(b)** Konfiguration

Abbildung A.2: Screenshots der iOS-App-Reiter 4-5

Anhang B: Formular für Probanden

Beispiel

Ermittelte Position (GPS-Koordinaten)

(50.9867567, 12.9699221)

Sicherheit in der Genauigkeit der Position:

○ sehr unsicher ○ unsicher ○ ungewiss ○ sicher ✕sehr sicher

Geschätzter Radius mit hoher Sicherheit:
(Kleinster Radius um ihre Position, in welchem Sie sicher sind, dass sich dort der Sender befindet.)

3m

Geschätzte Höhe:

2m (1. Stock)

Sicherheit in der Genauigkeit der Höhe:

○ sehr unsicher ○ unsicher ○ ungewiss ○ sicher ✕sehr sicher

Optionale Angaben:

Haus 1 – Raum 123

Vorgehensweise und Messwerte:

- *13.01.2023 16:40 Uhr, Signal 36% bei (50.9841753, 12.9605821)*
- *13.01.2023 18:37 Uhr, Signal 15% bei (51.0281461, 13.0125764)*
- *14.01.2023 09:12 Uhr, Signal 42% bei (50.9881457, 12.9678462)*
- *...*
- *14.01.2023 13:28 Uhr, Empfang GPS Koordinate (50.9800879, 12.96678751)*
- *15.01.2023 19:01 Uhr, Bluetooth Signal bei (50.9884641, 12.97648312)*
- *15.01.2023 19:22 Uhr, Verfolgung des Bluetooth Signals zum Stärksten Empfang*
- *Gerät liegt vermutlich in der süd-östlichen Ecke von Raum 123 in Haus 1*
- *Sender konnte nicht visuell bestätigt werden*

Anhang C: Konfiguration der Mesh-Hardware

```
Preferences: {
  "device": { "serialEnabled": true },
  "position": { "positionBroadcastSecs": 600, "positionBroadcastSmartEnabled":
    true, "gpsEnabled": true, "gpsUpdateInterval": 30, "gpsAttemptTime": 30,
    "positionFlags": 995 },
  "power": { "waitBluetoothSecs": 60, "meshSdsTimeoutSecs": 7200, "sdsSecs":
    4294967295, "lsSecs": 300, "minWakeSecs": 10 },
  "network": { "ntpServer": "0.pool.ntp.org" },
  "display": { "screenOnSecs": 600 },
  "lora": { "usePreset": true, "region": "EU_868", "hopLimit": 4, "txEnabled":
    true, "txPower": 27 },
  "bluetooth": { "enabled": true, "fixedPin": 123456 }
}

Module preferences: {
  "mqtt": {},
  "serial": {},
  "externalNotification": {},
  "rangeTest": {},
  "telemetry": { "deviceUpdateInterval": 900, "environmentUpdateInterval": 900
    },
  "cannedMessage": {}
}

Channels:
  PRIMARY psk=secret { "psk": "clFjV7hqKh7By0mletjv4U1Ng+Y7RprpbxYbrIGBTks=",
    "name": "Mobil", "id": 10 }
  SECONDARY psk=secret { "psk":
    "VPDoCKQhkj5Y9V3UzHX03oNSdugAmmZ7LHkrrUGi55c=", "name": "admin", "id": 1 }
  SECONDARY psk=secret { "psk":
    "VPDoCKQhkj5Y9V3UzHX03oNSdugAmmZ7LHkrrUGi55c=", "name": "Mobil_1", "id": 2 }
```

Abbildung C.1: Konfiguration Mobil 1

```
Preferences: {
  "device": { "serialEnabled": true },
  "position": { "positionBroadcastSecs": 600, "positionBroadcastSmartEnabled":
    true, "gpsEnabled": true, "gpsUpdateInterval": 30, "gpsAttemptTime": 30,
    "positionFlags": 995 },
  "power": { "waitBluetoothSecs": 60, "meshSdsTimeoutSecs": 7200, "sdsSecs":
    4294967295, "lsSecs": 300, "minWakeSecs": 10 },
  "network": { "ntpServer": "0.pool.ntp.org" },
  "display": { "screenOnSecs": 600 },
  "lora": { "usePreset": true, "modemPreset": "VERY_LONG_SLOW", "region":
    "EU_868", "hopLimit": 4, "txEnabled": true, "txPower": 27 },
  "bluetooth": { "enabled": true, "fixedPin": 123456 }
}

Module preferences: {
  "mqtt": {},
  "serial": {},
  "externalNotification": {},
  "rangeTest": {},
  "telemetry": { "deviceUpdateInterval": 900, "environmentUpdateInterval": 900
    },
  "cannedMessage": {}
}

Channels:
  PRIMARY psk=secret { "psk": "clFjV7hqKh7By0mletjv4U1Ng+Y7RprpbxYbrIGBTks=",
    "name": "Mobil", "id": 10 }
  SECONDARY psk=secret { "psk":
    "VPDoCKQhkj5Y9V3UzHX03oNSdugAmmZ7LHkrrUGi55c=", "name": "admin", "id": 1 }
  SECONDARY psk=secret { "psk":
    "21WhrXakVaRuC9nYgVJm//ZUkcy93ewFeRKx00yhFTs=", "name": "Mobil_2", "id": 2 }
```

Abbildung C.2: Konfiguration Mobil 2

```
Preferences: {
  "device": { "serialEnabled": true },
  "position": { "positionBroadcastSecs": 600, "positionBroadcastSmartEnabled":
    true, "gpsEnabled": true, "gpsUpdateInterval": 30, "gpsAttemptTime": 30,
    "positionFlags": 995 },
  "power": { "waitBluetoothSecs": 60, "meshSdsTimeoutSecs": 7200, "sdsSecs":
    4294967295, "lsSecs": 300, "minWakeSecs": 10 },
  "network": { "ntpServer": "0.pool.ntp.org" },
  "display": { "screenOnSecs": 600 },
  "lora": { "usePreset": true, "region": "EU_868", "hopLimit": 4, "txEnabled":
    true, "txPower": 27 },
  "bluetooth": { "enabled": true, "fixedPin": 123456 }
}

Module preferences: {
  "mqtt": {},
  "serial": {},
  "externalNotification": {},
  "rangeTest": {},
  "telemetry": { "deviceUpdateInterval": 900, "environmentUpdateInterval": 900
    },
  "cannedMessage": {}
}

Channels:
  PRIMARY psk=secret { "psk": "clFjV7hqKh7By0mletjv4U1Ng+Y7RprpbxYbrIGBTks=",
    "name": "Mobil", "id": 11 }
  SECONDARY psk=secret { "psk":
    "VPDoCKQhkj5Y9V3UzHX03oNSdugAmmZ7LHkrrUGi55c=", "name": "admin", "id": 1 }
  SECONDARY psk=secret { "psk":
    "21WhrXakVaRuC9nYgVJm//ZUkcy93ewFeRKx00yhFTs=", "name": "Mobil_3", "id": 4 }
```

