

## Bachelorarbeit

---

# Mobile Fingerprinting

---

OST - Ostschweizer Fachhochschule  
(Campus Rapperswil-Jona)

Departement Informatik

Herbstsemester 2020/2021

<b>Autoren</b>	Janik Schlatter	<b>Betreuer</b>	Prof. Beat Stettler
	Mike Schmid	<b>Experte</b>	Martin Willi
<b>Projektpartner</b>	INS	<b>Gegenleser</b>	Claudio Fuchs

---

## Abstract

Mobilgeräte senden für die Suche nach WLAN-Netzwerken Probe-Requests aus. In diesen Probe-Requests sind zusätzliche Informationen, wie beispielsweise die unterstützten Datenraten oder bekannte Netzwerke, enthalten. Seit Android und iOS 8 werden MAC-Adressen in Probe-Requests randomisiert.

Ziel der Arbeit ist, das Verhalten von verschiedenen Mobilgeräten mit modernen Betriebssystemversionen zu analysieren und auf Basis der Erkenntnisse ein Programm zu entwickeln, welches Mobilgeräte voneinander unterscheiden kann. Die Unterscheidung kann in Form eines Fingerprintings vorgenommen werden und allenfalls auch für eine Verfolgung von bekannten Geräten genutzt werden.

In der Arbeit wurden drei iOS-Geräte und neun Android-Geräte in insgesamt 108 Einzelmessungen untersucht. Mit den Ergebnissen wurde ein Prototyp entworfen, welcher Messungen aufgrund der zusätzlichen Felder in Probe-Requests filtern und die Gesamtzahl der Mobilgeräte im Empfangsbereich auswerten kann.

Ein Verfahren, mit dem man Mobilgeräte langfristig mit einem Fingerabdruck versehen kann, ist anhand der in den Messungen gewonnenen Erkenntnisse nicht umsetzbar. Es hat sich gezeigt, dass sich in den neueren Betriebssystemversionen die Probe-Requests nicht mehr wesentlich voneinander unterscheiden.

In künftigen Verfahren für die Erkennung, Unterscheidung und Verfolgung von Mobilgeräten wird deshalb auf weitere Informationsquellen wie die Bluetooth-Schnittstelle zurückgegriffen werden müssen.

## Bachelorarbeit "Mobile Fingerprinting"

Smartphones senden regelmässig WLAN-Probe-Requests, um Access Points zu finden. In der Vergangenheit konnte durch das Auslesen der MAC-Adresse aus diesen Requests ein Gerät über längere Zeit verfolgt werden.

Seit einigen Jahren wird diese Verfolgung durch die Verwendung von anonymisierten MAC-Adressen erschwert. Weitere Datenfelder in diesen Requests können für das Tracken aber weiterhin von Nutzen sein.

In einer früheren Arbeit wurde anhand bestehender Daten eines Industriepartners ein Top-Down-Ansatz mittels Maschine Learning erarbeitet, der aber nicht zu genügend genauen Resultaten führte.

### Aufgabe:

In dieser Arbeit soll mit einem Bottom-Up-Ansatz eruiert werden, wie sich die unterschiedlichen Geräte- und Betriebssystemtypen verhalten, das heisst ob und welche Anonymisierung der MAC-Adressen auf den gängigsten Mobilgeräten/ Betriebssystemen vorgenommen wird. Zudem müssen individuelle Attribute gefunden werden, welche auch Geräte des gleichen Typs voneinander unterscheiden lassen.

Basierend auf diesen Erkenntnissen soll ein Prototyp erstellt werden, der Geräte anhand der Verhaltensweise unterscheiden und einem Pseudonym zuordnen kann, damit die zwei folgenden Fragestellungen beantwortet werden:

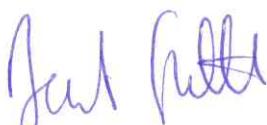
1. Wie viele Geräte befinden sich im Messbereich (Zählung)?
2. Befindet sich das gleiche Gerät später erneut im selben oder in einem anderen Messbereich? Dann soll es wiedererkannt werden (Tracking)

Der Prototyp soll bevorzugt in der Programmiersprache Python erstellt werden.

### Voraussetzung:

- Interesse an selbstständiger Erforschung eines komplexen Themas
- Lernbereitschaft, Freude an Experimentellem Arbeiten
- Flair für kreative Lösungsfindung

Unterschrift:



---

## Kurzübersicht

### Situation

Mobilgeräte verwenden für die Suche nach WLAN-Netzwerken sogenannte Probe-Requests. Diese Nachrichten werden an alle Geräte im Sendebereich des Mobilgeräts ausgesandt und enthalten oftmals weitere Informationen, wie beispielsweise bekannte Netzwerke oder unterstützte Datenraten. Bis vor einigen Jahren wurde in diesen Probe-Requests die einzigartige Geräteadresse mitgesendet. Nun wird diese Adresse jeweils zufalls generiert.

Ein Verfahren, welches erlaubt, Mobilgeräte durch diese Probe-Requests zu erkennen, voneinander zu unterscheiden und über längere Zeit zu verfolgen, kann in vielen Anwendungen Gebrauch finden. Ein Beispiel wäre die automatische Fahrgastzählung im mobilen Verkehr.

### Vorgehen

In dieser Arbeit wurden Messungen durchgeführt, um das Verhalten von Mobilgeräten zu untersuchen. Anhand der Messergebnisse wurde ein Prototyp in Python entwickelt, welcher in der Lage ist, Mobilgeräte voneinander zu unterscheiden und die Anzahl Mobilgeräte im Empfangsbereich der Messantenne zu ermitteln.

### Ergebnisse

Die Auswertung der Messergebnisse hat ergeben, dass iOS-Geräte im Verhalten beinahe identisch sind. Es ist schwierig bis unmöglich, diese Geräte anhand ausgesendeter Probe-Requests zu unterscheiden. Android-Geräte haben im Verhalten noch genügend Unterschiede, dass eine Unterteilung möglich ist. Eine längere Verfolgung lässt sich aber auch nicht umsetzen.

### Ausblick

Gerätehersteller verbessern mit jeder neuen Betriebssystemversion das Verhalten im Bezug auf die Erkennbarkeit und Unterscheidbarkeit. Zum einen weil gewisse Verfahren aufgrund neuer Normen und Privatsphäregesetze vorausgesetzt werden, aber auch weil die Hersteller die eigenen Technologien für die Lokalisierung ihrer Geräte dadurch besser verkaufen können.

Es ist möglich, dass mit zusätzlichen Messungen auf weiteren Mobilgeräten weitere Verhaltensunterschiede erkannt werden können, die für eine Verbesserung der Verfahren verwendet werden können. Außerdem kann mit hinzuziehen weiterer Signalquellen - beispielsweise Bluetooth - die Unterscheidung von Mobilgeräten weiter verfeinert werden.

---

## Danksagungen

Wir möchten den folgenden Personen für die Unterstützung unserer Bachelorarbeit danken:

- Beat Stettler, der uns bei Fragen und Problemen stets hilfreich zur Seite stand und uns im Rahmen seiner Möglichkeiten mit der benötigten Hardware versorgte.
- Marcel Kluser für das zur Verfügung stellen der Antennenmesskammer des ICOM (Institut für Kommunikationssysteme).
- Den Projektingenieur/-innen des ICOM, Hans-Dieter Lang, Selina Malacarne, Michel Nyffenegger und Nicola Ramangnano, die uns bei Fragen zum Gebrauch der Antennenmesskammer unterstützt haben und die Kammer für unsere Messungen vorbereitet haben.
- Christian Spielmann für die Aqise und Organisation von für die Messungen benötigten Hardware.
- Raphael Das Gupta für das zur Verfügung stellen von Mobilgeräten aus dem Institut für Software IFS
- Fanny Urech für das Gegenlesen und die Fehlerkorrekturen
- Raphael Jud für das zur Verfügung stellen seines iPhone X für die Messungen.
- Pascale Meier für das zur Verfügung stellen ihres iPhone 8 für die Messungen.
- Jenny Bösch für das zur Verfügung stellen ihres Samsung Galaxy S8 für die Messungen.

---

# Inhaltsverzeichnis

---

<b>I Analyse</b>	<b>1</b>
1 Einleitung . . . . .	2
1.1 Problemstellung . . . . .	2
1.2 Herausforderungen . . . . .	2
1.3 Vorarbeit . . . . .	3
2 Technischer Hintergrund . . . . .	4
2.1 MAC-Adressen . . . . .	4
2.2 IEEE 802.11 WLAN Standard . . . . .	6
2.3 802.11 Frames . . . . .	8
2.4 Probe-Requests . . . . .	12
2.5 Probe-Request-Burst . . . . .	14
2.6 Probe-Request Header-Frame . . . . .	15
3 Verwandte Arbeiten . . . . .	19
3.1 How Talkative is Your Mobile Device? . . . . .	19
3.2 Defeating MAC Address Randomization Through Timing Attacks . . . . .	23
3.3 Why MAC Address Randomization is not Enough . . . . .	25
3.4 Noncooperative 802.11 MAC Layer Fingerprinting . . . . .	27
4 Allgemeine Mobilgeräte-Analyse . . . . .	30
4.1 Smartphones . . . . .	30
4.2 Auswahl der Mobilgeräte . . . . .	33
5 iOS-Analyse . . . . .	35
5.1 Versionsgeschichte . . . . .	35
5.2 iOS 8 . . . . .	36
5.3 iOS 9 . . . . .	37
5.4 iOS 10 - 13 . . . . .	38
5.5 iOS 14 . . . . .	38
6 Android-Analyse . . . . .	39
6.1 Versionsgeschichte . . . . .	39
6.2 Android 8 - Oreo . . . . .	40

6.3	Android 9 - Pie . . . . .	40
6.4	Android 10 . . . . .	40
6.5	Android 11 . . . . .	40
<b>II</b>	<b>Versuche</b>	<b>41</b>
7	Einleitung . . . . .	42
7.1	Versuchsaufbau . . . . .	42
7.2	Verwendete Tools . . . . .	44
7.3	Durchzuführende Messungen . . . . .	45
7.4	Erwartete Messergebnisse . . . . .	46
7.5	Messplan . . . . .	46
7.6	Beschaffung der Geräte . . . . .	48
8	IOS-Messungen . . . . .	49
8.1	Versuchsdurchführung . . . . .	49
8.2	Ergebnisse . . . . .	50
9	Android-Messungen . . . . .	57
9.1	Versuchsdurchführung . . . . .	57
9.2	Ergebnisse . . . . .	58
10	Schlussfolgerungen . . . . .	68
10.1	Generelle Ansätze für ein Fingerprinting anhand ausgesendeter Probe-Requests . . . . .	69
10.2	Spezifische Ansätze für die Geräteunterscheidung . . . . .	72
<b>III</b>	<b>Prototyp</b>	<b>77</b>
11	Prototyp-Einführung . . . . .	78
11.1	Preprocessing . . . . .	78
11.2	Vorfilterung . . . . .	79
11.3	Filterung . . . . .	79
11.4	Verwerfen des Ansatzes zur Zeitbasierten Auswertung . . . . .	80
11.5	Konzept: Säuberung des Datensets . . . . .	81
12	Software-Entwicklung . . . . .	82
12.1	Beschreibung der Klassen . . . . .	82
13	Prototyp-Verifikation . . . . .	86
13.1	Messungen für die Verifikation . . . . .	86
<b>IV</b>	<b>Abschluss</b>	<b>90</b>
14	Weiterführende Arbeiten . . . . .	91
14.1	Entwicklung einer Zähleinrichtung . . . . .	91
14.2	Erweiterung der Filterung - Algorithmen . . . . .	92
14.3	Erweiterung der Filterung - Messungen . . . . .	93
14.4	Erweiterung der Aufzeichnungsmethoden . . . . .	93

## Inhaltsverzeichnis

---

15	Abschliessendes Urteil . . . . .	94
<b>V</b>	<b>Projektmanagement</b>	<b>95</b>
16	Changelog . . . . .	96
17	Einführung . . . . .	97
18	Projektübersicht . . . . .	98
19	Projektorganisation . . . . .	100
20	Management . . . . .	101
21	Risikomanagement . . . . .	105
22	Arbeitspakete . . . . .	109
23	Infrastruktur . . . . .	113
24	Qualitätsmassnahmen . . . . .	115
<b>Anhang</b>		<b>i</b>
Anhang A: Abkürzungsverzeichnis		i
Anhang B: Quellenverzeichnis und Bibliografie		iv
Anhang C: Abbildungsverzeichnis		x
Anhang D: Tabellenverzeichnis		xii
Anhang E: Experimentelle Daten		xiv
Anhang F: Verhaltenskatalog der Mobilgeräte		xxvi
Anhang G: Risikotabelle		xxxiv
Anhang H: Protokolle der Meetings		xxxviii
Anhang I: Zeitauswertung		li
Anhang J: Eigenschaftserklärung		liii
Anhang K: Urheber- und Nutzungsrechte		liv
Persönliche Berichte		lvi
Mobile Fingerprinting		vii

# **Teil I**

# **Analyse**

# 1 Einleitung

## 1.1 Problemstellung

Die Erkennung und Verfolgung von Mobilgeräten anhand ausgesendeter Probe-Requests ist in der Industrie eine bekannte Praxis. Früher konnte ein Mobilgerät anhand der in Probe-Requests ausgesendeten MAC-Adresse mit einem Fingerabdruck versehen werden und über grössere Distanzen und längere Zeit verfolgt werden. Seit einigen Betriebssystemversionen wird die MAC-Adresse aber in Probe-Requests zufallsgeneriert und kann nicht mehr für ein Fingerprinting verwendet werden. Die Frage, welche in dieser Arbeit beantwortet werden soll, ist: Senden Mobilgeräte in Probe-Requests genügend Informationen aus, um damit einen Fingerabdruck zu generieren und lassen sich Mobilgeräte anhand dieser Informationen unterscheiden und verfolgen. Spezifisch soll ein Verfahren entwickelt werden, welches erlaubt, Mobilgeräte im Empfangsbereich eines WLAN-Access-Point zu zählen. Um die Frage zu beantworten, sollen Messungen mit Mobilgeräten durchgeführt werden, um Gerätespezifisches Verhalten zu erkennen, welches für ein Fingerprinting verwendet werden kann.

## 1.2 Herausforderungen

Die grösste Herausforderung ist die Beschaffung der Mobilgeräte, die für die Messungen benötigt werden. Um statistisch relevante Ergebnisse erzeugen zu können, sollte eine Vielzahl von Messungen mit unterschiedlichen Geräteherstellern, -typen und -Betriebssystemversionen durchgeführt werden können. Weiterhin sollten die Messungen in einer Umgebung durchgeführt werden, in der möglichst keine Störsignale aufgezeichnet werden. Zusätzlich sollten die Messungen in einem möglichst realitätsnahen Umfeld durchgeführt werden, was in einer Störungsfreien Umgebung nicht möglich ist.

Eine weitere Herausforderung ist das Zeitmanagement. Werden mehr Messungen durchgeführt, bleibt weniger Zeit, um an einem Prototyp zu arbeiten. Wenn ein Verfahren in der Analyse vielversprechend ist, in der Umsetzung aber nicht funktioniert kann dadurch zusätzlich Zeit verloren gehen.

### 1.3 Vorarbeit

In einer Vorarbeit im Herbstsemester 2019/2020 wurde mit einem Machine-Learning-Verfahren gearbeitet, welches mit Daten aus der Industrie trainiert wurde. Insgesamt wurden 175 Millionen Probes in der Datensammlung analysiert und ein Prototyp entwickelt der gemäss der Dokumentation die Anzahl Passagiere in einem Bus mit 94% Genauigkeit voraussagen kann.