Abbildung C.3: Konfiguration Mobil 3

```
Preferences: {
  "device": { "serialEnabled": true },
  "position": { "positionBroadcastSecs": 600, "positionBroadcastSmartEnabled":
    true, "gpsEnabled": true, "gpsUpdateInterval": 30, "gpsAttemptTime": 30,
    "positionFlags": 995 },
  "power": { "waitBluetoothSecs": 60, "meshSdsTimeoutSecs": 7200, "sdsSecs":
    4294967295, "lsSecs": 300, "minWakeSecs": 10 },
  "network": { "ntpServer": "0.pool.ntp.org" },
  "display": { "screenOnSecs": 600 },
  "lora": { "usePreset": true, "region": "EU_868", "hopLimit": 4, "txEnabled":
    true, "txPower": 27 },
  "bluetooth": { "enabled": true, "fixedPin": 123456 }
}

Module preferences: {
  "mqtt": {},
  "serial": {},
  "externalNotification": {},
  "rangeTest": {},
  "telemetry": { "deviceUpdateInterval": 900, "environmentUpdateInterval": 900
    },
  "cannedMessage": {}
}

Channels:
  PRIMARY psk=secret { "psk": "WcRj+XEeyzeRLhYPdDWLu1Oxx3vJw3EA5VfI/W7zy74=",
    "name": "Station", "id": 11 }
  SECONDARY psk=secret { "psk":
    "VPDoCKQhkj5Y9V3UzHX03oNSdugAmmZ7LHkrrUGi55c=", "name": "admin", "id": 1 }
  SECONDARY psk=secret { "psk":
    "DqoaV6LM4HySIbT/3Io/mRnkWXhdPhJKZyRW8PRI7IY=", "name": "Mobil_4", "id": 5 }
```

Abbildung C.4: Konfiguration Mobil 4

```
Preferences: {
  "device": { "serialEnabled": true },
  "position": { "positionBroadcastSecs": 600, "positionBroadcastSmartEnabled":
    true, "gpsEnabled": true, "gpsUpdateInterval": 30, "gpsAttemptTime": 30,
    "positionFlags": 995 },
  "power": { "waitBluetoothSecs": 60, "meshSdsTimeoutSecs": 7200, "sdsSecs":
    4294967295, "lsSecs": 300, "minWakeSecs": 10 },
  "network": { "ntpServer": "0.pool.ntp.org" },
  "display": { "screenOnSecs": 600 },
  "lora": { "usePreset": true, "region": "EU_868", "hopLimit": 4, "txEnabled":
    true, "txPower": 27 },
  "bluetooth": { "enabled": true, "fixedPin": 123456 }
}

Module preferences: {
  "mqtt": {},
  "serial": {},
  "externalNotification": {},
  "rangeTest": {},
  "telemetry": { "deviceUpdateInterval": 900, "environmentUpdateInterval": 900
    },
  "cannedMessage": {}
}

Channels:
  PRIMARY psk=secret { "psk": "clFjV7hqKh7By0mletjv4U1Ng+Y7RprpbxYbrIGBTks=",
    "name": "Mobil", "id": 10 }
  SECONDARY psk=secret { "psk":
    "VPDoCKQhkj5Y9V3UzHX03oNSdugAmmZ7LHkrrUGi55c=", "name": "admin", "id": 1 }
  SECONDARY psk=secret { "psk":
    "WcRj+XEeyzeRLhYPdDWLu1Oxx3vJw3EA5VfI/W7zy74=", "name": "Station", "id": 11
    }
  SECONDARY psk=secret { "psk":
    "uR+UoQyC50jLUKHhTol6IOmLFSBuEkRtj9cEyRN5pBc=", "name": "Mobil_5", "id": 6 }
```

Abbildung C.5: Konfiguration Mobil 5

```
Preferences: {
  "device": { "serialEnabled": true },
  "position": { "positionBroadcastSecs": 600, "positionBroadcastSmartEnabled":
    true, "gpsEnabled": true, "gpsUpdateInterval": 30, "gpsAttemptTime": 30,
    "positionFlags": 995 },
  "power": { "waitBluetoothSecs": 60, "meshSdsTimeoutSecs": 7200, "sdsSecs":
    4294967295, "lsSecs": 300, "minWakeSecs": 10 },
  "network": { "ntpServer": "0.pool.ntp.org" },
  "display": { "screenOnSecs": 600 },
  "lora": { "usePreset": true, "region": "EU_868", "hopLimit": 4, "txEnabled":
    true, "txPower": 27 },
  "bluetooth": { "enabled": true, "fixedPin": 123456 }
}

Module preferences: {
  "mqtt": { "address": "mqtt.meshtastic.org", "username": "meshdev",
    "password": "large4cats" },
  "serial": {},
  "externalNotification": {},
  "rangeTest": {},
  "telemetry": { "deviceUpdateInterval": 900, "environmentUpdateInterval": 900
    },
  "cannedMessage": {}
}

Channels:
  PRIMARY psk=secret { "psk": "clFjV7hqKh7By0mletjv4U1Ng+Y7RprpbxYbrIGBTks=",
    "name": "Mobil", "id": 10 }
  SECONDARY psk=secret { "psk":
    "VPDoCKQhkj5Y9V3UzHX03oNSdugAmmZ7LHkrrUGi55c=", "name": "admin", "id": 1 }
  SECONDARY psk=secret { "psk":
    "WcRj+XEeyzeRLhYPdDWLu1Oxx3vJw3EA5VfI/W7zy74=", "name": "Station", "id": 11
    }
  SECONDARY psk=secret { "psk":
    "Pxifsw3ny+06kLGvwnkyVsq/5fobqaImNsPA1qgUjZg=", "name": "Mobil_6", "id": 7 }
```

Abbildung C.6: Konfiguration Mobil 6

```
Preferences: {
  "device": { "serialEnabled": true },
  "position": { "positionBroadcastSecs": 600, "positionBroadcastSmartEnabled":
    true, "gpsEnabled": true, "gpsUpdateInterval": 30, "gpsAttemptTime": 30,
    "positionFlags": 995 },
  "power": { "waitBluetoothSecs": 60, "meshSdsTimeoutSecs": 7200, "sdsSecs":
    4294967295, "lsSecs": 300, "minWakeSecs": 10 },
  "network": { "ntpServer": "0.pool.ntp.org" },
  "display": { "screenOnSecs": 600 },
  "lora": { "usePreset": true, "region": "EU_868", "hopLimit": 4, "txEnabled":
    true, "txPower": 27 },
  "bluetooth": { "enabled": true, "fixedPin": 123456 }
}

Module preferences: {
  "mqtt": { "address": "mqtt.meshtastic.org", "username": "meshdev",
    "password": "large4cats" },
  "serial": {},
  "externalNotification": {},
  "rangeTest": {},
  "telemetry": { "deviceUpdateInterval": 900, "environmentUpdateInterval": 900
    },
  "cannedMessage": {}
}

Channels:
  PRIMARY psk=secret { "psk": "clFjV7hqKh7By0mletjv4U1Ng+Y7RprpbxYbrIGBTks=",
    "name": "Mobil", "id": 10 }
  SECONDARY psk=secret { "psk":
    "VPDoCKQhkj5Y9V3UzHX03oNSdugAmmZ7LHkrrUGi55c=", "name": "admin", "id": 1 }
  SECONDARY psk=secret { "psk":
    "WcRj+XEeyzeRLhYPdDWLu1Oxx3vJw3EA5VfI/W7zy74=", "name": "Station", "id": 11
    }
  SECONDARY psk=secret { "psk":
    "g7V7Qa9mhOF0YWnvLIbwJw0iMh/M6kkXZH/xfPKZW0U=", "name": "Mobil_7", "id": 8 }
```

Abbildung C.7: Konfiguration Mobil 7

```
Preferences: {
  "device": { "serialEnabled": true },
  "position": { "positionBroadcastSecs": 600, "positionBroadcastSmartEnabled":
    true, "gpsEnabled": true, "gpsUpdateInterval": 30, "gpsAttemptTime": 30,
    "positionFlags": 995 },
  "power": { "waitBluetoothSecs": 60, "meshSdsTimeoutSecs": 7200, "sdsSecs":
    4294967295, "lsSecs": 300, "minWakeSecs": 10 },
  "network": { "ntpServer": "0.pool.ntp.org" },
  "display": { "screenOnSecs": 600 },
  "lora": { "usePreset": true, "modemPreset": "LONG_SLOW", "region": "EU_868",
    "hopLimit": 4, "txEnabled": true, "txPower": 27 },
  "bluetooth": { "enabled": true, "fixedPin": 123456 }
}

Module preferences: {
  "mqtt": {},
  "serial": {},
  "externalNotification": {},
  "rangeTest": {},
  "telemetry": { "deviceUpdateInterval": 900, "environmentUpdateInterval": 900
    },
  "cannedMessage": {}
}

Channels:
  PRIMARY psk=secret { "psk": "WcRj+XEeyzeRLhYPdDWLu1Oxx3vJw3EA5VfI/W7zy74=",
    "name": "Station", "id": 11 }
  SECONDARY psk=secret { "psk":
    "clFjV7hqKh7By0mletjv4U1Ng+Y7RprpbxYbrIGBTks=", "name": "Mobil", "id": 10 }
  SECONDARY psk=secret { "psk":
    "VPDoCKQhkj5Y9V3UzHX03oNSdugAmmZ7LHkrrUGi55c=", "name": "admin", "id": 1 }
  SECONDARY psk=secret { "psk":
    "VPDoCKQhkj5Y9V3UzHX03oNSdugAmmZ7LHkrrUGi55c=", "name": "Mobil_1", "id": 2 }
  SECONDARY psk=secret { "psk":
    "21WhrXakVaRuC9nYgVJm//ZUkcy93ewFeRKx00yhFTs=", "name": "Mobil_2", "id": 3 }
  SECONDARY psk=secret { "psk":
    "21WhrXakVaRuC9nYgVJm//ZUkcy93ewFeRKx00yhFTs=", "name": "Mobil_3", "id": 4 }
  SECONDARY psk=secret { "psk":
    "DqoaV6LM4HySIbT/3lo/mRnkWXhdPhJKZyRW8PRI7lY=", "name": "Mobil_4", "id": 5 }
  SECONDARY psk=secret { "psk":
    "uR+UoQyC50jLUKHhTol6IOmLFSBuEkRtj9cEyRN5pBc=", "name": "Mobil_5", "id": 6 }
```

Abbildung C.8: Konfiguration Station 1

```
Preferences: {
  "device": { "role": "ROUTER_CLIENT", "serialEnabled": true },
  "position": { "positionBroadcastSecs": 600, "positionBroadcastSmartEnabled":
    true, "gpsEnabled": true, "gpsUpdateInterval": 30, "gpsAttemptTime": 30,
    "positionFlags": 995 },
  "power": { "waitBluetoothSecs": 60, "meshSdsTimeoutSecs": 7200, "sdsSecs":
    4294967295, "lsSecs": 300, "minWakeSecs": 10 },
  "network": { "ntpServer": "0.pool.ntp.org" },
  "display": { "screenOnSecs": 600 },
  "lora": { "usePreset": true, "region": "EU_868", "hopLimit": 4, "txEnabled":
    true, "txPower": 27 },
  "bluetooth": { "enabled": true, "fixedPin": 123456 }
}

Module preferences: {
  "mqtt": {},
  "serial": {},
  "externalNotification": {},
  "rangeTest": {},
  "telemetry": { "deviceUpdateInterval": 900, "environmentUpdateInterval": 900
    },
  "cannedMessage": {}
}

Channels:
  PRIMARY psk=secret { "psk": "WcRj+XEeyzeRLhYPdDWLu1Oxx3vJw3EA5VfI/W7zy74=",
    "name": "Station", "id": 11 }
  SECONDARY psk=secret { "psk":
    "VPDoCKQhkj5Y9V3UzHX03oNSdugAmmZ7LHkrrUGi55c=", "name": "admin", "id": 1 }
```

Abbildung C.9: Konfiguration Station 2

```
Preferences: {
  "device": { "role": "CLIENT_MUTE", "serialEnabled": true },
  "position": { "positionBroadcastSecs": 600, "positionBroadcastSmartEnabled":
    true, "gpsEnabled": true, "gpsUpdateInterval": 30, "gpsAttemptTime": 30,
    "positionFlags": 995 },
  "power": { "waitBluetoothSecs": 60, "meshSdsTimeoutSecs": 7200, "sdsSecs":
    4294967295, "lsSecs": 300, "minWakeSecs": 10 },
  "network": { "ntpServer": "0.pool.ntp.org" },
  "display": { "screenOnSecs": 600 },
  "lora": { "usePreset": true, "region": "EU_868", "hopLimit": 4, "txEnabled":
    true, "txPower": 27 },
  "bluetooth": { "enabled": true, "fixedPin": 123456 }
}

Module preferences: {
  "mqtt": {},
  "serial": {},
  "externalNotification": {},
  "rangeTest": {},
  "telemetry": { "deviceUpdateInterval": 900, "environmentUpdateInterval": 900
    },
  "cannedMessage": {}
}

Channels:
  PRIMARY psk=secret { "psk": "WcRj+XEeyzeRLhYPdDWLu1Oxx3vJw3EA5VfI/W7zy74=",
    "name": "Station", "id": 11 }
  SECONDARY psk=secret { "psk":
    "VPDoCKQhkj5Y9V3UzHX03oNSdugAmmZ7LHkrrUGi55c=", "name": "admin", "id": 1 }
```

Abbildung C.10: Konfiguration Station 3

```
Preferences: {
  "device": { "role": "ROUTER", "serialEnabled": true },
  "position": { "positionBroadcastSecs": 43200,
    "positionBroadcastSmartEnabled": true, "gpsEnabled": true,
    "gpsUpdateInterval": 86400, "gpsAttemptTime": 300, "positionFlags": 995 },
  "power": { "waitBluetoothSecs": 1, "meshSdsTimeoutSecs": 4294967295,
    "sdsSecs": 86400, "lsSecs": 86400 },
  "network": { "ntpServer": "0.pool.ntp.org" },
  "display": { "screenOnSecs": 4294967295 },
  "lora": { "usePreset": true, "region": "EU_868", "hopLimit": 4, "txEnabled":
    true, "txPower": 27 },
  "bluetooth": { "enabled": true, "fixedPin": 123456 }
}

Module preferences: {
  "mqtt": { "address": "mqtt.meshtastic.org", "username": "meshdev",
    "password": "large4cats" },
  "serial": {},
  "externalNotification": {},
  "rangeTest": {},
  "telemetry": { "deviceUpdateInterval": 43200, "environmentUpdateInterval":
    43200 },
  "cannedMessage": {}
}

Channels:
  PRIMARY psk=secret { "psk": "WcRj+XEeyzeRLhYPdDWLu1Oxx3vJw3EA5VfI/W7zy74=",
    "name": "Station", "id": 11 }
  SECONDARY psk=secret { "psk":
    "VPDoCKQhkj5Y9V3UzHX03oNSdugAmmZ7LHkrrUGi55c=", "name": "admin", "id": 1 }
```