Die grössten Herausforderungen in der Vorarbeit bestanden darin, dass der Prototyp neben Mobilgeräten auch weitere Geräte im WLAN erkennt und diese nicht Filtern kann und dass Geräte vom selben Hersteller schwierig zu unterscheiden sind.

Vor allem das zweite Problem konnte in den Messungen dieser Arbeit beobachtet und bestätigt werden.

Da in der Vorarbeit mit Datensätzen gearbeitet wurde, welche keine Angaben haben, welcher Probe-Request von welchem Gerät ausgesendet wurde, können diese Daten nicht wiederverwendet werden.

## 2 Technischer Hintergrund

In diesem Abschnitt wird das für die späteren Abschnitte benötigte Hintergrundwissen beschrieben. Zuerst wird auf MAC-Adressen (Media Access Control) im Allgemeinen eingegangen, dann wird der WLAN Standard genauer beschrieben und danach wird auf Details der Probe-Requests eingegangen.

### 2.1 MAC-Adressen

Die Media-Access-Control Adresse ist eine ID zur eindeutigen Identifikation von Netzwerkcontrollern (NIC, Network Interface Controller), um in einem Netzwerk mit anderen Geräten kommunizieren zu können. Die MAC wird in den meisten Netzwerktechnologien- z.B. Wi-Fi, Bluetooth oder Ethernet- verwendet, um sich mit einem Netzwerk zu verbinden und ist einzigartig für jeden NIC. MAC-Adressen werden von der IEEE (Institute of Electrical and Electronics Engineers) für jeden Hersteller zugewiesen.

Die MAC-Adresse setzt sich aus sechs Bytes (48 Bit) zusammen und wird vom Gerätehersteller in der Manufaktur direkt hartcodiert. In der Abbildung 1 ist der Aufbau der MAC ersichtlich. Üblicherweise sind die ersten drei Bytes (24 Bit) die Beschreibung des Herstellers, auch Herstellerkennung oder OUI (Organizational Unique Identifier) genannt.

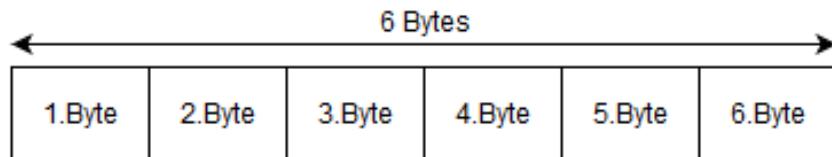


Abbildung 1: MAC-Adresse Aufbau

Die restlichen drei Bytes sind für die Identifikation des NIC reserviert. Es gibt zwei Bit, beide im ersten Byte der Adresse, die für die MAC-Adresse eine besondere Relevanz haben. Das Gruppenbit, welches sich an der letzten Stelle im Byte befindet, wird für die Unterscheidung von Unicast- und Multicast-Adressen verwendet. Das Lokale oder U/L-Bit an der zweitletzten Stelle unterscheidet zwischen global einzigartigen (und von der IEEE zugewiesenen) Adressen und lokal verwalteten Adressen.

## 2. TECHNISCHER HINTERGRUND

---

In der Abbildung 2 ist die Struktur der MAC-Adresse dargestellt.

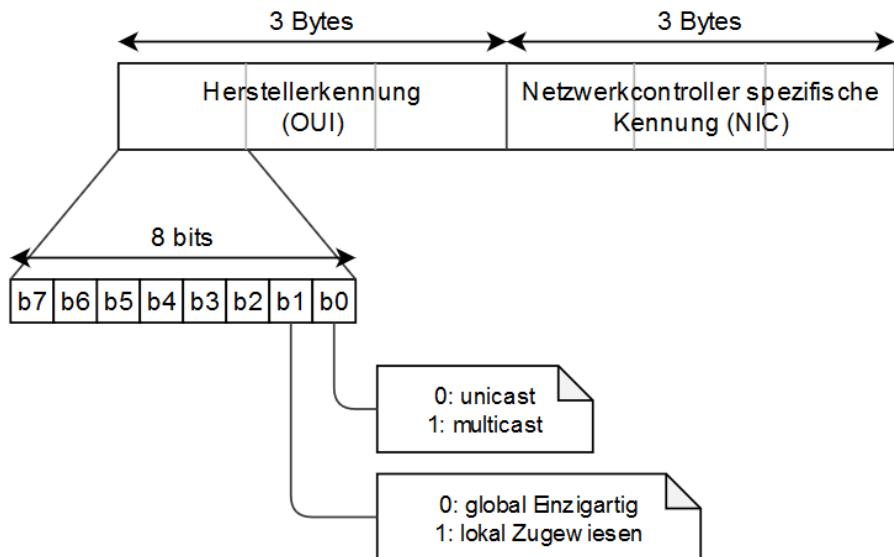


Abbildung 2: Detaillierte MAC-Adresse

Falls das lokale Bit gesetzt ist, kann dies anhand der MAC-Adresse sehr schnell erkannt werden, da das zweite Zeichen im ersten Byte eine von vier Zahlen gemäss der Tabelle 1 ist.

x2-xx-xx-xx-xx-xx-xx
x6-xx-xx-xx-xx-xx-xx
xA-xx-xx-xx-xx-xx-xx
xE-xx-xx-xx-xx-xx-xx

Tabelle 1: Lokale MAC-Adressen

## 2.2 IEEE 802.11 WLAN Standard

Der IEEE 802.11 Standard ist eine Ansammlung von Protokollen, die das Zusammenspiel von Komponenten in kabellosen Netzwerken (WLAN, Wireless Local Area Network) regeln, und baut auf dem 802 Standard Für lokale Computernetzwerke (LAN) auf.

Der WLAN Standard definiert unter anderem auf welche Art und Weise Computer, Mobiltelefone und andere Geräte miteinander interagieren welche Frequenzbänder sie dabei verwenden, welche Formate die übertragenen Datenpakete haben müssen und welche Sicherheitsvorkehrungen getroffen werden sollen.

Nachfolgend wird auf die wichtigsten Spezifikationen und Konzepte eingegangen, die im Rahmen dieser Arbeit relevant sind.

### Frequenzbänder

Die meistverwendeten Frequenzbänder im WLAN sind das 2.4-GHz-Band und das 5-GHz-Band. Diese Frequenzbänder sind jeweils in gleich grosse Kanäle unterteilt. Um Störungen durch überlappende Frequenzbänder zu verhindern, werden im 2.4-GHz-Bereich in Europa üblicherweise die Kanäle 1, 6 und 11 verwendet. In der Abbildung 3 ist die Unterteilung der Kanäle ersichtlich. Im 5-GHz-Bereich wird die Kanalunterteilung in Europa gemäss der Tabelle 2 vorgenommen.

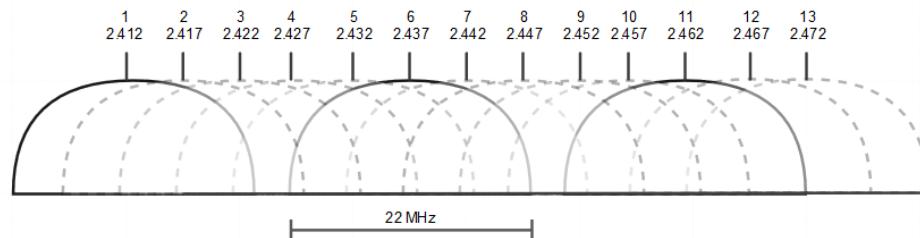


Abbildung 3: Kanalzuteilung im 2.4-GHz-Frequenzband

## 2. TECHNISCHER HINTERGRUND

Kanal	Frequenz	Kanal	Frequenz
36	5,180	108	5,540
40	5,200	112	5,560
44	5,220	116	5,580
48	5,240	120	5,600
52	5,260	124	5,620
56	5,280	128	5,640
60	5,300	132	5,660
64	5,320	136	5,680
100	5,500	140	5,700
104	5,520		

Tabelle 2: Kanalzuteilung 5-GHz-Frequenzband. Jeder Kanal ist 20 MHz breit.

Geräte, die über das WLAN kommunizieren, werden auf die Kanäle aufgeteilt, so dass alle Kanäle gleichmässig ausgelastet sind, um eine möglichst grosse Datenrate für jedes Endgerät zu gewährleisten. Somit ist es nicht möglich, nur einen Kanal zu überwachen, um sämtliche Endgeräte in einem WLAN-Netzwerk zu identifizieren.

## 2. TECHNISCHER HINTERGRUND

### 2.3 802.11 Frames

Im IEEE 802.11 Standard ist definiert, wie die Datenpakete, sogenannte Frames, aufgebaut sein müssen. Diese Frames befinden im OSI-Modell (Siehe Abbildung 4) auf dem zweiten Layer, dem Data Link Layer (Sicherungsschicht).

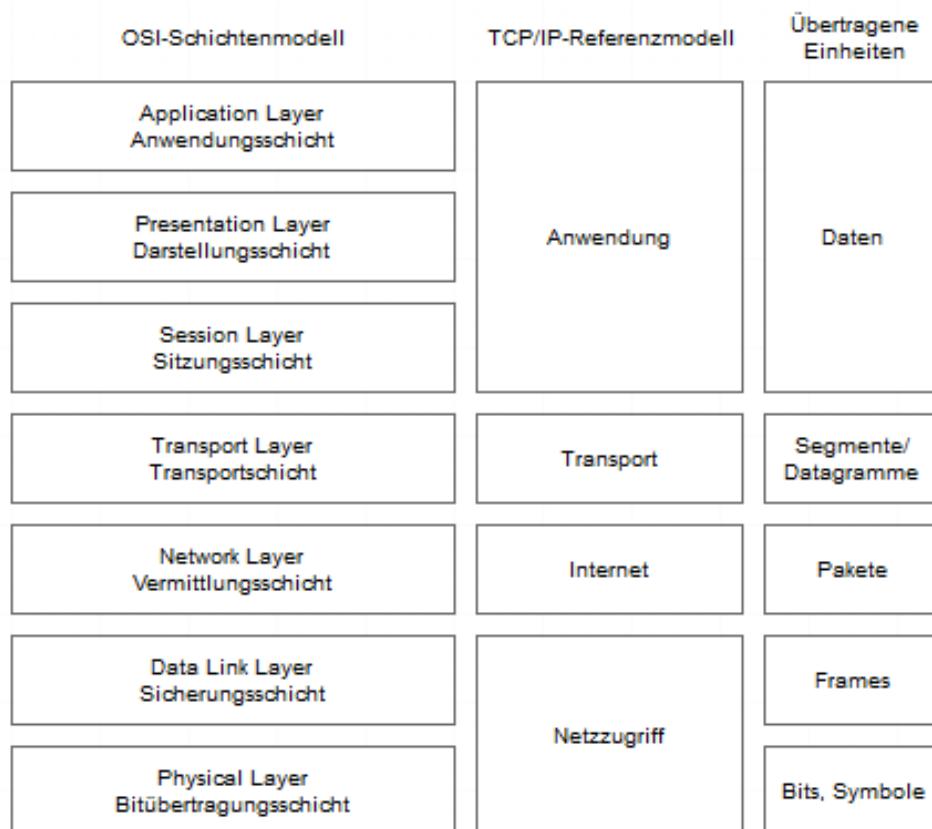


Abbildung 4: OSI- und TCP/IP-Referenzmodell

## 2. TECHNISCHER HINTERGRUND

Frames können in drei Kategorien unterteilt werden:

Datenframes, welche Daten von höheren Schichten übertragen, z.B. TCP-Segmente aus der Transportschicht.

Control-Frames, die bei der Übertragung der Datenframes den Kontrollfluss steuern und die Zuverlässigkeit der WLAN-Übertragung gewährleisten. Beispiele für Control-Frames sind Request-To-Send- oder Acknowledgment-Nachrichten.

Die dritte Kategorie sind die Management-Frames, welche verschiedene Dienste auf dem Data Link Layer für WLAN anbieten, die im Kabelgebundenen LAN nicht benötigt werden. Ein Beispiel hierfür wäre die Identifizierung von verfügbaren Netzwerken. Die in dieser Arbeit untersuchten Probe-Requests gehören in diese dritte Kategorie. In der Abbildung 5 ist ein Management-Frame dargestellt, wie es üblicherweise versendet wird. Die MAC-Header-Informationen sind in jedem Frame enthalten, aber die Information Elements können sich je nach Typ des Management-Frames unterscheiden. Eine Erklärung der einzelnen Felder ist in der Tabelle 3 angegeben.

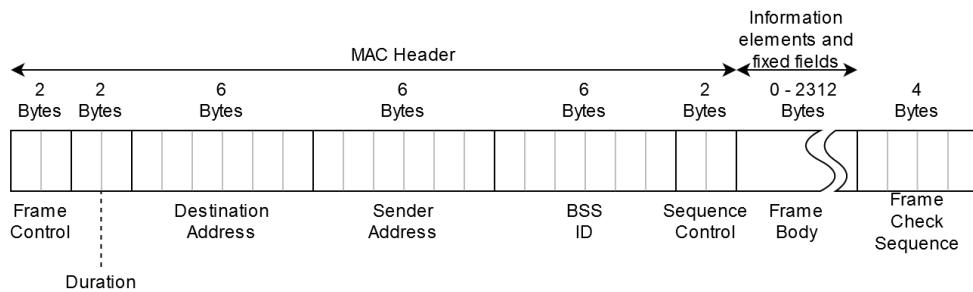


Abbildung 5: Generisches Management-Frame

## 2. TECHNISCHER HINTERGRUND

---

Feld	Länge	Bedeutung
Frame Control	2 Byte	Gibt die Protokollversion und Subtypen-Informationen für das Frame an.
Duration	2 Byte	Gibt die Zeit an, wie lange das Übertragungsmedium während der Übertragung des Frames nicht verwendet werden darf, um Kollisionen zu vermeiden.
Dest. Address	6 Byte	MAC-Adresse des Empfängers des Frames.
Sender Address	6 Byte	MAC-Adresse des Sendegeräts des Frames.
BSSID	6 Byte	BSSID steht für Basic Service Set Identifier und beschreibt eine Zuordnung verschiedener Geräte zu einem spezifischen Netzwerk. Üblicherweise ist die BSSID die MAC-Adresse des Access Points
Sequence Control	2 Byte	Dieses Feld besteht aus einer Fragmentnummer und einer Sequenznummer. Die Fragmentnummer beschreibt die Anzahl Frames, die für einen Request benötigt werden. Die Sequenznummer wird für jede Übertragung separat festgelegt und verwendet, falls Übertragungen wiederholt werden müssen.
Frame-Body	variabel	Im Frame-Body werden die spezifischen Informationen des jeweiligen Frames übertragen.
Frame Check Sequence	2 Byte	Redundanzcheck für die Validierung, dass das Frame korrekt übertragen wurde.

Tabelle 3: Felder in einem Management-Frame

## 2. TECHNISCHER HINTERGRUND

Weiterhin können Frames in drei verschiedene Klassen unterteilt werden. Die Klassen bestimmen, welche Frames - abhängig vom Zustand des Geräts - überhaupt übertragen werden dürfen. Die verschiedenen Zustände und die dazugehörigen erlaubten Klassen sind in der Abbildung 6 dargestellt. Im initialen Zustand sind Geräte nicht authentifiziert und nicht einem Access Point zugeordnet. Im zweiten Zustand hat sich das Gerät gegenüber dem Access Point authentifiziert, ist aber noch nicht zugewiesen. Im dritten Zustand ist das Gerät authentifiziert und zugewiesen. Erst im dritten Zustand ist der Austausch von Daten erlaubt.