Abbildung C.11: Konfiguration Station 4

```
Preferences: {
  "device": { "role": "ROUTER_CLIENT", "serialEnabled": true },
  "position": { "positionBroadcastSecs": 600, "positionBroadcastSmartEnabled":
    true, "gpsEnabled": true, "gpsUpdateInterval": 30, "gpsAttemptTime": 30,
    "positionFlags": 995 },
  "power": { "waitBluetoothSecs": 60, "meshSdsTimeoutSecs": 7200, "sdsSecs":
    4294967295, "lsSecs": 300, "minWakeSecs": 10 },
  "network": { "ntpServer": "0.pool.ntp.org" },
  "display": { "screenOnSecs": 600 },
  "lora": { "usePreset": true, "region": "EU_868", "hopLimit": 4, "txEnabled":
    true, "txPower": 27 },
  "bluetooth": { "enabled": true, "fixedPin": 123456 }
}

Module preferences: {
  "mqtt": {},
  "serial": {},
  "externalNotification": {},
  "rangeTest": {},
  "telemetry": { "deviceUpdateInterval": 900, "environmentUpdateInterval": 900
    },
  "cannedMessage": {}
}

Channels:
  PRIMARY psk=secret { "psk": "WcRj+XEeyzeRLhYPdDWLu1Oxx3vJw3EA5VfI/W7zy74=",
    "name": "Station", "id": 11 }
  SECONDARY psk=secret { "psk":
    "VPDoCKQhkj5Y9V3UzHX03oNSdugAmmZ7LHkrrUGi55c=", "name": "admin", "id": 1 }
```

Abbildung C.12: Konfiguration Station 5

```
Preferences: {
  "device": { "serialEnabled": true },
  "position": { "positionBroadcastSecs": 600, "positionBroadcastSmartEnabled":
    true, "gpsUpdateInterval": 30, "gpsAttemptTime": 30, "positionFlags": 995 },
  "power": { "waitBluetoothSecs": 60, "meshSdsTimeoutSecs": 7200, "sdsSecs":
    4294967295, "lsSecs": 300, "minWakeSecs": 10 },
  "network": { "ntpServer": "0.pool.ntp.org" },
  "display": { "screenOnSecs": 600, "autoScreenCarouselSecs": 5 },
  "lora": { "usePreset": true, "region": "EU_868", "hopLimit": 4, "txEnabled":
    true, "txPower": 27 },
  "bluetooth": { "enabled": true, "fixedPin": 123456 }
}

Module preferences: {
  "mqtt": {
  "address": "mqtt.meshtastic.org", "username": "meshdev", "password":
    "large4cats" },
  "serial": {},
  "externalNotification": {},
  "rangeTest": {},
  "telemetry": { "deviceUpdateInterval": 900, "environmentUpdateInterval": 900
    },
  "cannedMessage": {}
}

Channels:
  PRIMARY psk=secret { "psk": "WcRj+XEeyzeRLhYPdDWLu1Oxx3vJw3EA5VfI/W7zy74=",
    "name": "Station", "id": 11 }
  SECONDARY psk=secret { "psk":
    "VPDoCKQhkj5Y9V3UzHX03oNSdugAmmZ7LHkrrUGi55c=", "name": "admin", "id": 1 }
```

Abbildung C.13: Konfiguration Station 6

Literaturverzeichnis

[1] Espressif. „ESP32“. (2016), Adresse: https://www.espressif.com/en/products/socs/esp32. (letzter Zugriff: 05.11.2022 00:53 Uhr).

[2] Espressif. „ESP8266“. (2018), Adresse: https://www.espressif.com/en/products/socs/esp8266. (letzter Zugriff: 05.11.2022 00:59 Uhr).

[3] R. P. Foundation. „Raspberry Pi Zero W“. (2017), Adresse: https://www.raspberrypi.com/products/raspberry-pi-zero-w/. (letzter Zugriff: 05.11.2022 01:02 Uhr).

[4] R. P. Foundation. „Raspberry Pi Pico“. (2022), Adresse: https://www.raspberrypi.com/products/raspberry-pi-pico/. (letzter Zugriff: 05.11.2022 01:05 Uhr).

[5] Espressif. „ESP-WIFI-MESH“. (2022), Adresse: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/esp-wifi-mesh.html. (letzter Zugriff: 05.11.2022 00:40 Uhr).

[6] Espressif. „ESP-BLE-MESH“. (2022), Adresse: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/bluetooth/esp-ble-mesh.html. (letzter Zugriff: 05.11.2022 00:41 Uhr).

[7] Meshtastic®. „Introduction“. (2022), Adresse: https://meshtastic.org/docs/about. (letzter Zugriff: 05.11.2022 00:42 Uhr).

[8] Meshtastic®. „Introduction“. (2023), Adresse: https://meshtastic.org/docs/introduction. (letzter Zugriff: 14.02.2023 13:20 Uhr).

[9] Meshtastic®. „Meshtastic Encryption“. (2023), Adresse: https://meshtastic.org/docs/overview/encryption. (letzter Zugriff: 09.02.2023 09:28 Uhr).

[10] Meshtastic®. „Position Configuration“. (2023), Adresse: https://meshtastic.org/docs/settings/config/position. (letzter Zugriff: 14.02.2023 13:48 Uhr).

[11] Meshtastic®. „Device Configuration“. (2023), Adresse: https://meshtastic.org/docs/settings/config/device. (letzter Zugriff: 14.02.2023 13:51 Uhr).

[12] Meshtastic®. „Send a message“. (2023), Adresse: https://meshtastic.org/docs/software/android/usage#send-a-message. (letzter Zugriff: 14.02.2023 13:56 Uhr).

[13] F. Hüning, *Embedded Systems für IoT*, ger, Ser. SpringerLink Bücher. 2019, S. 19–21, ISBN: 9783662579015.

[14] I. Bluetooth SIG. „Bluetooth Technologie-Übersicht“. (2023), Adresse: https://www.bluetooth.com/de/learn-about-bluetooth/tech-overview/. (letzter Zugriff: 18.02.2023 22:56 Uhr).

[15] J. Rech, *Wireless LANs 802.11-WLAN-Technologie und praktische Umsetzung im Detail ; 802.11a/h, 802.11b, 802.11g, 802.11i, 802.11n, 802.11d, 802.11e, 802.11f, 802.11s, 802.11ac, 802.11ad*, ger, 4., aktualisierte und erweiterte Auflage. 2012, S. 4–9, ISBN: 9783936931839.

[16] Espressif. „Wi-Fi Protocol Mode“. (2022), Adresse: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/wifi.html#wi-fi-protocol-mode. (letzter Zugriff: 24.11.2022 00:54 Uhr).

[17] L. Espressif Systems (Shanghai) Co., „ESP32 Series Datasheet v4.2“, S. 8–11, 2023. Adresse: https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf.

[18] M. Reichert, „GPS-Tracker gegen Holzklau“, *Westdeutscher Rundfunk (WDR)*, 2022. Adresse: https://www1.wdr.de/nachrichten/rheinland/holzklau-tracking-gps-100.html, (letzter Zugriff: 14.02.2023 14:31 Uhr).

[19] P. UG. „GPS Tracker für Holz“. (2023), Adresse: https://www.paj-gps.de/gps-tracker-fuer-holz/. (letzter Zugriff: 14.02.2023 14:35 Uhr).

[20] J. McNeff, „The global positioning system“, *IEEE Transactions on Microwave Theory and Techniques*, Jg. 50, Nr. 3, S. 645–648, 2002. DOI: 10.1109/22.989949.