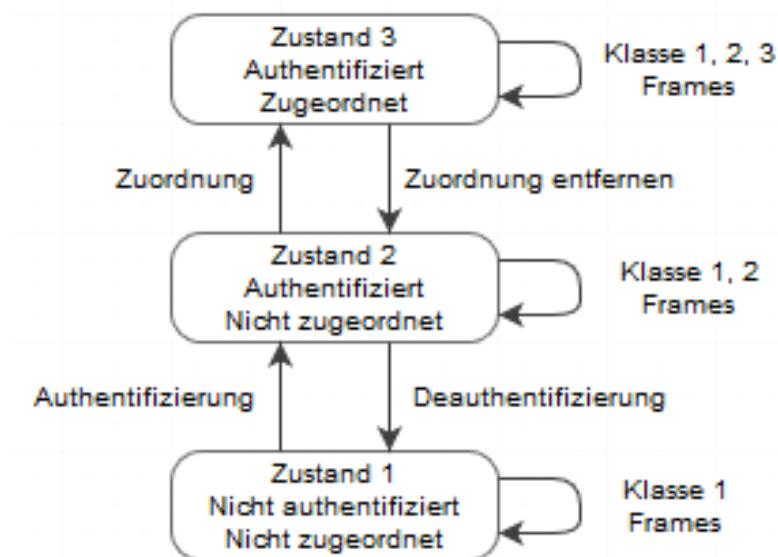


Abbildung 6: Zustände und Klassen für Frames

## 2.4 Probe-Requests

Probe-Requests sind Klasse-1 Management-Frames. Sie werden von Endgeräten verwendet, um innerhalb des Sendebereiches nach WLAN-Access-Points zu suchen. Alternativ können Netzwerke auch gefunden werden, indem ausgesendete Beacon-Messages der Access Points empfangen werden. Um sich mit einem Access Point zu verbinden, sendet ein Gerät zuerst Probe-Requests als Broadcast aus, um verfügbare Access Points kennen zu lernen. Alle Access Points im Sendebereich des Geräts, die den Request erhalten, senden eine Probe-Response, um die vom Gerät benötigten Parameter und die eigenen Betriebsparameter zu übermitteln. Das Gerät evaluierter die erhaltenen Responses, wählt für sich den besten Access Point aus und sendet diesem einen Auth.-Request mit den für die Authentifizierung benötigten Informationen. Sind diese Informationen für den Access Point korrekt, sendet dieser ein Auth.-Response und registriert das Gerät bei sich. Das Gerät sendet daraufhin einen Association-Request welcher vom Access Point mit einem Association-Response beantwortet wird, bevor dann der Austausch von Daten zwischen Endgerät und Access Point beginnt. Der gesamte Ablauf ist in der Abbildung 7 dargestellt.

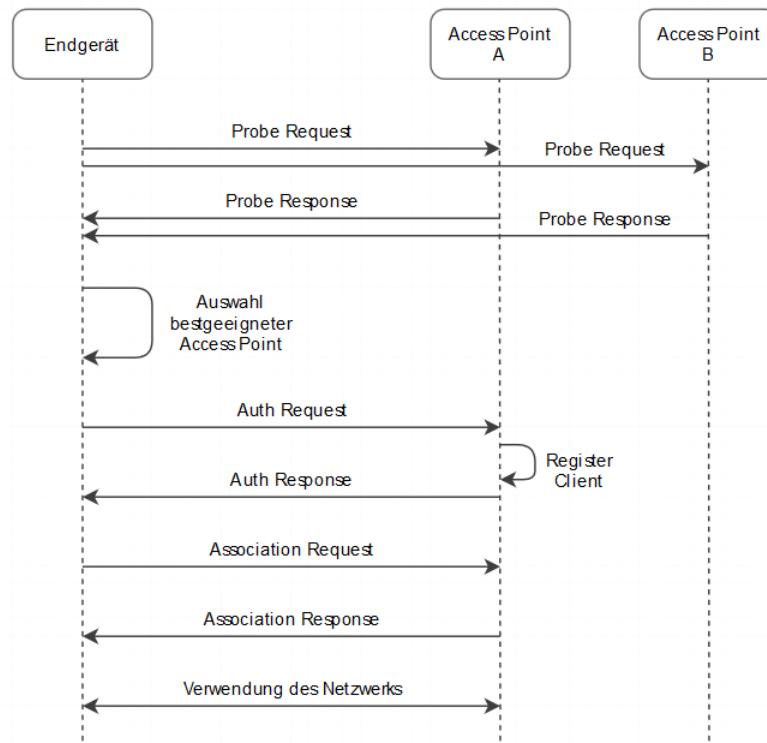


Abbildung 7: Sequenzdiagramm Association und Authentifizierung

## 2. TECHNISCHER HINTERGRUND

---

Probe-Requests beinhalten alle notwendigen Informationen, die ein Access Point benötigt, um zu entscheiden, ob er die Anforderungen des Client-Geräts erfüllen kann. In der Abbildung 8 ist das Frame eines Probe-Requests ersichtlich und in der Tabelle 4 sind die einzelnen Felder erklärt.

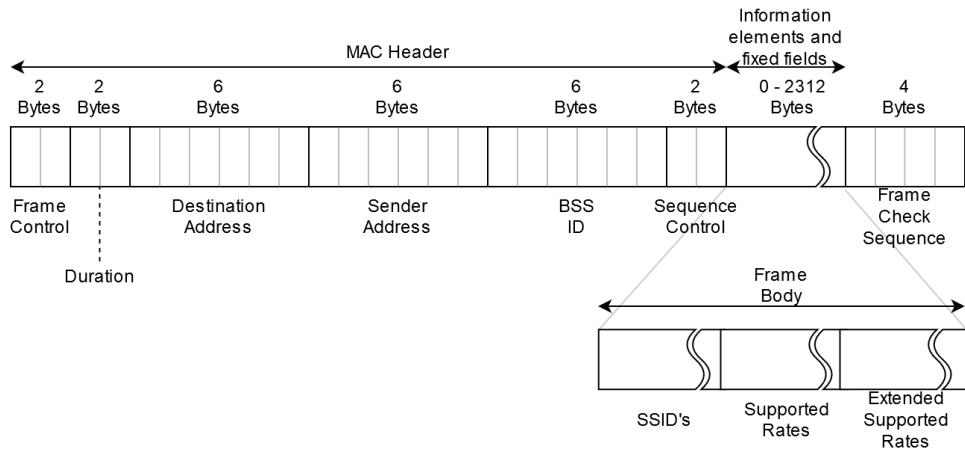


Abbildung 8: Frame eines Probe-Requests.

Feld	Länge	Bedeutung
SSID	variabel	Abfolge von Zeichen, um ein Netzwerk zu identifizieren. Wird manchmal auch als Netzwerkname bezeichnet.
(Extended) Supported Rates	variabel	Datenraten, die vom Client-Gerät unterstützt werden. Im IEEE 802.11 sind verschiedene Datenraten von 1Mbit/s bis 54MBit/s standardisiert.

Tabelle 4: Felder in einem Probe-Request

## 2.5 Probe-Request-Burst

Mobilgeräte senden Probe-Requests in Gruppen aus, welche Bursts genannt werden. Bursts zeichnen sich dadurch aus, dass jeder Probe-Request darin dieselbe MAC-Adresse verwendet. Die Anzahl Probe-Request kann dabei von Burst zu Burst variieren. Weiterhin sind die Sequenznummern innerhalb eines Bursts aufsteigend, auch wenn die Sequenznummer vom Mobilgerät zufallsgeneriert sind. Die Information-Element-Felder der Probe-Requests in einem Burst sind ebenfalls identisch.

Diese Information kann in einem Prototyp dazu verwendet werden, aufgezeichnete Frames in Bursts zu sortieren, bevor diese klassifiziert werden. Dadurch müssen weniger einzelne Frames klassifiziert werden und die Effizienz eines Prototyps wird gesteigert, während die Fehlerrate gesenkt wird.

Die Abbildung 9 zeigt eine Aufzeichnung von Probe-Requests, und wie deren Aufteilung in Bursts erkannt werden kann.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	62:5e:da:65:95:e5	Broadcast	802.11	112	Probe Request, SN=397, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
2	0.055076	62:5e:da:65:95:e5	Broadcast	802.11	112	Probe Request, SN=398, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
3	0.218328	da:0a:89:1f:0a:39	Broadcast	802.11	112	Probe Request, SN=2791, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
4	0.238513	da:0a:89:1f:0a:39	Broadcast	802.11	112	Probe Request, SN=2793, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
5	0.331559	aa:16:4c:6:c:ba:f4	Broadcast	802.11	112	Probe Request, SN=4081, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
6	0.367807	aa:16:4c:6:c:ba:f4	Broadcast	802.11	112	Probe Request, SN=4085, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
7	0.437615	aa:16:4c:6:c:ba:f4	Broadcast	802.11	112	Probe Request, SN=4089, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
8	0.457828	aa:16:4c:6:c:ba:f4	Broadcast	802.11	112	Probe Request, SN=4090, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
9	18.382444	92:88:cc:4:e:8d:e8	Broadcast	802.11	112	Probe Request, SN=438, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
10	18.437634	92:88:cc:4:e:8d:e8	Broadcast	802.11	112	Probe Request, SN=439, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
11	18.596597	12:4e:26:5:a:a:0:88	Broadcast	802.11	112	Probe Request, SN=3847, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
12	18.638449	12:4e:26:5:a:a:0:88	Broadcast	802.11	112	Probe Request, SN=3849, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
13	18.692528	de:76:6:f:79:7:a:16	Broadcast	802.11	112	Probe Request, SN=2742, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
14	18.728644	de:76:6:f:79:7:a:16	Broadcast	802.11	112	Probe Request, SN=2746, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
15	18.748802	de:76:6:f:79:7:a:16	Broadcast	802.11	112	Probe Request, SN=2748, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
16	18.818890	de:76:6:f:79:7:a:16	Broadcast	802.11	112	Probe Request, SN=2752, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
17	18.839214	de:76:6:f:79:7:a:16	Broadcast	802.11	112	Probe Request, SN=2753, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
18	18.863665	de:76:6:f:79:7:a:16	Broadcast	802.11	112	Probe Request, SN=2754, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
19	18.883894	de:76:6:f:79:7:a:16	Broadcast	802.11	112	Probe Request, SN=2755, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
20	18.953946	de:76:6:f:79:7:a:16	Broadcast	802.11	112	Probe Request, SN=2761, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)
21	18.974108	de:76:6:f:79:7:a:16	Broadcast	802.11	112	Probe Request, SN=2763, FH=0, Flags=....., C, SSID=Wildcard (Broadcast)

Abbildung 9: Wireshark-Aufzeichnung von Probe-Requests. Die Bursts sind farbig gekennzeichnet.

Die Probe-Requests in dieser Abbildung stammen alle vom selben Gerät. Man kann erkennen, dass die MAC-Adresse und Sequenznummer für jeden Burst neu zufallsgeneriert wird und dass die Frames alle dieselbe Länge haben, was bedeutet, dass sie die selben Information-Element-Felder haben.

### 2.6 Probe-Request Header-Frame

Der MAC-Header in einem Probe-Request hat einige für Fingerprinting interessante Felder. Bevor die MAC-Adresse zufällig gesetzt wurde, konnte ein Client-Gerät einfach anhand der Sendeadresse erkannt und verfolgt werden. Seit einigen Android/iOS-Versionen werden diese Adressen aber zufallsgeneriert. Die Felder, welche für ein Fingerprinting genutzt werden können, werden nachfolgend genauer erläutert:

#### Frame Control

Das Feld enthält die in der Tabelle 5 beschriebenen Attribute.

In Probe-Requests sind die meisten Attribute auf 0b0 gesetzt. Das Protokoll-Version-Attribut und der Typ (Management) sind auch auf 0b0 und der Subtyp auf 0b0100 (4: Probe-Request) gesetzt.

Diese Informationen können für die Filterung von aufgezeichneten Probe-Requests verwendet werden, sind in der Unterscheidung von verschiedenen Geräten aber nicht relevant.

#### Duration

Das Duration-Feld wird unter anderem dazu verwendet, dem Empfänger mitzuteilen, wie lange die Übertragung für das aktuelle Frame etwa dauern wird. Bei Probe-Requests beträgt die geschätzte Zeit Null Mikrosekunden. Daher lässt sich das Duration-Feld für Fingerprinting auch nicht verwenden.

#### Destination Address

Die Empfängeradresse bei Probe-Requests ist üblicherweise die Broadcast-Adresse (ff:ff:ff:ff:ff:ff). Auch dieses Feld lässt sich für Fingerprinting nicht verwenden.

#### Sender Address

Die MAC-Adresse des Absenders ist, vorausgesetzt die richtigen Geräteeinstellungen sind gesetzt, randomisiert. Es ist möglich, dass nur die NIC (Netzwerkcontroller spezifische Kennung) zufällig gesetzt wird, oder dass die gesamte MAC-Adresse randomisiert ist. Dieses Verhalten soll in der Bachelorarbeit ermittelt werden.

## 2. TECHNISCHER HINTERGRUND

---

<b>Attribut</b>	<b>Länge</b>	<b>Bedeutung</b>
Protokoll Version	2 bit	Dieses Attribut unterscheidet verschiedene WLAN-Standards. Falls ein WLAN-2-Standard entwickelt würde, kann dies in der Protokoll Version vom aktuellen Standard unterschieden werden.
Type	2 bit	Das Type-Attribut unterscheidet zwischen Control-, Daten- und Management-Frames
Subtype	4 bit	Der Subtype unterscheidet die verschiedenen Frames zusätzlich. Probe-Requests haben den Subtype 4 (0b0100)
To DS	1 bit	Wird gesetzt, wenn das Frame von einem Client-Gerät zum Access Point gesandt wird.
From DS	1 bit	Wird gesetzt, wenn das Frame vom Access Point zum Client-Gerät gesandt wird.
More Fragments	1 bit	Wird gesetzt, falls bei Datenframes die Daten in mehreren Fragmenten übertragen werden.
Retry	1 bit	Falls ein Frame nochmals gesendet werden muss, weil ein Fehler passiert ist, wird dieses Bit auf 1 gesetzt.
Power Management	1 bit	Akkubetriebene Geräte signalisieren mit diesem Attribut, dass sie nach dem Versenden des Frames in den Energiesparmodus wechseln und der Access Point allfällige Antworten zwischenspeichern soll.
More Data	1 bit	Falls ein Access Point Daten für ein Client-Gerät zwischenspeichert, dann wird dieses Attribut gesetzt, um zu indizieren, dass das Frame für ein Gerät im Ruhemodus gedacht ist.
WEP	1 bit	Dieses Attribut indiziert, dass der Frame-Inhalt verschlüsselt übertragen wird.
Order	1 bit	Wenn dieses Attribut gesetzt ist, dann werden Frames und dazugehörige Fragmente in der korrekten Reihenfolge übertragen.

Tabelle 5: Attribute im Frame-Control-Feld

## 2. TECHNISCHER HINTERGRUND

### **BSSID**

Basic Service Sets werden vergeben, um verschiedene WLAN-Netze im selben Gebiet untereinander zu unterscheiden. Wie bei der Empfängeradresse wird in Probe-Requests für die BSSID auch die Broadcast-Adresse (ff:ff:ff:ff:ff:ff) verwendet und kann für ein Fingerprinting nicht verwendet werden.

### **Sequence Control**

Das Sequence Control Feld hat zwei Attribute. Die Fragmentnummer wird verwendet, um Datenframes, die zu gross sind und in einzelne Fragmente aufgeteilt werden, in die korrekte Reihenfolge zu sortieren. Die Sequenznummer dient der Identifikation der logischen Abfolge von Frames. Fragmentierte oder wiederholt gesendete Frames haben die gleiche Sequenznummer. Die Sequenznummer würde sich für ein Fingerprinting eignen, falls sie nicht bei neuen Probe-Requests jedes Mal zufällig gesetzt wird.

### **SSID**

In Probe-Request-Frames wird das SSID Field verwendet, um ein Probe-Request zu einem spezifischen Access Point (z.B. Eduroam) zu senden. Falls der Probe-Request an alle verfügbaren Netzwerke versendet wird, wird die Broadcast-SSID, manchmal auch Wildcard-SSID genannt, verwendet. Spezifische SSIDs können für ein Fingerprinting interessant sein. Wenn ein Client-Gerät in verschiedenen Netzwerken Probe-Requests an dieselbe SSID sendet, kann das Gerät, falls keine weiteren Geräte nach dieser SSID suchen, erkannt werden.

### **(Extended)-Supported-Rates**

Mit dem (Extended)-Supported-Rates Feld teilt das Client-Gerät dem Access Point mit, welche WLAN-Datenraten unterstützt werden. Der Access Point entscheidet unter anderem anhand dieser Datenraten, ob er dem Client-Gerät einen Probe-Response senden wird. Die unterstützten Datenraten könnten für eine Erkennung von Geräten genutzt werden.

## 2. TECHNISCHER HINTERGRUND

### **Frame Check Sequence**

Die Frame Check Sequence ist eine 32 Bit lange hexadezimale Zahl, die anhand der im Frame gesetzten Parameter berechnet wird und eine Überprüfung der Frames beim Empfänger ermöglicht. Wird ein Frame empfangen, kann der Empfänger die Frame Check Sequence über das Frame berechnen und diese mit dem im Frame mitgelieferten Wert vergleichen. Falls die beiden Werte nicht übereinstimmen, wird das Frame verworfen, oder allenfalls neu angefordert. Die Frame Check Sequence lässt sich nicht für ein Fingerprinting verwenden.

### **Weitere Information-Element-Felder**

Es ist möglich, dass neben den vorgehend erwähnten Feldern im Frame, welche alle vorhanden sein müssen, noch optionale Information-Element-Felder übermittelt werden, um weitere Funktionalitäten abzudecken oder weitere Eigenschaften des Client Geräts zu übermitteln. Diese IE-Felder sind für das Fingerprinting wahrscheinlich die interessanteste Quelle von Informationen, wenn diese sich von Mobilgerät zu Mobilgerät tatsächlich unterscheiden.

### 3 Verwandte Arbeiten

Es gibt duzende Papers zum Thema Fingerprinting auf Mobilgeräten. Die relevantesten vier Arbeiten wurden in den folgenden Unterabschnitten in der Reihenfolge ihrer Publikation zusammengefasst.

Es gilt zu erwähnen, dass in diesem Abschnitt Informationen aus den Papers direkt übersetzt und übernommen wurde.

#### 3.1 How Talkative is Your Mobile Device?

- Name: How Talkative is your Mobile Device? An Experimental Study of Wi-Fi Probe-Requests
- Autoren: Julien Freudinger
- Publikationsjahr: 2015

Das Paper beschäftigt sich mit der Frage, welche Informationen von einem Mobilgerät über Probe-Requests versendet werden. Die wichtigsten Erkenntnisse sind eine Evaluation für den experimentellen Versuchsaufbau, um Probe-Requests zu sniffen, die Messresultate aus den durchgeföhrten Messungen und eine Evaluation eines kommerziell genutzten MAC-Address-Randomization-Algorithmus.

#### Experimenteller Versuchsaufbau und Resultate

Die Experimente, welche im Paper beschrieben werden, beschreiben acht unterschiedliche Konfigurationen für die Messeinrichtung und weitere acht für die Mobilgeräte und die aus den Messungen gewonnenen Resultate. Für die Messungen wurden ein Samsung S3 mit Android 4.4.2, ein iPhone 6 mit IOS 8.1.3 und ein Nexus 5 mit Android L 5.0.1 verwendet.

### 3. VERWANDTE ARBEITEN

Die Konfigurationen für die Messeinrichtung sind nachfolgend aufgelistet. Die Kanäle 1, 6 und 11 werden nachfolgend als reservierte Kanäle bezeichnet.

- 1.dynamic: Eine Antenne, welche über die drei reservierten 2.4-GHz-Kanäle misst.
- 1.static: Drei Mess-Konfigurationen, jeweils statisch auf einem separaten reservierten 2.4-GHz-Kanal.
- 3.dynamic: Drei Antennen, die koordiniert über die reservierten 2.4-GHz-Kanäle messen. (Die Antennen springen in regelmässigen Abständen auf den nächsten Kanal und sind jeweils um einen Kanal verschoben)
- 3.dynamic.s: Drei Antennen, die über die reservierten 2.4-GHz-Kanäle messen. (Jede Antenne springt frei über die reservierten Kanäle.)
- 3.dynamic.sn: Drei Antennen die über die reservierten 2.4-GHz-Kanäle und die in der Tabelle 2 erwähnten 5-GHz-Kanäle messen.
- 3.static: Pro reserviertem Kanal wird jeweils eine Antenne statisch eingestellt.

Die Messungen wurden jeweils eine Stunde lang durchgeführt und fünfmal wiederholt. Als Ergebnisse wurden die durchschnittlich aufgezeichnete Anzahl der Probe-Requests und eine Schätzung der nicht aufgezeichneten Requests gespeichert. Die nicht aufgezeichneten Probe-Requests können aus den Sequenznummern abgeschätzt werden. Werden in einer Messung beispielsweise drei Probe-Requests mit den Sequenznummern 1, 3 und 5 aufgezeichnet, kann davon ausgegangen werden, dass zwei Probes mit den Sequenznummern 2 und 4 nicht aufgezeichnet wurden.

Die Verwendung von mehreren Antennen ist ein Ansatz, welcher in anderen Arbeiten z.T. nicht beachtet wurde und dazu führen kann, dass nur ca. 57 Prozent der Probe Requests tatsächlich gemessen und in den Ergebnissen berücksichtigt wurden.

### 3. VERWANDTE ARBEITEN

Die Konfiguration der Mobilgeräte ist nachfolgend aufgelistet:

- Default: Das Mobilgerät ist gesperrt, an ein Ladekabel angeschlossen, nicht mit einem WLAN verbunden und Bluetooth ist ausgeschaltet. Das Gerät kennt vier Netzwerke.
- Not Charging: Das Mobilgerät ist nicht an ein Ladekabel angeschlossen.
- Screen ON: Das Gerät wird alle fünf Minuten entsperrt und wieder gesperrt.
- Wi-Fi Connected: Das Gerät ist mit einem WLAN verbunden.
- Wi-Fi Settings ON: Das Mobilgerät befindet sich im Menu für die Auswahl des WLANs, es wird aber sonst nicht mit dem Gerät interagiert.
- Bluetooth ON: Bluetooth ist während den Messungen eingeschaltet.
- Airplane ON: Der Flugmodus ist während den Messungen eingestellt und das WLAN auf dem Mobilgerät eingeschaltet.
- Known In Proximity: Ein dem Gerät bekanntes WLAN erscheint alle fünf Minuten für 10 Sekunden im Empfangsbereich des Mobilgeräts.

Die Resultate zeigen, dass wenn die WLAN-Einstellungen eingeschaltet werden oder wenn der Bildschirm aktiviert wird, die Anzahl ausgesendeter Probe-Requests ansteigt. Eine Ausnahme ist hierbei das Nexus 5, welches keinen signifikanten Unterschied bei den ausgesendeten Probes aufweist.

Weiterhin wurde gemessen, welche Auswirkung die Anzahl bekannter Netzwerke auf einem Mobilgerät auf die Anzahl ausgesendeter Probe-Requests hat. Das iPhone sendet erst ab 20 bekannten SSIDs eine erhöhte Anzahl von Probes aus. Auf dem Nexus werden unabhängig von den bekannten Netzwerken immer etwa gleich viel Probe-Requests ausgesandt. Lediglich das Samsung S3 versendet mit steigender Anzahl bekannter Netzwerke eine erhöhte Anzahl Probes.

Im Paper wird jeweils davon ausgegangen, dass diese aufgezeichneten Geräteverhalten vom Betriebssystem abhängen. Es wurden keine Messungen mit unterschiedlichen Geräten mit demselben Betriebssystem vorgenommen, um den Einfluss von Hardware auf das Probing- Verhalten zu untersuchen.

### 3. VERWANDTE ARBEITEN

#### **MAC-Address-Randomization-Algorithmus**

In den durchgeföhrten Messungen werden verschleierte MAC-Adressen des iPhones untersucht und festgestellt, dass die Sequenznummern im iOS 8 nicht zufällig initialisiert werden, sondern aufsteigend vorkommen. Ausserdem wird in iOS 8 die MAC-Adresse nur dann verschleiert, wenn das Gerät im Ruhemodus ist. Das bedeutet, dass iPhones mit iOS 8 relativ einfach identifiziert werden können, wenn man die nicht randomisierte MAC-Adresse aufzeichnet und die Sequenznummern mit denjenigen Probe-Requests vergleicht, die eine verschleierte MAC-Adresse haben. Hat man beispielsweise in Wireshark einen Probe mit der MAC ‘Apple\_12:84:05’ und der Sequenznummer ‘1337’ und einen Probe-Request mit der MAC ‘52:61:6E:64:6F:6D’ und der Sequenznummer ‘1340’, kann man davon ausgehen, dass die beiden Probe-Requests vom selben Gerät ausgesendet wurden.

Weiterhin senden Mobilgeräte mit iOS 8 oder Android 4 und 5 in den Information- Element-Feldern herstellerspezifische Informationen mit, welche es zusätzlich vereinfachen, Probe-Requests mit verschleierten MAC-Adressen zu einem spezifischen Gerät zuzuordnen.

## 3.2 Defeating MAC Address Randomization Through Timing Attacks

- Name: Defeating MAC Address Randomization Through Timing Attacks
- Autoren: Célestin Matte et al.
- Publikationsjahr: 2016

In diesem Paper wird ein Verfahren erarbeitet, mit dem die Verschleierung von MAC-Adressen mittels eines Timing-Angriffs neutralisiert wird. Es wird davon ausgegangen, dass die verschiedenen Betriebssysteme dahingehend angepasst wurden, dass die Mobilgeräte keine identifizierenden Informationen mehr aussenden und nur das Timing der Probe-Requests für ein mögliches Fingerprinting verwendet werden kann. Dabei wird vorausgesetzt, dass die Frames, die von Geräten ausgesendet werden, einem Muster folgen, welches sich für eine Identifikation nutzen lässt.

MAC-Adressen werden in vielen Implementationen nach einigen Probe-Requests neu randomisiert. Eine Gruppe von Probe-Requests, die in einem Zeitfenster von ungefähr zehn Millisekunden ausgesandt werden, wird auch Burst genannt. Außerdem ist in der Praxis die tatsächliche Anzahl Mobilgeräte im Empfangsbereich eines WLAN-Access-Points nicht bekannt. Die Schwierigkeit eines solchen Ansatzes besteht darin, mit einer sehr geringen Menge von Informationen ein zuverlässiges Fingerprinting durchzuführen zu können, ohne dass zuvor eine Datensammlung möglich gewesen ist. Die Autoren beschreiben als Lösung für diese Problematik die Verwendung inkrementeller Lernmethoden für die Clusterbildung unter der Verwendung einer handelsüblicher Netzwerkkarte, die nur einen Kanal gleichzeitig überwachen kann.

### Zeitbasierte Signatur

Probe-Requests lassen sich gruppieren, indem die Inter-Frame-Arrival-Time (Zwischenankunftszeit, nachfolgend IFAT genannt) der einzelnen Frames mit der selben MAC-Adresse analysiert wird. Der Algorithmus 1 beschreibt in Pseudocode, wie ein Identifikator zu einem Frame hinzugefügt werden kann.

**Input:**

$G$ : groups of burst sets, grouped by MAC

$t$ : distance threshold

$d$ : a distance function

**Result:**  $A$ : dictionary of aliases

```

1  $A \leftarrow \emptyset;$ 
2  $D \leftarrow \emptyset;$                                 // Database of signatures
3 foreach  $B \in G$  do
4    $S \leftarrow \text{signature}(B);$ 
5    $d_{\min} \leftarrow \min(d(S, S') \text{ where } S' \in D);$ 
6   if  $d_{\min} < t$  then
7     |  $A[B.mac] \leftarrow A[S'.mac];$ 
8   end
9   else
10    |  $A[B.mac] \leftarrow B.mac;$ 
11  end
12   $D \leftarrow D \cup S$ 
13 end
14 return  $A$ 

```

Algorithmus 1: Algorithmus für die Identifikation von MAC-Frames

Der Algorithmus hat als Input ein beliebiges Capture-File mit gemessenen Probe-Requests und liefert als Output ein Mapping von Identifikatoren und den Frames. Für alle Frames mit der selben MAC-Adresse wird die IFAT, die Auftretenswahrscheinlichkeit und der Durchschnitt der IFAT berechnet. Anhand dieser Berechnungen wird basierend auf der eingegebenen Distanzgrenze (distance threshold) entschieden, ob zwei Signaturen zum selben Gerät gehören.

Im Paper werden drei Distanzmass-Algorithmen genannt, auf die hier nicht weiter eingegangen wird. Bei Interesse des Lesers empfiehlt sich das Studium des Papers. Weiterhin wird der Algorithmus an einem Datenset von 120'000 Proberequests validiert und hat eine Erfolgsquote von ungefähr 77,2 Prozent.

#### 3.3 Why MAC Address Randomization is not Enough

- Name: Why MAC Address Randomization is not Enough: An Analysis of Wi-Fi Network Discovery Mechanisms
- Autoren: Mathy Vanhoef et al.
- Publikationsjahr: 2016

In der Recherche zum Thema Mobile Fingerprinting taucht dieses Paper in sehr vielen anderen Quellen als Referenz auf. Darin wird eine fundierte Analyse diverser Methoden für die Deanonymisierung von Mobilgeräten vorgenommen und erwiesen, dass die Verschleierung von MAC-Adressen allein keine Garantie für die Privatsphäre der Nutzer ist.

Der Hauptansatz besteht in der Analyse von Information-Element-Feldern und den Sequenznummern, um Mobilgeräte voneinander zu unterscheiden. Insgesamt werden zwölf IE-Felder genannt, die für ein Fingerprinting relevant sein können und nachfolgend aufgelistet sind.

- HT capabilities info
- Ordered list of tags numbers
- Extended capabilities
- HT A-MPDU parameters
- HT MCS set bitmask
- Supported rates
- Interworking access net. type
- Extended supported rates
- WPS UUID
- HT extended capabilities
- HT TxB beam Forming Cap.
- HT Antenna Selection Cap.

Weiterhin könnte die Reihenfolge der IE-Tags eine weitere Informationsquelle für ein mögliches Fingerprinting sein.

Auch in diesem Paper wird ein Algorithmus beschrieben, welcher ein Fingerprinting auf Mobilgeräte ermöglicht.

**Input:**

$P$ : List of captured probe requests  
 $\Delta T$ : maximum time between two probes  
 $\Delta S$ : maximum sequence number distance  
**Result:** Set of clusters corresponding to devices

```

1  $M \leftarrow \emptyset;$  //  $M$  maps fingerprints to clusters
2 forall  $p \in P$  do
3    $f \leftarrow \text{fingerprint}(p);$  // Calculate IE fingerprint
4    $M[f].append(p);$  // Append probe to cluster
5 end
6  $D \leftarrow [];$  // List of clusters representing devices
7 forall  $C \in M$  do
8    $S \leftarrow [];$  // Will contain subdivision of  $C$ 
9    $m \leftarrow \max(p.seq \text{ for } p \text{ in } C);$ 
10  forall  $p \in C$  do
11    Find  $i$  such that: // Find matching cluster
12       $d(S[i].last.seq, p.seq, m) \leq \Delta S;$ 
13      and  $p.time - S[i].last.time \leq \Delta T;$ 
14    if no  $i$  found then
15       $| i \leftarrow |S|;$  // Create new subcluster
16    end
17     $S[i].append(p);$  // Add  $p$  to subcluster
18  end
19   $D.extend(S);$  // Extend list  $D$  with  $S$ 
20 end
21 return  $D$ 
```

Algorithmus 2: Gruppierungsalgorithmus basierend auf IE-Feldern

Der Algorithmus berechnet zuerst für alle Probe-Requests einen Fingerprint anhand der IE-Felder und weist diesen einem Cluster zu. Die Cluster werden basierend auf den Sequenznummern in Gruppen unterteilt.