[21] B. Hofmann-Wellenhof, *GNSS – Global Navigation Satellite Systems GPS, GLONASS, Galileo, and more*, eng, 1st ed. 2008. 2008, S. 397–400, ISBN: 3211730176.

[22] u-blox AG, „NEO-6 u-blox 6 GPS Modules“, S. 6, 2011. Adresse: https://content.u-blox.com/sites/default/files/products/documents/NEO-6_DataSheet_%28GPS.G6-HW-09005%29.pdf.

[23] A. Spiess. „#120 LoRa / LoRaWAN Range World Record Attempt. Will I succeed?“ (2017), Adresse: https://www.youtube.com/watch?v=adhWIo-7gr4. (letzter Zugriff: 07.11.2022 00:58 Uhr).

[24] Meshtastic und David. „Range Tests“. (2021), Adresse: https://meshtastic.org/docs/overview/range-tests#current-record. (letzter Zugriff: 30.01.2023 23:31 Uhr).

[25] Semtech. „What Is LoRa®“. (2022), Adresse: https://www.semtech.com/lora/what-is-lora. (letzter Zugriff: 07.11.2022 00:32 Uhr).

[26] M. Centenaro, L. Vangelista, A. Zanella und M. Zorzi, „Long-range communications in unlicensed bands: the rising stars in the IoT and smart city scenarios“, *IEEE Wireless Communications*, Jg. 23, Nr. 5, S. 60–67, 2016. DOI: 10.1109/MWC.2016.7721743. Adresse: https://ieeexplore.ieee.org/document/7721743.

[27] Meshtastic®. „RadioInterface.cpp“. (2022), Adresse: https://github.com/meshtastic/firmware/blob/568434d0fa4556cad0472285f252d464a2d07453/src/mesh/RadioInterface.cpp. Zeile 359-414, (letzter Zugriff: 07.11.2022 00:58 Uhr).

[28] S. Rubinstein-Salzedo, *Cryptography*, eng, 1st ed. 2018., Ser. Springer Undergraduate Mathematics Series. 2018, S. 71, ISBN: 3319948180.

[29] J. Daemen, *The Design of Rijndael The Advanced Encryption Standard (AES)*, eng, 2nd ed. 2020., Ser. Information Security and Cryptography. 2020, S. 1, ISBN: 3662607697.

[30] J. Loo, J. L. Mauri und J. Hamilton, *Mobile ad hoc networks : current status and future trends*, eng, 1st edition. 2012, S. 5, ISBN: 978-1-4398-5650-5.

[31] Meshtastic®. „About Meshtastic“. (2022), Adresse: https://meshtastic.org/docs/about. (letzter Zugriff: 24.08.2022 09:13 Uhr).

[32] Meshtastic®. „Device firmware“. (2022), Adresse: https://meshtastic.org/docs/software/device/. (letzter Zugriff: 24.08.2022 09:15 Uhr).

[33] LILYGO. „T-Echo SotfRF“. (2023), Adresse: https://www.lilygo.cc/products/t-echo-nrf52840. (letzter Zugriff: 14.02.2023 15:56 Uhr).

[34] Meshtastic®. „Nano G1 device“. (2022), Adresse: https://meshtastic.org/docs/hardware/supported/nano-g1. (letzter Zugriff: 04.11.2022 14:49 Uhr).

[35] Meshtastic®. „Meshtastic Repositories“. (2022), Adresse: https://github.com/orgs/meshtastic/repositories. (letzter Zugriff: 06.11.2022 22:10 Uhr).

[36] Meshtastic®. „goTenna Mesh“. (2023), Adresse: https://gotennamesh.com/products/mesh?utm_source=internal-link&utm_medium=menu&utm_campaign=gotenna.com. (letzter Zugriff: 14.02.2023 20:40 Uhr).

[37] Meshtastic®. „Getting Started“. (2023), Adresse: https://meshtastic.org/docs/getting-started. (letzter Zugriff: 14.02.2023 15:26 Uhr).

[38] Meshtastic®. „RAK WisBlock Devices“. (2023), Adresse: https://meshtastic.org/docs/hardware/devices/rak/. (letzter Zugriff: 14.02.2023 15:31 Uhr).

[39] Meshtastic®. „rak4631_5005“. (2022), Adresse: https://meshtastic.org/img/hardware/rak4631_5005.png. (letzter Zugriff: 03.11.2022 10:53 Uhr).

[40] Meshtastic®. „LILYGO® TTGO T-Echo devices“. (2023), Adresse: https://meshtastic.org/docs/hardware/devices/techo/. (letzter Zugriff: 14.02.2023 15:51 Uhr).

[41] LILYGO. „LILYGO T-Echo“. (2022), Adresse: https://ae01.alicdn.com/kf/H0d80240739114a3790e87935b555d3592.jpg. (letzter Zugriff: 03.11.2022 10:54 Uhr).

[42] Meshtastic®. „LILYGO® TTGO T-Beam Devices“. (2023), Adresse: https://meshtastic.org/docs/hardware/devices/tbeam/. (letzter Zugriff: 14.02.2023 15:57 Uhr).

[43] LILYGO. „T-Beam“. (2023), Adresse: https://www.lilygo.cc/products/t-beam-v1-1-esp32-lora-module?variant=42204035023029. (letzter Zugriff: 14.02.2023 15:59 Uhr).

[44] aliexpress. „Kaufoptionen T-Beam auf Aliexpress“. (), Adresse: https://de.aliexpress.com/item/4001178678568.html?spm=a2g0o.detail.1000023.3.5dd621b9vJFlsI&gatewayAdapt=glo2deu. (letzter Zugriff: 03.11.2022 12:03 Uhr).

[45] P. U. TonyG. „T-Beam case for Meshtastic (v5)“. (2022), Adresse: https://www.printables.com/de/model/127253-t-beam-case-for-meshtastic. (letzter Zugriff: 03.11.2022 15:39 Uhr).

[46] LILYGO. „LILYGO T-Beam V1.1“. (2022), Adresse: https://ae01.alicdn.com/kf/H91b40bc504ae4b28a4e91f8c573e845dh.jpg. (letzter Zugriff: 03.11.2022 16:42 Uhr).

[47] Meshtastic®. „LILYGO T-Beam V1.1“. (2022), Adresse: https://meshtastic.org/assets/files/t-beam-meshtastic-41aa747cc67809ff3013e69c2558833f.png. (letzter Zugriff: 03.11.2022 16:43 Uhr).

[48] Meshtastic®. „LILYGO® TTGO Lora Devices“. (2023), Adresse: https://meshtastic.org/docs/hardware/devices/lora/. (letzter Zugriff: 14.02.2023 16:09 Uhr).

[49] LILYGO. „LoRa32 V2.1_1.6“. (2023), Adresse: https://www.lilygo.cc/products/lora3?variant=42476923682997. (letzter Zugriff: 14.02.2023 16:07 Uhr).

[50] LILYGO. „LILYGO T-Lora32“. (2022), Adresse: http://www.lilygo.cn/Private/ProductImg/Other//20190819154150443%E2%88%AEH248-1.jpg. (letzter Zugriff: 03.11.2022 17:28 Uhr).

[51] N. Hao. „Meshtastic Mesh Device Nano Edition“. (2022), Adresse: https://uniteng.com/wiki/lib/exe/fetch.php?w=400&tok=7c6449&media=meshtastic:meshtastic_mesh_device_nano_edition_overview.jpg. (letzter Zugriff: 04.11.2022 14:40 Uhr).

[52] Meshtastic®. „HELTEC® Lora 32“. (2023), Adresse: https://meshtastic.org/docs/hardware/devices/heltec/. (letzter Zugriff: 14.02.2023 16:12 Uhr).

[53] Heltec. „WiFi LoRa 32 (V2.1) Phaseout“. (2023), Adresse: https://heltec.org/project/wifi-lora-32/. (letzter Zugriff: 14.02.2023 16:13 Uhr).

[54] Meshtastic®. „Heltec® device“. (2022), Adresse: https://meshtastic.org/docs/hardware/supported/heltec. (letzter Zugriff: 28.10.2022 18:27 Uhr).

[55] Heltec. „WiFi LoRa 32 (V2.1) Phaseout“. (2022), Adresse: https://heltec.org/wp-content/uploads/2020/05/IMG_1297_800X800-1.jpg. (letzter Zugriff: 04.11.2022 14:50 Uhr).

[56] Meshtastic®. „Meshtastic Firmware Releases“. (2022), Adresse: https://github.com/meshtastic/firmware/releases. (letzter Zugriff: 06.11.2022 15:12 Uhr).

[57] Meshtastic®. „Meshtastic Firmware Releases“. (2022), Adresse: https://github.com/meshtastic/python/releases. (letzter Zugriff: 06.11.2022 15:28 Uhr).

[58] Meshtastic®. „Meshtastic firmware“. (2022), Adresse: https://github.com/meshtastic/firmware/tree/master. (letzter Zugriff: 06.11.2022 22:53 Uhr).

[59] G. LLC. „Protocol Buffers“. (2022), Adresse: https://developers.google.com/protocol-buffers/. (letzter Zugriff: 06.11.2022 21:34 Uhr).

[60] Meshtastic®. „Meshtastic protobufs“. (2022), Adresse: https://github.com/meshtastic/protobufs. (letzter Zugriff: 06.11.2022 22:17 Uhr).

[61] Meshtastic®. „Meshtastic web“. (2022), Adresse: https://github.com/meshtastic/web. (letzter Zugriff: 06.11.2022 23:28 Uhr).

[62] G. LLC. „Meshtastic“. (2022), Adresse: https://play.google.com/store/apps/details?id=com.geeksville.mesh&gl=us. (letzter Zugriff: 06.11.2022 23:35 Uhr).

[63] Meshtastic®. „Meshtastic android“. (2022), Adresse: https://github.com/meshtastic/Meshtastic-Android. (letzter Zugriff: 06.11.2022 23:29 Uhr).

[64] Meshtastic®. „Meshtastic Apple Clients“. (2023), Adresse: https://github.com/meshtastic/Meshtastic-Apple. (letzter Zugriff: 14.02.2023 20:25 Uhr).

[65] Meshtastic®. „Meshtastic python“. (2022), Adresse: https://github.com/meshtastic/python. (letzter Zugriff: 06.11.2022 23:31 Uhr).

[66] Meshtastic®. „Meshtastic python“. (2022), Adresse: https://github.com/meshtastic/Meshtastic-gui-installer. (letzter Zugriff: 06.11.2022 23:32 Uhr).

[67] A. Augustin, J. Yi, T. Clausen und W. M. Townsley, „A Study of LoRa: Long Range & Low Power Networks for the Internet of Things“, *Sensors*, Jg. 16, Nr. 9, S. 4–8, 2016, ISSN: 1424-8220. DOI: 10.3390/s16091466. Adresse: https://www.mdpi.com/1424-8220/16/9/1466.

[68] Meshtastic®. „LoRa Configuration“. (2023), Adresse: https://meshtastic.org/docs/settings/config/lora. (letzter Zugriff: 15.02.2023 13:46 Uhr).

[69] L. NETVOX TECHNOLOGY CO. „R312-Wireless Door Bell Button“. (2023), Adresse: http://www.netvox.com.tw/product.asp?pro=R312. (letzter Zugriff: 15.02.2023 13:56 Uhr).

[70] Meshtastic®. „StreamAPI.cpp“. (2023), Adresse: https://github.com/meshtastic/firmware/blob/v2.0.5.65e8209/src/mesh/StreamAPI.cpp. (letzter Zugriff: 15.02.2023 14:16 Uhr).

[71] BSD, „magic - file command's magic pattern file“, *Linux manual page*, 2022. Adresse: https://man7.org/linux/man-pages/man4/magic.4.html, (letzter Zugriff: 15.02.2023 14:23 Uhr).

[72] Meshtastic®. „MeshRadio.h“. (2022), Adresse: https://github.com/meshtastic/firmware/blob/aac9b5db30a7266509e5216170ae56f3d2d43f23/src/mesh/MeshRadio.h. (letzter Zugriff: 15.02.2023 14:36 Uhr).

[73] B. für Flugsicherung. „Luftverkehrs-Ordnung (LuftVO) § 20 Erlaubnisbedürftige Nutzung des Luftraums“. (2023), Adresse: https://www.gesetze-im-internet.de/luftvo_2015/__20.html. (letzter Zugriff: 15.02.2023 15:39 Uhr).

[74] S. Corporation. „DATASHEET SX1276/77/78/79“. (2020), Adresse: https://semtech.file.force.com/sfc/dist/version/download/?oid=00DE0000000JelG&ids=0682R000006TQEdQAO&d=%2Fa%2F2R0000001Rc1%2FQnUuV9TviODKUgt_rpBlPz.EZA_PNK7Rpi8HA5..Sbo&operationContext=DELIVERY&asPdf=true&viewId=05H3n000002qNNeEAM&dpt=. (letzter Zugriff: 15.02.2023 16:33 Uhr).

[75] I. LoRa Alliance. „LoRaWAN 1.0.3 Regional Parameters“. (2018), Adresse: https://lora-alliance.org/wp-content/uploads/2020/11/lorawan_regional_parameters_v1.0.3reva_0.pdf. (letzter Zugriff: 15.02.2023 16:58 Uhr).

[76] Meshtastic®. „Meshtastic mesh.proto“. (2023), Adresse: https://github.com/meshtastic/protobufs/blob/master/meshtastic/mesh.proto. (letzter Zugriff: 02.02.2023 10:55 Uhr).

[77] Meshtastic®. „View your network“. (2023), Adresse: https://meshtastic.org/docs/software/android/usage#view-your-network. (letzter Zugriff: 15.02.2023 17:09 Uhr).

[78] J. Hightower und G. Borriello, „Location Sensing Techniques“, *University of Washington, Computer Science and Engineering*, Jg. 01-07, 2001. Adresse: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&%20doi=3a7cb58dec6c39d31db6c36aec6091e3149baaf6, (letzter Zugriff: 16.02.2023 13:21 Uhr).

[79] Z. Yang, Y. Liu und X.-Y. Li, „Beyond trilateration: On the localizability of wireless ad-hoc networks“, in *IEEE INFOCOM 2009*, 2009, S. 2392–2400. DOI: 10.1109/INFCOM.2009.5062166. Adresse: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5062166, (letzter Zugriff: 16.02.2023 13:17 Uhr).

[80] F. Subhan u. a., „Experimental analysis of received signals strength in Bluetooth Low Energy (BLE) and its effect on distance and position estimation“, *Transactions on Emerging Telecommunications Technologies*, Jg. 33, Nr. 2, e3793, 2022, e3793 ETT-19-0391.R1. DOI: https://doi.org/10.1002/ett.3793. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/ett.3793. Adresse: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.3793, (letzter Zugriff: 16.02.2023 13:52 Uhr).