Da in modernen Betriebssystemen die Sequenznummer auch zufällig gesetzt wird, kann das Clustering nicht mehr anhand der Sequenznummer durchgeführt werden. In einem eigenen Prototypen müsste für die Clusterbildung eine eigene Methodik entworfen werden.

#### 3.4 Noncooperative 802.11 MAC Layer Fingerprinting

- Name: Noncooperative 802.11 MAC Layer Fingerprinting and Tracking of Mobile Devices
- Autoren: Pieter Robyns et al.
- Publikationsjahr: 2017

In diesem Paper wird eine pro-Bit-Entropie Analyse von Probe-Request-Frames durchgeführt und eine Technik vorgestellt, die mittels eines Fingerprints und zeitlichen Daten ein Gerät mit bis zu 80 Prozent Genauigkeit erkennen und verfolgen kann, ohne dass dafür spezielle Hardware benötigt wird.

##### Identifikatoren und Fingerprinting

Es wird zwischen zwei Typen von Identifikatoren unterschieden. Explizite Identifikatoren, zu denen eine MAC-Adresse gehört, erlauben es, ein Gerät eindeutig zu identifizieren. Implizite Identifikatoren sind Informationen, die nicht absichtlich für die Identifikation gedacht sind, aber sich von Gerät zu Gerät genügend unterscheiden, um damit ein Fingerprinting durchzuführen. Unter der Annahme, dass die MAC-Adresse in Mobilgeräten anonymisiert wird, müssen für ein Fingerprinting die impliziten Identifikatoren verwendet werden.

##### Bitweise MAC-Header-Analyse

In weiteren Arbeiten wird oftmals ein spezifisches Element des MAC-Headers für ein Fingerprinting verwendet und das restliche Frame verworfen. Dieses Paper schlägt vor, anhand der Entropie der einzelnen Header-Felder automatisch zu evaluieren, welche Informationen für ein Fingerprinting verwendet werden sollen. Dabei wird davon ausgegangen, dass das Mobilgerät jedes Frame einzeln anonymisiert und nicht mit einem Access Point assoziiert ist. Das bedeutet, die vorgeschlagene Technik ist für die Anwendung auf ein einzelnes Frame ausgelegt.

Im Paper werden drei wichtige Metriken vorgestellt:

- Bit-Variabilität: Einzelne Bits unterscheiden sich von Gerät zu Gerät dahingehend, dass sie sich für ein Fingerprinting eignen.
- Bit-Stabilität: Für ein Gerät bleiben die Bits, die sich zu anderen Geräten unterscheiden, in nacheinander auftretenden Frames soweit stabil, dass sie weiterhin für einen Fingerprint verwendet werden können.
- Bit-Verwendbarkeit: Die Verwendbarkeit ist eine Kombination der Variabilität und der Stabilität und sagt aus, ob das Bit für ein Fingerprinting tatsächlich verwendet wird.

### Berechnung der Variabilität

Die Variabilität ist die Entropie eines Bits gemessen über mehrere Frames welche jeweils von einem anderen Mobilgerät ausgesendet werden. Dabei wird pro Gerät nur ein einzelnes Frame in Betracht gezogen, um keinen Bias in das System einzufügen.

Um die Entropie eines Bits an der Position  $i$  in einem Frame zu berechnen, wird zuerst die diskrete Wahrscheinlichkeitsdichte ( $P(X_i)$ ) für jeden Bitwert an der Position  $i$  berechnet.  $X_i$  ist dabei die Zufallsvariable, welche einen Bitwert repräsentiert. Somit ist  $X_i \in \{0, 1, U\}$ , wobei  $U$  bedeutet, dass das Bit nicht vorhanden ist. Die Wahrscheinlichkeitsdichte kann dann verwendet werden, um die Shannon-Entropie zu berechnen, wie in der Formel 1 dargestellt.

$$H(X_i) = - \sum_{x \in \{0, 1, U\}} P(X_{ix}) \log_3 P(X_{ix}) \quad (1)$$

Da für die Berechnung der Entropie ein Bit mit drei möglichen Zuständen (Wert 1 oder 0 und nicht vorhanden) beschrieben wird, wird der  $\log_3$  anstatt dem üblichen  $\log_2$  verwendet.  $H(X_i)$  hat nun einen Wert zwischen 0 und 1, wobei 0 bedeutet, dass keine Entropie vorhanden ist und 1 bedeutet, dass eine maximale Entropie vorhanden ist. Das Resultat kann als Vektor  $\vec{v}$  repräsentiert werden:

$$\vec{v} = [H_1 \ H_2 \ \cdots \ H_{n-1} \ H_n] \quad (2)$$

Hierbei ist  $n$  die Anzahl der Bits im Frame und  $H_n$  die Entropie des jeweiligen Bits ist.

### Berechnung der Stabilität

Die Stabilität eines Bits wird als 1 minus die Entropie dieses Bits, gemessen über mehrere Frames von einem Gerät, definiert. Der Wert sagt aus, wie hoch die Wahrscheinlichkeit ist, dass das Bit über mehrere Übertragungen gleich bleibt. Der Nutzen der Stabilität ist, dass diejenigen Bits, die sich pro Gerät häufig verändern, im Fingerprinting nicht berücksichtigt werden. Die Formel 3 zeigt den Resultierenden Stabilitätsvektor  $\vec{s}_d$  pro Mobilgerät  $d$ :

$$\vec{s}_d = [1 - H_{d1} \ 1 - H_{d2} \ \cdots \ 1 - H_{dn-1} \ 1 - H_{dn}] \quad (3)$$

### 3. VERWANDTE ARBEITEN

---

Im Gegensatz zur Variabilität bedeutet ein hoher Wert bei der Entropie, dass die Stabilität geringer wird. Darum wird die Stabilität mit  $1 - H(X_i)$  berechnet. Da die Stabilität pro Gerät berechnet wird, muss der Vektor  $\vec{s}$  durch eine Mittelwertsberechnung der Werte des Vektors  $\vec{s}_d$  berechnet werden, um diesen für die weiteren Operationen verwenden zu können.

#### Berechnung der Verwendbarkeit

Im Idealfall sind Variabilität und Stabilität beide genügend hoch, um damit ein Fingerprinting durchführen zu können. Um die Verwendbarkeit - dargestellt durch den Vektor  $\vec{u}$  - zu berechnen, werden im Paper zwei Ansätze vorgestellt, die beide in der Praxis anwendbar sind.

Der erste Ansatz ist ein statistisches Vorgehen, welches die Variabilität und Stabilität als Wahrscheinlichkeit betrachtet, da beide im Wertebereich  $[0, 1]$  sind. Davon ausgehend, dass  $\vec{v}$  und  $\vec{s}$  voneinander unabhängige Variablen sind, kann die Verwendbarkeit mit der Formel

$$\vec{u} = \vec{v} \odot \vec{s} \quad (4)$$

berechnet werden.

Der zweite Ansatz geht davon aus, dass die Stabilität zu bevorzugen ist und deshalb in der Berechnung nur die Variabilität verwendet wird. Diejenigen Bits, deren Stabilität unter einem gewissen Schwellwert  $\lambda$  liegen, werden aus der Berechnung ausgeschlossen.

$$\vec{u}_i = \begin{cases} \vec{v}_i, & \text{if } \vec{s}_i \geq \lambda, \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

Falls beispielsweise der Schwellwert  $\lambda = 1$  gesetzt wird, wird dadurch sichergestellt, dass kombinierte Bits im Fingerprint über die Zeit immer stabil bleiben.

Da die IE-Felder nicht unbedingt in allen Probe-Requests in der gleichen Reihenfolge vorkommen, müssen die Variabilität und Stabilität für die IE-Felder unabhängig ihrer Reihenfolge berechnet werden.

## 4. ALLGEMEINE MOBILGERÄTE-ANALYSE

### 4 Allgemeine Mobilgeräte-Analyse

Bevor in der Bachelorarbeit damit begonnen werden konnte, Auswertungen über Mobilgeräte vorzunehmen, musste die allgemeine Verteilung von Smartphones auf dem Schweizer Markt analysiert werden. Dafür wurden Zahlen vom Bundesamt für Statistik aus dem Jahr 2019 ausgewertet und ergänzend Verkaufszahlen und Nutzerstatistiken bekannter Hersteller gesucht.

#### 4.1 Smartphones

In der Schweiz gibt es im Jahr 2020 ungefähr 6.8 Millionen Personen, die ein Smartphone besitzen. Prognosen sagen aus, dass bis im Jahr 2025 über 7 Millionen Personen ein Mobilgerät besitzen werden. In der Abbildung 10 sind die statistischen Zahlen und Prognosen bis zum Jahr 2022 ersichtlich.

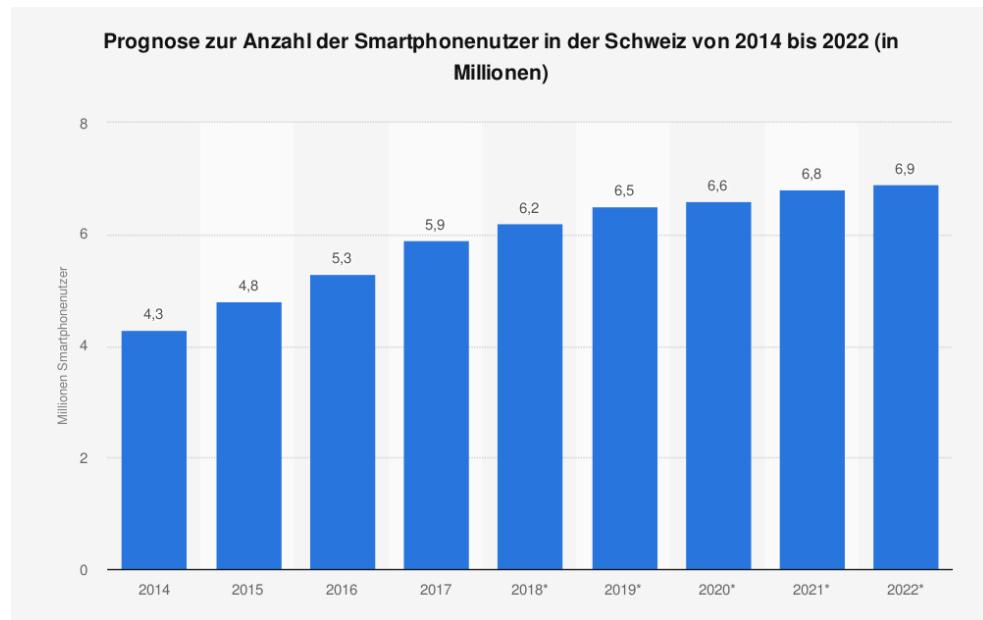


Abbildung 10: Anzahl Smartphone User in der Schweiz

#### 4. ALLGEMEINE MOBILGERÄTE-ANALYSE

Um die Nutzung der Mobilgeräte noch weiter zu unterteilen, wurde eine Aufteilung nach Betriebssystem der Smartphones durchgeführt. Die Ergebnisse sind in der Abbildung 11 aufgezeigt.

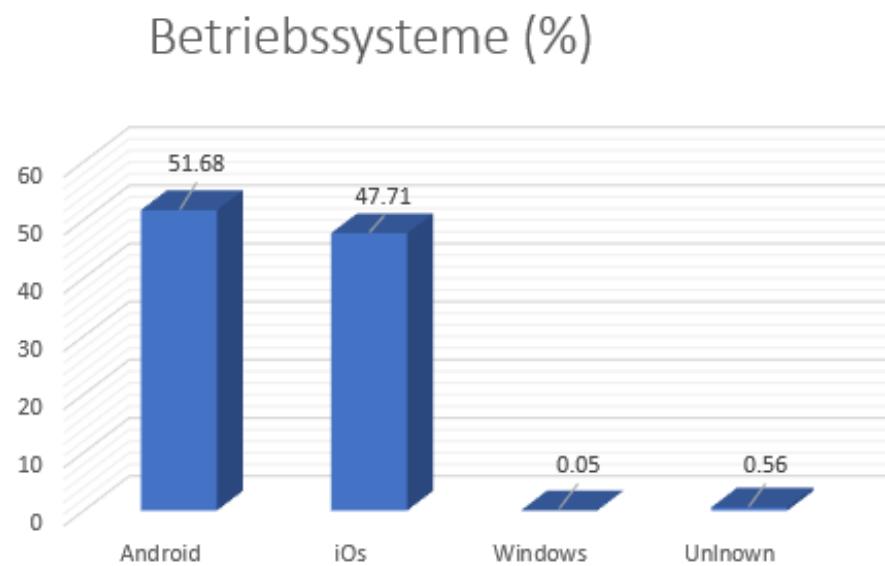


Abbildung 11: Verteilung der Betriebssysteme

Hauptsächlich sind die beiden Betriebssysteme Android und IOS mit zusammen über 99 Prozent Marktanteil im Jahr 2019 vertreten.

#### 4. ALLGEMEINE MOBILGERÄTE-ANALYSE

IOS wird nur von IPHones verwendet, aber Android-Geräte lassen sich noch weiter nach Hersteller unterteilen.

In der Abbildung 12 ist die Verteilung nach Hersteller im Jahr 2019 aufgelistet.

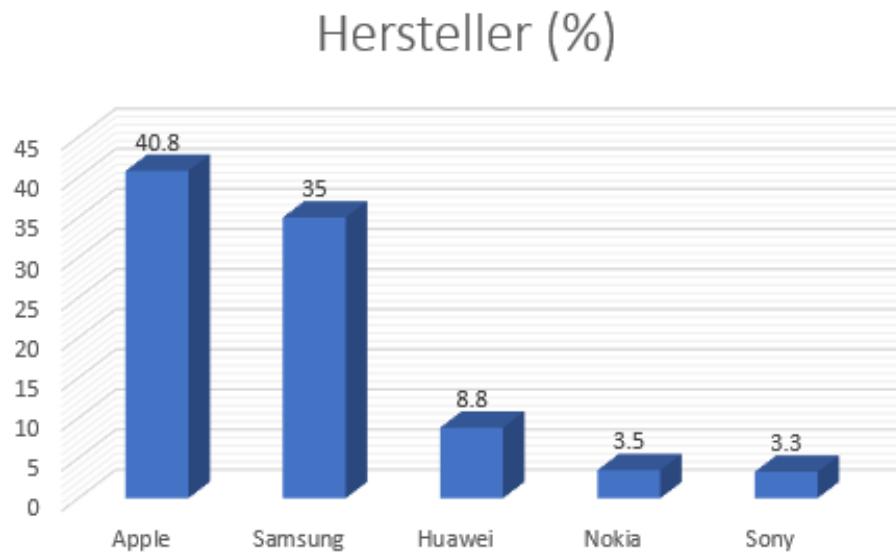


Abbildung 12: Verteilung der Hersteller

Die meistverwendeten Smartphones sind entweder IPHones oder Samsung-Geräte. Huawei hat mit 8.8 Prozent einen eher kleinen Marktanteil und wird in naher Zukunft ein eigenes Betriebssystem für seine Geräte anbieten, da der Hersteller aufgrund diverser Privatsphäreanschuldigungen in der USA auf schwarzen Liste steht.