[81] D. Eridani, A. F. Rochim und F. N. Cesara, „Comparative Performance Study of ESP-NOW, Wi-Fi, Bluetooth Protocols based on Range, Transmission Speed, Latency, Energy Usage and Barrier Resistance“, in *2021 International Seminar on Application for Technology of Information and Communication (iSemantic)*, 2021, S. 322–328. DOI: 10.1109/iSemantic52711.2021.9573246. Adresse: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9573246, (letzter Zugriff: 16.02.2023 14:49 Uhr).

[82] Kismet. „Kismet: Wi-Fi, Bluetooth, RF, and more“. (2023), Adresse: https://www.kismetwireless.net/. (letzter Zugriff: 16.02.2023 14:55 Uhr).

[83] Meshtastic®. „Connecting“. (2023), Adresse: https://meshtastic.org/docs/software/android/usage#connecting. (letzter Zugriff: 16.02.2023 13:45 Uhr).

[84] Meshtastic®. „Web Client Overview“. (2023), Adresse: https://meshtastic.org/docs/software/web-client. (letzter Zugriff: 16.02.2023 14:06 Uhr).

[85] Meshtastic®. „Network Configuration“. (2023), Adresse: https://meshtastic.org/docs/settings/config/network. (letzter Zugriff: 16.02.2023 14:10 Uhr).

[86] Espressif. „esp_wifi.h“. (2023), Adresse: https://github.com/espressif/esp-idf/blob/master/components/esp_wifi/include/esp_wifi.h. (letzter Zugriff: 16.02.2023 14:17 Uhr).

[87] Kismet. „Kismet Github Repository“. (2023), Adresse: https://github.com/kismetwireless/kismet. (letzter Zugriff: 16.02.2023 14:55 Uhr).

[88] B. Hofmann-Wellenhof, *GNSS – Global Navigation Satellite Systems GPS, GLONASS, Galileo, and more*, eng, 1st ed. 2008. 2008, S. 55, ISBN: 3211730176.

[89] B. Hofmann-Wellenhof, *GNSS – Global Navigation Satellite Systems GPS, GLONASS, Galileo, and more*, eng, 1st ed. 2008. 2008, S. 91, ISBN: 3211730176.

[90] G. I. Limited. „Meshtastic“. (2023), Adresse: https://play.google.com/store/apps/details?id=com.geeksville.mesh. (letzter Zugriff: 16.02.2023 15:30 Uhr).

[91] Meshtastic®. „View the map“. (2023), Adresse: https://meshtastic.org/docs/software/android/usage#view-the-map. (letzter Zugriff: 16.02.2023 15:33 Uhr).

[92] G. I. Limited. „Standortgenauigkeit ermitteln und verbessern“. (2023), Adresse: https://support.google.com/maps/answer/2839911?hl=de&co=GENIE.Platform%3DAndroid. (letzter Zugriff: 16.02.2023 15:37 Uhr).

[93] Meshtastic®. „PacketDao.kt“. (2023), Adresse: https://github.com/meshtastic/Meshtastic-Android/blob/c4b20912bd2c45a4e54159afe17e8a07630c2b47/app/src/main/java/com/geeksville/mesh/database/dao/PacketDao.kt. (letzter Zugriff: 16.02.2023 19:30 Uhr).

[94] Meshtastic®. „Remote Node Administration“. (2023), Adresse: https://meshtastic.org/docs/configuration/remote-admin. (letzter Zugriff: 16.02.2023 19:37 Uhr).

[95] Meshtastic®. „Setup a channel“. (2023), Adresse: https://meshtastic.org/docs/software/android/usage#setup-a-channel. (letzter Zugriff: 16.02.2023 19:39 Uhr).

[96] Meshtastic®. „Configuration options“. (2023), Adresse: https://meshtastic.org/docs/software/android/usage#configuration-options. (letzter Zugriff: 16.02.2023 19:42 Uhr).

[97] Meshtastic®. „Mesh Broadcast Algorithm“. (2023), Adresse: https://meshtastic.org/docs/overview/mesh-algo. (letzter Zugriff: 16.02.2023 16:23 Uhr).

[98] Meshtastic®. „screen.cpp“. (2023), Adresse: https://github.com/meshtastic/firmware/blob/master/src/graphics/Screen.cpp. (letzter Zugriff: 16.02.2023 19:48 Uhr).

[99] Meshtastic®. „Meshtastic Python CLI Guide“. (2023), Adresse: https://meshtastic.org/docs/software/python/cli. (letzter Zugriff: 16.02.2023 19:52 Uhr).

[100] Meshtastic®. „Channel Configuration“. (2023), Adresse: https://meshtastic.org/docs/settings/config/channels. (letzter Zugriff: 16.02.2023 16:38 Uhr).

[101] Meshtastic®. „portnums.proto“. (2022), Adresse: https://github.com/meshtastic/protobufs/blob/afa4605699e9ba9e2d0f0407bbc32dcd133f76af/portnums.proto. (letzter Zugriff: 02.12.2022 16:28 Uhr).

[102] D. Johnson, Y. Hu und D. Maltz. „RFC 4728: The Dynamic Source Routing Protocol“. (2007), Adresse: https://www.rfc-editor.org/rfc/rfc4728. (letzter Zugriff: 02.12.2022 16:17 Uhr).

[103] S. L. CC. „Unishox2“. (2022), Adresse: https://github.com/siara-cc/Unishox2. (letzter Zugriff: 05.12.2022 15:48 Uhr).

[104] Meshtastic®. „Message Fragment“. (2022), Adresse: https://github.com/meshtastic/Meshtastic-Android/blob/79b3b1c0244c04244fb93fe1a35a46c3d29cefd9/app/src/main/res/layout/messages_fragment.xml. (letzter Zugriff: 05.12.2022 16:13 Uhr).

[105] Meshtastic®. „telemetry.proto“. (2023), Adresse: https://github.com/meshtastic/protobufs/blob/master/meshtastic/telemetry.proto. (letzter Zugriff: 16.02.2023 20:08 Uhr).

[106] Meshtastic®. „admin.proto“. (2023), Adresse: https://github.com/meshtastic/protobufs/blob/master/meshtastic/admin.proto. (letzter Zugriff: 16.02.2023 20:11 Uhr).

[107] M. Ryan. „crackle“. (2023), Adresse: https://github.com/mikeryan/crackle/. (letzter Zugriff: 16.02.2023 20:14 Uhr).

[108] M. Ryan. „Frequently Asked Questions“. (2023), Adresse: https://github.com/mikeryan/crackle/blob/master/FAQ.md. (letzter Zugriff: 16.02.2023 20:27 Uhr).

[109] Aircrack-ng. „Introduction aircrack-ng“. (2023), Adresse: https://www.aircrack-ng.org/. (letzter Zugriff: 16.02.2023 21:23 Uhr).

[110] Aircrack-ng. „Tutorial: How to Crack WPA/WPA2“. (2023), Adresse: https://www.aircrack-ng.org/doku.php?id=cracking_wpa. (letzter Zugriff: 16.02.2023 21:26 Uhr).

[111] Meshtastic®. „Channel“. (2022), Adresse: https://github.com/meshtastic/firmware/blob/5ed2a4e8bb2052116295516c4fa893d7203d27c8/src/mesh/Channels.cpp. (letzter Zugriff: 12.12.2022 17:20 Uhr).