## 4. ALLGEMEINE MOBILGERÄTE-ANALYSE

### **4.2 Auswahl der Mobilgeräte**

Basierend auf den gewonnenen Erkenntnissen konnten die für die Versuche benötigten Mobilgeräte ausgewählt werden.

In der Analyse der beiden Betriebssysteme IOS (Siehe Abschnitt 5) und Android (Siehe Abschnitt 6) hat sich gezeigt, dass IOS-Geräte ab iPhone 8 und Android-Geräte mit der Android-Version 9 oder neuer für die Versuche in Betracht gezogen werden. Nachfolgend sind die Geräte aufgelistet.

#### **IOS-Geräte**

- iPhone 8
- iPhone X
- iPhone XR
- iPhone XS
- iPhone 11
- iPhone 11 Pro
- iPhone SE

#### **Samsung-Geräte**

- Samsung Galaxy S8
- Samsung Galaxy S9
- Samsung Galaxy S9+
- Samsung Galaxy S10
- Samsung Galaxy S10 light
- Samsung Galaxy S20
- Samsung Galaxy S20+
- Samsung Galaxy S20 ultra

#### **Google-Geräte**

- Pixel 3
- Pixel 3 XL
- Pixel 3a
- Pixel 3a XL
- Pixel 4
- Pixel 4 XL

## 4. ALLGEMEINE MOBILGERÄTE-ANALYSE

### Weitere Hersteller

- OnePlus 7T
- Oneplus 7T Pro
- Oneplus 8
- Oneplus 8 Pro
- Huawei P20
- Huawei P30
- Fairphone 3
- Fairphone 3+
- Weitere, falls Android-Version 9 oder neuer

Die Auflistung zeigt nur die Geräte, welche für die Versuche in Betracht gezogen werden. Die tatsächlich verwendeten Geräte werden im Kapitel II: Versuche aufgelistet.

## 5 IOS-Analyse

Apple verwendet MAC-Adress-Randomisierung seit der IOS Version 8, welche am 17. September 2014 veröffentlicht wurde. Die Einführung von Randomisierung wurde nicht nur als Absicht aufgefasst, die Privatsphäre der Apple-Benutzer besser zu schützen. Es wurde spekuliert, dass Apple die eigene Positionierungstechnologie iBeacon vorantreiben möchte, ein Service, der es erlaubt, IOS-Geräte zu lokalisieren und Pushnachrichten oder personalisierte Werbung auf diesen Geräten zu schalten.

In den folgenden Untersektionen werden die verschiedenen IOS-Versionen ab der Version 8 analysiert und aufgezeigt, wie sich die Verschleierung von MAC-Adressen weiterentwickelt hat. Es gilt noch zu erwähnen, dass Apple keine Dokumentationen für diese Features herausgibt und sämtliche Informationen über verschiedene Quellen zusammengesucht werden müssen. Eine Auflistung der Versionsgeschichte findet sich in der Tabelle 6.

### 5.1 Versionsgeschichte

Aktuellste Version	Erscheinungsdatum	Letzte Version für
3.1.3	02.02.2010	iPhone 2G
4.2.1	22.11.2010	iPhone 3G
5.1.1	07.05.2012	-
6.1.6	21.02.2014	iPhone 3GS
7.1.2	30.06.2014	iPhone 4
8.4.1	13.08.2015	-
9.3.5	25.08.2016	-
9.3.6	22.07.2019	iPhone 4S
10.3.3	19.07.2017	iPhone 5C
10.3.4	22.07.2019	iPhone 5
12.4.7	20.05.2020	iPhone 5S, iPhone 6
13.7	01.09.2020	-
14	16.09.2020	-

Tabelle 6: Versionsgeschichte des iOS für iPhones, alle grau eingefärbten Zeilen benutzen MAC Randomisierung

### 5.2 iOS 8

Das iOS 8 wurde am 2. Juni 2014 vorgestellt, am 14. September 2014 eingeführt und von den Geräten iPhone 4 bis 6 verwendet. Die Version 8 ist die erste, die bei iPhones bei Probe-Requests die MAC-Adresse randomisiert. Die offizielle Ankündigung auf der Apple-Webseite ist in der Abbildung 13 ersichtlich.

#### Randomized Wi-Fi addresses

When you're out running errands with your phone in your pocket, Wi-Fi hotspots have the ability to track your movements and behavior by scanning your Wi-Fi MAC address. A MAC address is a string of characters that uniquely identifies your device on a network. With iOS 8, we've introduced an innovative feature designed to protect your privacy by randomizing your device's MAC address when the device is passively scanning for Wi-Fi networks. Because your MAC address now changes when you're not connected to a network, it can't be used to persistently track you. This is in line with Apple's industry-leading effort to do away with persistent identifiers, and is unique to iOS devices.

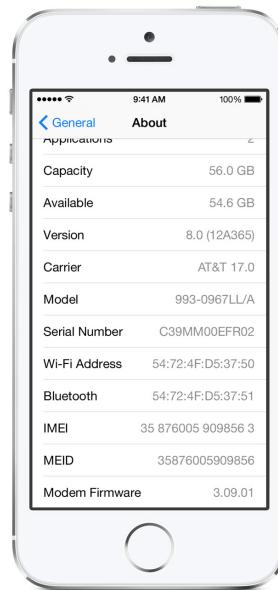


Abbildung 13: Offizielle Randomisierungs-Ankündigung von Apple

Allerdings wird die Randomisierung nur dann ausgeführt, wenn die folgenden drei Bedingungen erfüllt sind.

- Das Wifi muss aktiviert sein, das Gerät darf aber nicht mit einem Hotspot assoziiert sein.
- Das Smartphone muss sich im Sleep Mode befinden.
- Der Location Service muss in den privacy Settings explizit ausgeschaltet sein.

Somit wird die Randomisierung nur dann verwendet, wenn der Benutzer sich nicht bereits mit einem Netzwerk verbunden hat, das Gerät gerade nicht verwendet und explizit die Location Services ausgeschaltet hat. Viele Applikationen diese Location Services aber voraus und die meisten Nutzer machen sich nicht den Aufwand, jedes Mal die Einstellungen anzupassen, wenn sie eine Applikation verwenden wollen. Diese Faktoren haben dazu geführt, dass iOS-Geräte die MAC-Adresse häufig nicht zufallsgeneriert haben, obwohl Hard- und Software dies ermöglicht hätten.

### 5.3 iOS 9

Nachdem die Implementation der Randomisierung in der iOS Version 8 kritisiert wurde, wurde das Verfahren in der Version 9 überarbeitet und verbessert. In der Tabelle 7 ist der direkte Vergleich von iOS 8 zu iOS 9 ersichtlich.

	iOS 8	iOS 9
Unassociated PNO Scans	Yes	Yes
Unassociated ePNO Scans	Yes	Yes
Location Scans	No	Yes
Auto Join Scans	No	Yes

Tabelle 7: Abdeckung der verschiedenen Scans in iOS 8 & 9

Jeder dieser vier Scans wird in Form eines Probe-Requests versendet und der einzige Unterschied besteht darin, unter welchen Bedingungen diese Requests versendet werden. PNO und ePNO sind (enhanced) Preferred Network Offload Scans, die verwendet werden, um nach bekannten Netzwerken zu suchen, während sich das Gerät im Ruhemodus befindet. Location Scans werden von Applikationen verwendet, welche die Positionierung des Geräts erkennen möchten, wie beispielsweise eine Navigations-Applikation. Auto Join Scans sind Probe-Requests für bekannte Netzwerke wie Starbucks oder Eduroam.

### 5.4 iOS 10 - 13

Es wurden in der Recherche keine Informationen gefunden, ob und wie die MAC-Address- Randomisierung weiterentwickelt wurde. In der offiziellen Apple Platform Security Dokumentation wird erwähnt, dass ab dem iPhone 7 in Probe-Requests zusätzlich zur Randomisierung der MAC-Adresse auch die Sequenznummer zufallsgeneriert wird. Es wird nicht angegeben, welche iOS Version das Feature implementiert, aber anhand der unterstützten Geräte (iPhone, iPad, MacBooks etc.) lässt sich schliessen, dass die Sequenzverschleierung frühestens ab iOS 11 eingeführt wurde.

### 5.5 iOS 14

iOS 14 wurde am 22. Juni 2020 vorgestellt und am 16.09.2020 veröffentlicht. Mit der neuen Software-Version wurden auch diverse Änderungen vorgestellt, auf welche Art die MAC-Randomisierung vorgenommen wird und wie der User Einfluss auf die Privatssphäreinstellungen nehmen kann.

In der neuen Version verwendet ein iPhone für jedes Wi-Fi Netzwerk eine eigene randomisierte MAC-Adresse um sich mit dem Netzwerk zu verbinden. Zusätzlich wird die MAC alle 24 Stunden geändert, damit die eigentliche MAC-Adresse des iPhone dem Access Point niemals bekannt gegeben wird.

Die Verschleierung von MAC-Adressen ist in iOS 14 standardmäßig aktiviert und die User haben mehr Möglichkeiten, die Privatssphäreinstellungen ihren eigenen Bedürfnissen anzupassen.

Allerdings gibt es auch Kritik an den neuen Praktiken von Apple. Wenn ein Mobilgerät sich nur über komplett Randomisierte MAC-Adressen mit Netzwerken verbinden, können dadurch Probleme auftreten, da für viele Betreiber von Access Points die MAC-Adresse eine Form der Authentifizierung ist.

Weiterhin ist zu erwähnen, dass iOS 14 erst gerade auf den Markt gebracht wurde und in vergangenen Versionen mit minor Releases immer neue Funktionen und Verhalten hinzugefügt wurden. Somit ist es gut möglich, dass in der Version 14.1 die MAC-Address-Randomisierung bereits komplett anders gehandhabt wird.

### 6 Android-Analyse

Android implementierte erstmals mit Android 6 (Marshmallow) Support für Developer für die Verschleierung von MAC-Adressen in Probe-Requests.

In Versuchen im Paper "A Study of MAC Adress Randomization in Mobile Devices and when it Fails" von Jeremy Martin et. al. aus dem Jahr 2017 wird beschrieben, dass die meisten Geräte mit Android 6 die Randomisierung der MAC-Adresse auf Hardwareebene gar nicht unterstützen da die Chipsets nicht dazu in der Lage waren.

Erst ab Android 8 (Oreo) wurde die Randomisierung der MAC-Adressen bei Probe-Requests von Android Geräten offiziell als Funktion implementiert (zuvor war die Option nur für Developer verfügbar) und standardmäßig verwendet.

Die Versionsgeschichte des Android Betriebssystem ist in der Tabelle 8 ersichtlich.

#### 6.1 Versionsgeschichte

Name	Nummer	Erscheinung	Supported	API Lvl
Honeycomb	3.0 - 3.2.6	22.02.2011	No	11 - 13
Icecream Sandwich	4.0 - 4.0.4	18.10.2011	No	14 - 15
Jelly Bean	4.1 - 4.3.1	09.07.2012	No	16 - 18
KitKat	4.4 - 4.4.4	31.10.2013	No	19 - 20
Lollipop	5.0 - 5.1.1	12.11.2014	No	21 - 22
Marshmallow	6.0 - 6.0.1	05.10.2015	No	23
Nougat	7.0 - 7.1.2	22.08.2016	No	24 - 25
Oreo	8.0 - 8.1	28.08.2017	Yes	26 - 27
Pie	9	06.08.2018	Yes	28
Android 10	10	03.09.2019	Yes	29
Android 11	11	08.09.2020	Yes	30

Tabelle 8: Versionsgeschichte des Android, alle grau eingefärbten Zeilen benutzen MAC-Randomisierung. Supported Devices erhalten noch aktuelle Sicherheitsuptades

### 6.2 Android 8 - Oreo

Ab Android 8.0 verwenden Android-Geräte bei der Suche nach Wi-Fi Netzwerken zufällige MAC-Adressen in Probe-Requests. Für jeden Scan wird eine neue zufällige Adresse generiert und zusätzlich wird die Sequenznummer zufällig generiert.

### 6.3 Android 9 - Pie

Mit Android 9.0 wurde die Developer-Option hinzugefügt, für jede Verbindung mit einem Wi-Fi-Netzwerk eine zufällige MAC-Adresse zu verwenden.

### 6.4 Android 10

Die Verwendung von zufälligen MAC-Adressen für jedes Netzwerk wurde in Android 10 standardmäßig aktiviert.

### 6.5 Android 11

In Android 11 Beta 1 wurde die Option "Wi-Fi enhanced MAC Randomization" hinzugefügt. Wenn der Access Point dies erlaubt, bzw. wenn auf dem Access Point die MAC-Randomisierung aktiviert ist, dann wird jedes Mal, wenn sich das Gerät mit diesem Access Point verbindet, eine neue MAC-Adresse generiert.

## **Teil II**

# **Versuche**

## 7 Einleitung

Um das Verhalten von Mobilgeräten aufzuzeichnen, muss ein Versuchsumfeld geschaffen werden, welches erlaubt, die Geräte unter möglichst realitätsnahen Bedingungen zu messen. Weiterhin soll das Versuchsumfeld verhindern, dass Störfaktoren die Messergebnisse beeinflussen.

Die beiden Anforderungen Wirklichkeitstreue und Störfaktorfreiheit schliessen sich zum Teil gegenseitig aus. Zum einen kann in einem isolierten Messraum nicht das selbe Umfeld simuliert werden, welches ein Mobilgerät im täglichen Gebrauch antrifft. Zum anderen können Probe-Requests von anderen Geräten die Messergebnisse erheblich verfälschen, vor allem wenn diese Probe-Requests mit anonymisierten MAC-Adressen durchgeführt werden und das zu messende Gerät nicht von den störenden Geräten unterschieden werden kann.

Es wurde entschieden, die Messungen in einem Faraday-Käfig durchzuführen, damit Fremdgeräte nicht die Versuche beeinflussen. Das Institut für Kommunikationstechnik ICOM hat eine Antennenmesskammer für die Messungen von Antennen und Kommunikationssystemen. Diese Messkammer kann Signale von ausserhalb des Messraums genügend abschirmen, dass diese nicht die Elektronik im Messraum beeinflussen. Nach einer kurzen Recherche wurde an Marcel Kluser verwiesen, der den Gebrauch der Messkammer für die Dauer der Experimente bewilligt hat.

### 7.1 Versuchsaufbau

Die Messkammer besteht aus einem Innenraum mit  $1.5m \times 2.5m$  Grundfläche, ist ca.  $2m$  hoch und in der Kammer mit Schaumstoffspitzen ausgekleidet, welche die Signalausbreitung aus der Kammer eindämmen. Die Aussenhülle besteht aus Metall und kann durch eine Türe den Innenraum komplett von der Aussenwelt abschotten.

In der Messkammer sind jeweils folgende Geräte im Betrieb:

- Das zu messende Mobilgerät
- Ein Laptop mit Wireshark im Monitormodus oder ein WLAN-Messgerät, angeschlossen an einen Laptop.
- Ein weiteres Mobilgerät für die Simulation eines WLAN-Access-Points.

Die Abbildung 14 zeigt den Versuchsaufbau mit den verwendeten Geräten.

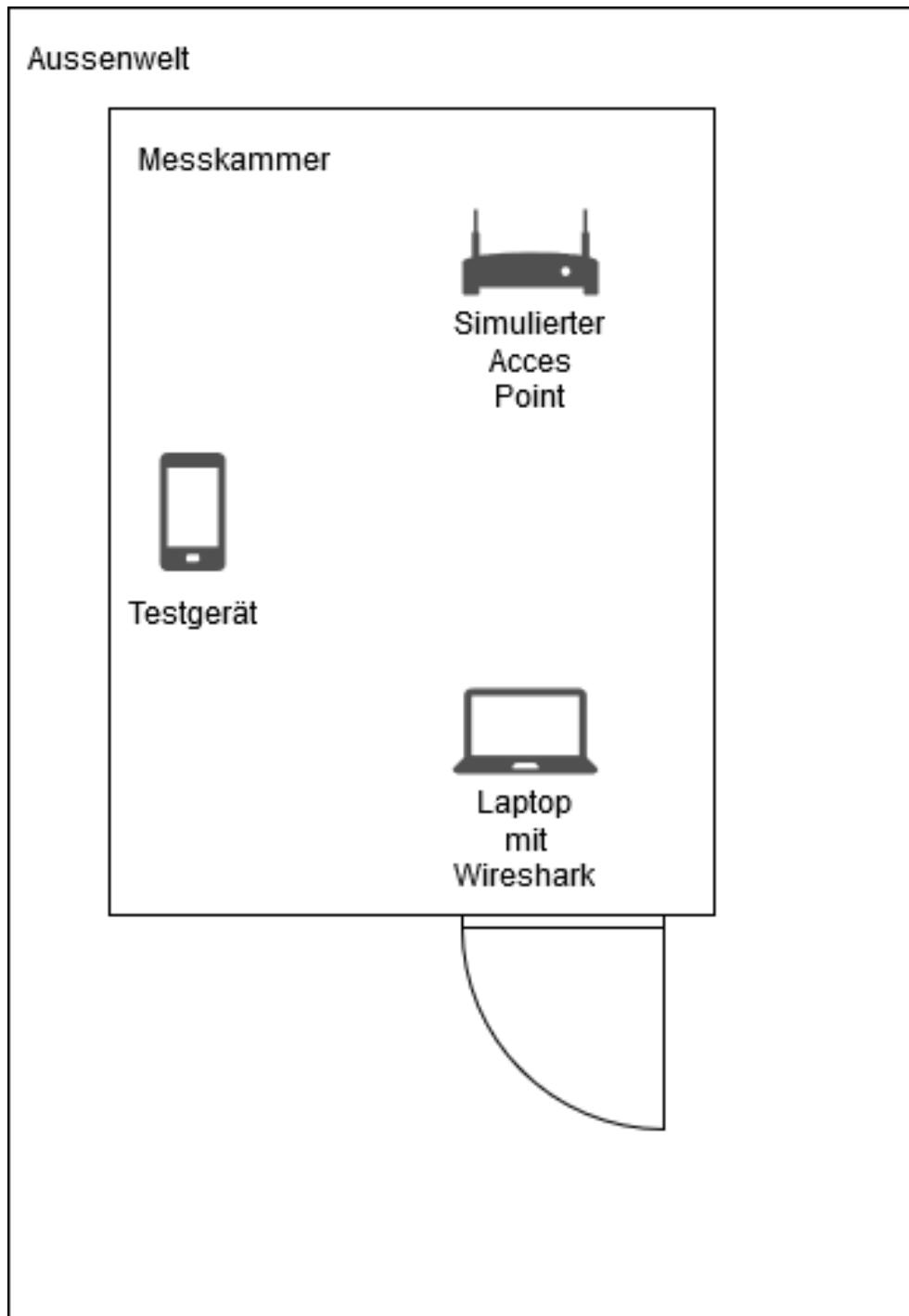


Abbildung 14: Messaufbau in der Antennenmesskammer.

### 7.2 Verwendete Tools

Für die Versuchsdurchführung wurde ein Macbook verwendet, welches über eine Netzwerkkarte verfügt, die das Messen von Probe-Requests ermöglicht. Auf dem Macbook ist Wireshark installiert, ein Netzwerk-Protokoll-Analysetool welches die Analyse von Computersignalen verschiedener Protokolle erlaubt. Der Monitormodus in Wireshark wird verwendet, um WLAN-Kommunikation im Empfangsbereich des Macbooks aufzuzeichnen. Solange das Macbook Wireshark im Monitormodus betreibt, werden keine eigenen Probe-Requests ausgesendet. Weiterhin wurde ein WaveXpert WLAN-Messgerät verwendet, welches über einen Laptop betrieben wird und auf mehreren Kanälen gleichzeitig Wireless-Kommunikation aufzeichnen kann. Bei der Verwendung des WaveXpert werden die Probe-Request des Betriebslaptops mit aufgezeichnet und müssen bei der Auswertung der Messergebnisse durch Filterregeln im Wireshark entfernt werden.

Mit Wireshark aufgezeichnete Messungen werden als .pcapng-Dateien gespeichert, welche mit Wireshark wieder geöffnet und analysiert werden können. Es ist möglich, die Messungen in diversen Formaten zu exportieren, wobei das JSON-Format die ursprünglichen Informationen der Messung am besten erhält und Informationen in einer hierarchischen Struktur speichert, die es erlaubt, die Messdaten einfach weiter zu verarbeiten.

Es wurde ein Python-Skript geschrieben, welches die Messdaten aus dem JSON-Format extrahiert und in ein Excel einträgt. Excel wurde für die Analyse ausgewählt, da in Excel sehr schnell einfache Auswertungen durchgeführt werden können. Für die Weiterverarbeitung in einem Prototyp können die Daten in ein dafür geeignetes Format codiert werden indem das Skript angepasst wird.

Um einen Access Point zu simulieren, wird ein weiteres Mobilgerät verwendet, welches in den dafür vorgesehenen Messungen einen Hotspot generiert. Während den Messungen, die keinen Hotspot benötigen, befindet sich das Mobilgerät im Flugmodus, um selbst keine Probe-Requests auszusenden. Als zusätzliche Sicherheit wurde die Randomisierung von MAC-Adressen auf dem Mobilgerät ausgeschaltet, damit fälschlich ausgesendete Probes einfach durch Filter entfernt werden können.

Die zu testenden Mobilgeräte werden in den jeweiligen Unterabschnitten genauer beschrieben.

### 7.3 Durchzuführende Messungen

Um das Verhalten der Mobilgeräte in verschiedenen Situationen aufzuzeichnen, wurden diverse Tests spezifiziert.

Da die Mobilgeräte gemäss der Recherchen ein unterschiedliches Verhalten haben, je nachdem ob sie in Gebrauch oder im Ruhemodus sind, werden die Tests in beiden Zuständen durchgeführt.

Es soll gemessen werden, ob Geräte Probes aussenden, wenn die Wireless-Einstellung ausgeschaltet ist. Da es möglich ist, dass dieses Verhalten in den Einstellungen angepasst werden kann, müssen die Messungen jeweils mit den Einstellungen ein- und ausgeschaltet durchgeführt werden (falls vorhanden).

Falls die Verschleierung der MAC-Adressen eine Einstellung auf dem Gerät ist, soll jeweils für beide Einstellungen eine Messung durchgeführt werden.

Da in früheren Arbeiten die Anzahl bekannter Access Points auf dem Mobilgerät die Anzahl ausgesandter Probe-Request beeinflusst hat, sollen alle Messungen mit unterschiedlicher Anzahl bekannter SSID's durchgeführt werden.

Damit das Probing-Verhalten eines Mobilgeräts während dem Einschaltvorgang und beim Wechsel in den bzw. aus dem Flugmodus aufgezeichnet werden kann, werden Messungen durchgeführt, während das Mobilgerät respektive der Flugmodus abwechselungsweise ein- und ausgeschaltet wird.

Weiterhin soll untersucht werden, ob Mobilgeräte, die mit einem Access Point verbunden sind, weiterhin Probe-Requests aussenden und ob in diesen Frames die MAC-Adresse verschleiert ist.

Für den Betrieb im Ruhemodus, im aktiven Modus und mit verbundem WLAN ist jeweils zusätzlich eine Langzeitmessung von einer Stunde vorgenommen, um genügend Frames aufzuzeichnen, dass ein mögliches Pattern in der Randomisierung der MAC-Adresse oder der Sequenznummer erkannt werden kann. Die anderen Messungen sind auf ein Zeitfenster von zehn Minuten begrenzt, wobei bei den Messungen mit ausgeschaltetem WLAN der Versuch abgebrochen wird, falls nach einer Minute keine Probe-Requests aufgezeichnet werden.

Gesamthaft werden pro Mobilgerät neun Stunden lang Messungen durchgeführt, wobei zwei Stunden und 40 Minuten eingespart werden können, wenn mit ausgeschaltetem WLAN keine Probe-Requests ausgesendet werden.

Zusätzlich sollen Messungen mit mehreren Mobilgeräten im Empfangsbereich durchgeführt werden, um realistische Messdaten zu erhalten. Mit diesen Messungen kann ein Prototyp während der Entwicklung verifiziert werden.

#### 7.4 Erwartete Messergebnisse

In der Recherche hat sich herausgestellt, dass moderne Betriebssysteme von Mobilgeräten die MAC-Adresse für sämtliche Probe-Requests randomisieren sollten, solange sie nicht mit einem Access Point assoziiert sind. Weiterhin sollten die Sequenznummern auch zufällig für jede Gruppe von Probe-Requests neu gesetzt werden. Es wird davon ausgegangen, dass die herstellerspezifischen Felder in den Frames der Probes nicht mitgesendet werden und die IE-Felder zwischen den Probe- Requests von unterschiedlichen Geräten mehrheitlich identisch sind.

Die einzigen zwei Möglichkeiten, den Geräten einen Fingerprint zuzuweisen, sehen die Autoren in der Reihenfolge und Anzahl der IE-Felder oder über einen Timing-basierten Ansatz.

#### 7.5 Messplan

Die Tabellen 9 und 10 zeigen die Messplanung, welche für die Versuche in der Messkammer verwendet wurde.

Testbeschreibung	WiFi	Privacy-Settings	Zeit
Messung im Startup	Ein	Ein	10 min
Mit WLAN verbunden	Ein	Ein	10 min
Suche nach versteckter SSID	Ein	Ein	10 min
Messung im Flugmodus	Ein	Ein	10 min
Langzeitmessung Ruhemodus	Ein	Ein	60 min
Langzeitmessung Aktivmodus	Ein	Ein	60 min
Langzeitmessung Verbunden	Ein	Ein	60 min

Tabelle 9: Messplan: SSID-unabhängige Messungen

## 7. EINLEITUNG

---

<b>Testbeschreibung</b>	<b>WiFi</b>	<b>Privacy-Settings</b>	<b>Zeit</b>	<b>Bekannte SSID's</b>
Gerät im Ruhemodus	Ein	Ein	10 min	10
	Ein	Aus	10 min	
	Aus	Ein	10 min	
	Aus	Aus	10 min	
	Ein	Ein	10 min	
	Ein	Aus	10 min	
	Aus	Ein	10 min	
	Aus	Aus	10 min	
Gerät aktiv	Ein	Ein	10 min	5
	Ein	Aus	10 min	
	Aus	Ein	10 min	
	Aus	Aus	10 min	
	Ein	Ein	10 min	
	Ein	Aus	10 min	
	Aus	Ein	10 min	
	Aus	Aus	10 min	
Gerät im Ruhemodus	Ein	Ein	10 min	1
	Ein	Aus	10 min	
	Aus	Ein	10 min	
	Aus	Aus	10 min	
	Ein	Ein	10 min	
	Ein	Aus	10 min	
	Aus	Ein	10 min	
	Aus	Aus	10 min	
Gerät aktiv	Ein	Ein	10 min	0
	Ein	Aus	10 min	
	Aus	Ein	10 min	
	Aus	Aus	10 min	
	Ein	Ein	10 min	
	Ein	Aus	10 min	
	Aus	Ein	10 min	
	Aus	Aus	10 min	

Tabelle 10: Messplan: SSID-abhängige Messungen

### 7.6 Beschaffung der Geräte

Um an Mobilgeräte für die Messungen zu kommen, wurden verschiedene mögliche Quellen angefragt. Das Institut für Softwareentwicklung IFS unterhält mehrere Mobilgeräte für Test-, Entwicklungs- und Ausstellungszwecke, die für die Versuche ausgeliehen werden durften. Weiterhin wurde im Freundes- und Familienkreis nachgefragt, ob moderne Mobilgeräte mit den zu messenden Betriebssystemversionen verwendet werden und ob diese ausgeliehen werden dürfen. Es hat sich herausgestellt, dass iPhones im persönlichen Umfeld weniger in Gebrauch sind als Android-Geräte. Dies stellt allerdings kein grösseres Problem dar, da Android auf einem breiteren Spektrum von Hardware verwendet wird, die potentiell grösseren hardwarespezifische Abweichungen im Verhalten aufweisen können.

Welche Mobilgeräte tatsächlich für die Messungen verwendet wurde, wird in den jeweiligen Abschnitten beschrieben.

## 8 IOS-Messungen

Nachfolgend sind die Messungen auf Mobilgeräten mit iOS-Betriebssystem und daraus gewonnene Erkenntnisse beschrieben.

Gesamthaft wurden 38 Einzelmessungen in 13 Stunden und 20 Minuten durchgeführt.

### 8.1 Versuchsdurchführung

Die im Unterabschnitt 7.3 genannten Messungen wurden auf insgesamt drei iPhones durchgeführt, welche in der Tabelle 11 aufgeführt werden.

iPhone Typ	iOS-Version	Besitzer
iPhone 8	14.0.1	Pascale Meier
iPhone X	14.0.1	Raphael Jud
iPhone X	12.3.1 & 14.0.1	IFS

Tabelle 11: Gemessene iOS-Geräte

Die zwei Messungen mit dem iPhone X vom IFS und die Messung mit dem iPhone 8 konnten komplett durchgeführt werden. Das iPhone X von Raphael Jud stand uns nur für fünf Stunden zur Verfügung und wir mussten uns auf die wichtigsten Messungen beschränken. Dazu haben wir uns entschieden, die Langzeitmessungen zu bevorzugen, da in diesen Messungen das Verschleiern der MAC-Adressen besser nachvollzogen werden kann. Weiterhin haben wir uns entschieden, dass im Bedarfsfall die nicht durchgeführten Messungen mit dem iPhone X von Raphael Jud zu einem späteren Zeitpunkt nachgeholt werden müssen.

Zusätzlich ist uns aufgefallen, dass die Anzahl der bekannten SSID's in iPhones vom Gerät selbst verwaltet wird und wir keinen Zugriff auf die bekannten Netzwerke haben und diese nicht beeinflussen können. Eine kurze Internetrecherche hat ergeben, dass ohne Applikationen von Drittherstellern diese Einstellung nicht vorgenommen werden kann. Es gibt weiterhin die Möglichkeit, die Netzwerkeinstellungen auf dem jeweiligen iPhone über die Geräteeinstellungen zu löschen. Beide Optionen konnten auf dem iPhone X von Raphael Jud und dem iPhone 8 nicht durchgeführt werden, da beides Leihgeräte sind und wir keine Geräteeinstellungen verändern wollen, die sich nicht rückgängig machen lassen. Somit mussten wir die Messungen mit unterschiedlichen bekannten SSID's auch aus der Versuchsdurchführung streichen.

Eine weitere Einstellung, die auf Android-Geräten vorgenommen werden kann, auf iPhones aber nicht existiert, sind die Privacy Settings. Auf Androidgeräten kann in den Lokalitätseinstellungen bestimmt werden, ob das Gerät Probe-Requests aussendet, obwohl das WLAN ausgeschaltet ist. iOS erlaubt nur das ein- und ausschalten der Standorteinstellung. Wir haben den Messplan dahingehend angepasst, dass die Messungen mit iOS jeweils mit dem Standort anstatt der Privatsphäreinstellung durchgeführt werden.

### 8.2 Ergebnisse

Mit den im vorherigen Unterabschnitt genannten Einschränkungen konnten die folgenden Messergebnisse produziert werden.