[112] Meshtastic®. „PSK“. (2023), Adresse: https://meshtastic.org/docs/settings/config/channels#psk-1. (letzter Zugriff: 16.02.2023 22:14 Uhr).

[113] Meshtastic®. „Channel“. (2022), Adresse: https://github.com/meshtastic/firmware/blob/e85af2f7327fcb2d7029ad0ca0b460d9aa5e03ec/src/mesh/generated/channel.pb.h. (letzter Zugriff: 12.12.2022 16:53 Uhr).

[114] M. Rezl. „Meshtastic - An unofficial guide to the open source, off-grid, GPS mesh communicator.“ (2023), Adresse: https://meshtastic.letstalkthis.com/. (letzter Zugriff: 16.02.2023 22:14 Uhr).

[115] Meshtastic®. „__main__.py“. (2023), Adresse: https://github.com/meshtastic/python/blob/93441a473f48b6209e3804c877942c5b820be3f4/meshtastic/__main__.py. (letzter Zugriff: 17.02.2023 10:52 Uhr).

[116] G. I. Limited. „BLE Scanner“. (2023), Adresse: https://play.google.com/store/apps/details?id=com.macdom.ble.blescanner&gl=US&pli=1. (letzter Zugriff: 17.02.2023 10:56 Uhr).

[117] Meshtastic®. „Python Library“. (2022), Adresse: https://pypi.org/project/meshtastic/2.0.3/. (letzter Zugriff: 30.01.2023 13:44 Uhr).

[118] Meshtastic®. „meshtastic.__init__.py“. (2022), Adresse: https://github.com/meshtastic/python/blob/e6e3ad121dfbbe1797a0324f59bdf406460f6a24/meshtastic/__init__.py. (letzter Zugriff: 30.01.2023 19:42 Uhr).

[119] Meshtastic®. „__init__.py“. (2023), Adresse: https://github.com/meshtastic/python/blob/7133c859d66fd6beee6b5bc799f5aba758f48924/meshtastic/__init__.py. (letzter Zugriff: 17.02.2023 11:03 Uhr).

[120] D. S. Foundation. „Django Overview“. (2023), Adresse: https://www.djangoproject.com/start/overview/. (letzter Zugriff: 17.02.2023 11:07 Uhr).

[121] D. S. Foundation. „Django Documentation Models“. (2023), Adresse: https://docs.djangoproject.com/en/4.1/topics/db/models/. (letzter Zugriff: 17.02.2023 11:09 Uhr).

[122] Meshtastic®. „Supported Devices“. (2023), Adresse: https://meshtastic.org/docs/supported-hardware. (letzter Zugriff: 17.02.2023 11:11 Uhr).

[123] D. S. Foundation. „Django Databases“. (2023), Adresse: https://docs.djangoproject.com/en/4.1/ref/databases/. (letzter Zugriff: 31.01.2023 20:09 Uhr).

[124] T. S. Consortium. „Appropriate Uses For SQLite“. (2023), Adresse: https://www.sqlite.org/whentouse.html. (letzter Zugriff: 31.01.2023 20:26 Uhr).

[125] T. Christie. „Quickstart“. (2023), Adresse: https://www.django-rest-framework.org/tutorial/quickstart/. (letzter Zugriff: 01.02.2023 13:46 Uhr).

[126] P. S. Foundation. „JSON encoder and decoder“. (2023), Adresse: https://docs.python.org/3/library/json.html. (letzter Zugriff: 01.02.2023 13:55 Uhr).

[127] T. Christie. „Serializers“. (2023), Adresse: https://www.django-rest-framework.org/api-guide/serializers/. (letzter Zugriff: 03.02.2023 13:53 Uhr).

[128] T. Christie. „CreateAPIView“. (2023), Adresse: https://www.django-rest-framework.org/api-guide/generic-views/#createapiview. (letzter Zugriff: 03.02.2023 14:27 Uhr).

[129] D. S. Foundation. „How to create custom django-admin commands“. (2023), Adresse: https://docs.djangoproject.com/en/4.1/howto/custom-management-commands/. (letzter Zugriff: 17.02.2023 11:16 Uhr).

[130] Meshtastic®. „Router.cpp“. (2023), Adresse: https://github.com/meshtastic/firmware/blob/ade32b1827b7dc554716625e50c3ccfd8b87085b/src/mesh/Router.cpp. (letzter Zugriff: 17.02.2023 14:47 Uhr).

[131] R. M. GmbH. „Stadtverkehr Mittweida A, Baumwollspinnerei Wendeschleife - Markt - Bahnhof“. (2023), Adresse: https://www.regiobus.com/fileadmin/Mittweida/Fahrplaene/Stadtverkehr/A_h_Mittweida.pdf. (letzter Zugriff: 17.02.2023 15:02 Uhr).

[132] C.-B. C. Gmb. „Fahrplan C14“. (2023), Adresse: https://www.city-bahn.de/csdata/download/4/de/c14_2023_8206.pdf. (letzter Zugriff: 17.02.2023 15:11 Uhr).

[133] M. Regiobahn. „Fahrzeiten RB45“. (2023), Adresse: https://download.transdev.de/transdev/uploads/mdr/schedule/1042/fahrplan-rb-45-chemnitz-elsterwerda.pdf. (letzter Zugriff: 17.02.2023 15:13 Uhr).

[134] Meshtastic®. „Meshtastic Firmware 2.0.15.aafbde0 Alpha“. (2023), Adresse: https://github.com/meshtastic/firmware/releases/tag/v2.0.15.aafbde0. (letzter Zugriff: 17.02.2023 14:47 Uhr).

[135] Meshtastic®. „Radio Settings“. (2023), Adresse: https://meshtastic.org/docs/overview/radio-settings. (letzter Zugriff: 17.02.2023 15:48 Uhr).

[136] Meshtastic®. „Meshtastic Firmware 2.0.0.18ab874“. (2022), Adresse: https://github.com/meshtastic/firmware/releases/tag/v2.0.0.18ab874. (letzter Zugriff: 06.02.2023 13:59 Uhr).

[137] I. Amazon.com. „2er Pack USB Netzteil - Ladegerät - Steckdosenadapter - Stecker 5V-1A Universal – Kompatibel mit Handy,Kamera,Tablets, MP3 usw. (Weiß)“. (2023), Adresse: https://www.amazon.de/2er-Pack-USB-Netzteil-Steckdosenadapter-White/dp/B08ZM87MXZ/ref=sr_1_11?keywords=YSONIC&qid=1676644635&sr=8-11. (letzter Zugriff: 17.02.2023 15:38 Uhr).

[138] Meshtastic®. „Releases“. (2023), Adresse: https://github.com/meshtastic/firmware/releases. (letzter Zugriff: 19.02.2023 11:34 Uhr).

[139] F. S. S.L. „High performance Satellites“. (2023), Adresse: https://fossa.systems/satellites/. (letzter Zugriff: 09.02.2023 09:40 Uhr).

Eidesstattliche Erklärung

Hiermit versichere ich – Felix Fischer – an Eides statt, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe.

Sämtliche Stellen der Arbeit, die im Wortlaut oder dem Sinn nach Publikationen oder Vorträgen anderer Autoren entnommen sind, habe ich als solche kenntlich gemacht.

Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt oder anderweitig veröffentlicht.

Mittweida, 20. Februar 2023 | Felix Fischer, B.Sc.

Ort, Datum | Felix Fischer, B.Sc.