#### Probe-Requests

In der Tabelle 12 ist ersichtlich, wie viele Probe-Requests pro Messung aufgezeichnet wurden, wie viele davon in Bursts gruppiert waren, die minimale, durchschnittliche und maximale Burstgrösse sowie eine Schätzung der nicht aufgezeichneten Frames und die durchschnittliche Zwischenankunftszeit zwischen zwei Bursts.

iPhone	iOS Version	Anzahl Probes	Anzahl Bursts	min. Burstgrösse	avg. Burstgrösse	max. Burstgrösse	Verpasste Frames	Zwischenankunftszeit
iPhone 8	iOS 14.0.1	1219	327	1	4.372	26	1196	67.02 s
iPhone X - R.J.	iOS 14.0.1	1070	309	1	3.849	12	2382	58.27 s
iPhone X - IFS	iOS 12.3.1	762	193	1	4.051	12	1809	48.51 s
iPhone X - IFS	iOS 14.0.1	1784	400	1	4.609	18	2320	36.03 s
TOTAL		4835	1229	1	4.220	26	7707	52.46 s

Tabelle 12: Ergebnisse der iOS-Messungen

Die genauen Auswertungen der Messergebnisse finden sich im Anhang 24. Die Daten sind auf dem Repository im Ordner "Experimente" in den jeweiligen Unterordnern zu finden.

Nachfolgend sind in den Abbildungen die Graphische Auswertung nach Gesamtzahl (Abbildung 15) der Messwerte und die Messergebnisse nach Kategorie für die einzelnen iPhone-Geräte (Abbildungen 16 bis 19) dargestellt.

## Mobile Fingerprinting

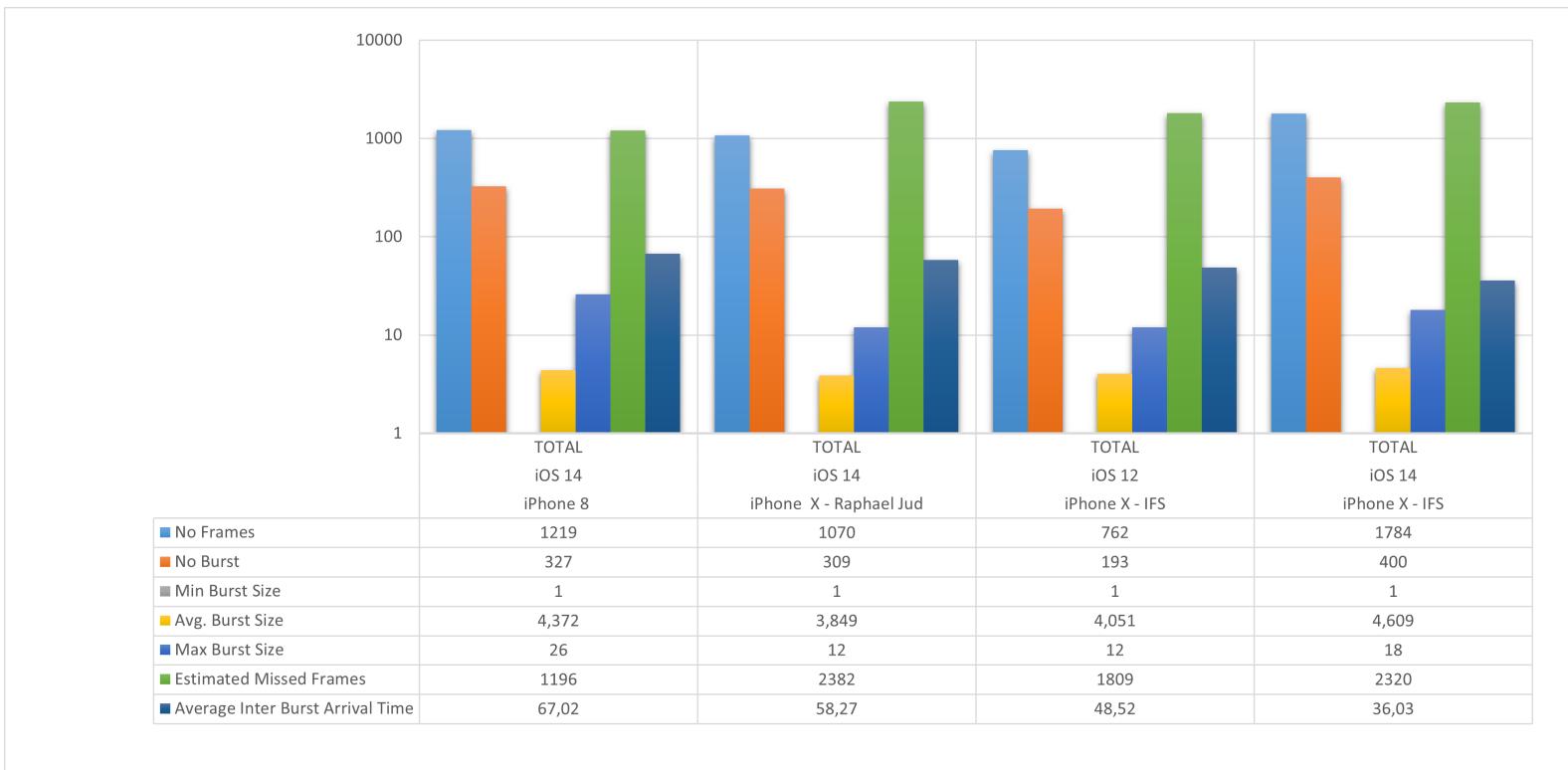


Abbildung 15: Gesamtergebnis der iOS-Messungen

## 8. IOS-MESSUNGEN

---

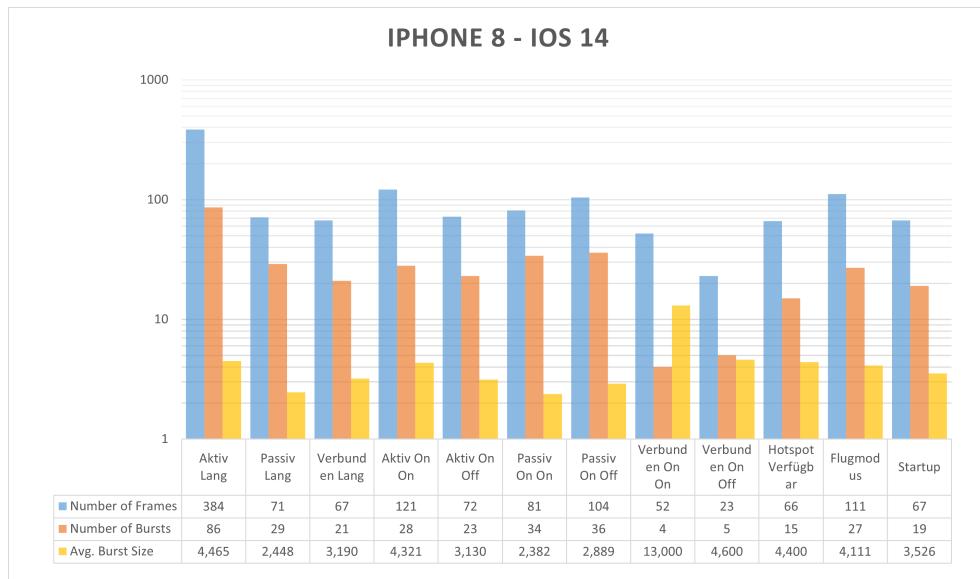


Abbildung 16: Messergebnisse iPhone 8 - iOS 14.0.1

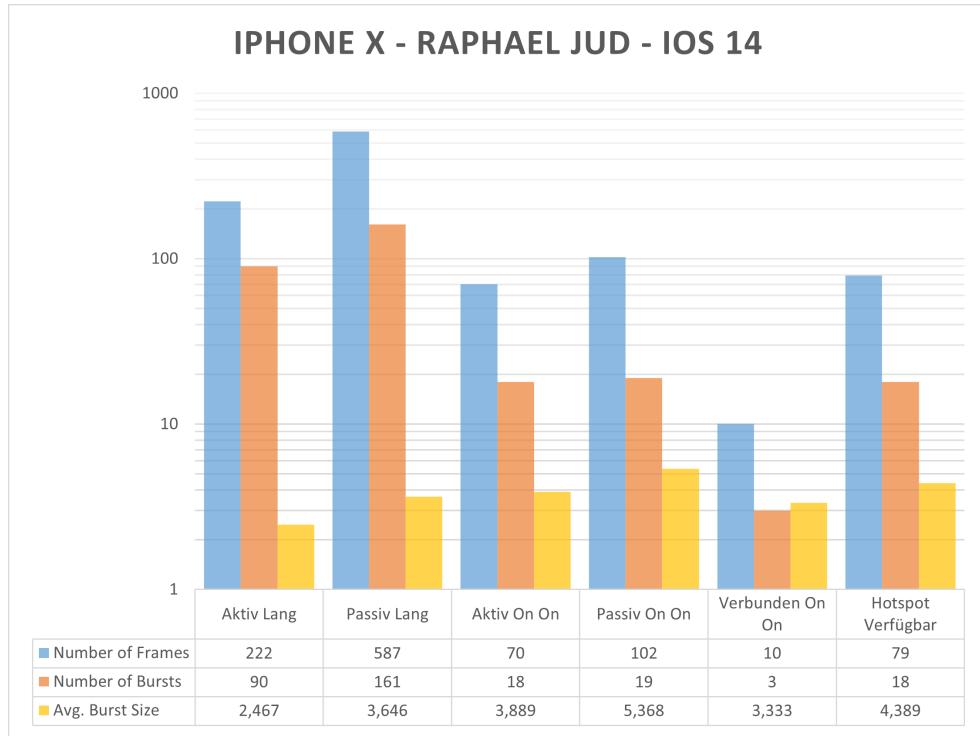


Abbildung 17: Messergebnisse iPhone X - Raphael Jud - iOS 14.0.1

## 8. IOS-MESSUNGEN

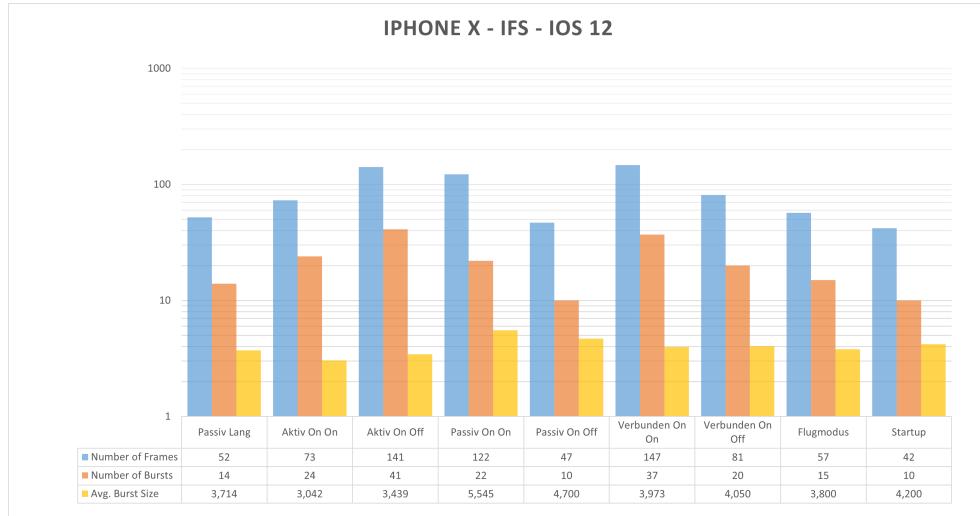


Abbildung 18: Messergebnisse iPhone X - IFS - iOS 12.3.1

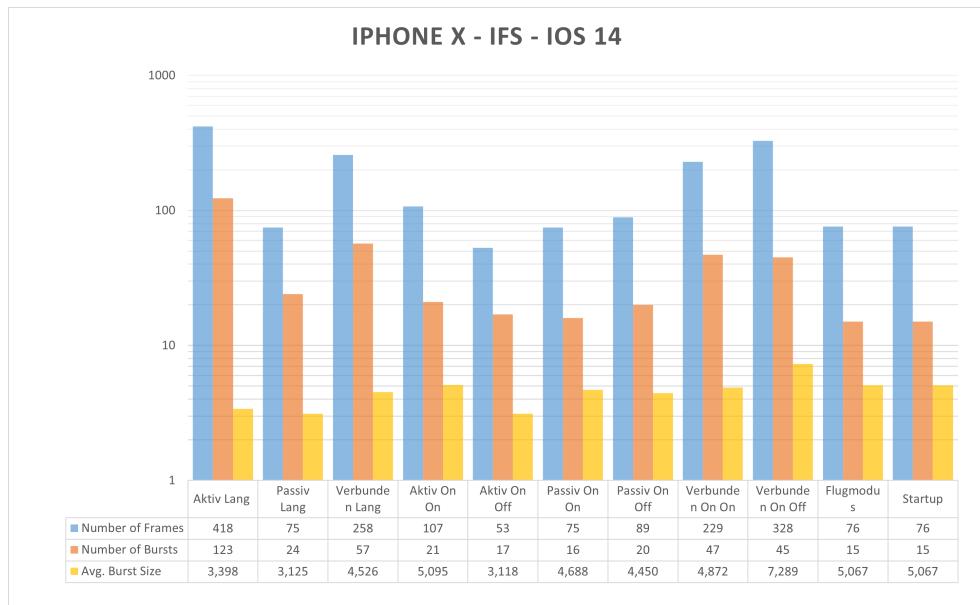


Abbildung 19: Messergebnisse iPhone X - IFS - iOS 14.0.1

### Erkenntnisse aus den iOS-Messungen

In total 4835 aufgezeichneten Probe-Requests sind im Schnitt jeweils 4.22 Frames pro Burst mit einer durchschnittlichen Zwischenankunftszeit von 52.46s gemessen worden. Die Bursts haben eine Grösse zwischen einem und 26 Frames.

Anhand der Sequenznummern innerhalb eines Bursts kann abgeschätzt werden, wie viele Frames im Burst nicht aufgezeichnet (verpasst) werden. In den iOS-Messungen wurden schätzungsweise 7707 Frames verpasst. Eine mögliche Erklärung für die hohe Anzahl verpasster Frames ist, dass iPhones zwischen einzelnen Frames den Kanal wechseln bzw. Probe-Requests auf wechselnden Kanälen aussenden, anstatt die Probes für einen Kanal zu gruppieren.

Bei genauerer Betrachtung der pcap oder JSON-Dateien zu den einzelnen Messungen fällt auf, dass iPhones IE-Felder gemäss den Tabellen 13, 14 und 15 beinhalten.

Tag-NR	0	1	50	3
Tag-Name	SSID	Supported Rates	Ex. Supp. Rates	DS Params Set
Geräte	alle	alle	alle	alle

Tabelle 13: IE-Felder, die in allen Messungen vorkommen

Tag-NR	45	127	107
Tag-Name	HT Capabilities	Extended Capabilities	Interworking
Phones	alle	alle	alle

Tabelle 14: IE-Felder der einzelnen iPhones

Es wird davon ausgegangen, dass die IE-Felder mit den Nummern 0, 1, 50, 3, 45 und 127 in allen iOS-Geräten verwendet werden. Der Interworking Tab mit der Tagnummer 107 wird in verschiedenen Messungen mitgesendet, es kann aber aus den Ergebnissen keine Regelmässigkeit herausgelesen werden, nach welchen Voraussetzungen der Tag verwendet wird.

	<b>Apple</b>	<b>Microsoft Corp.</b>	<b>Broadcom</b>
iPhone 8	x		
iPhone X - Raphael Jud - 14			
iPhone X - IFS - 12	x	x	x
iPhone X - IFS - 14		x	

Tabelle 15: Herstellerspezifische Felder (Vendor Specific - 221)

Pro Burst wird sowohl die MAC-Adresse als auch die Sequenznummer zufallsgeneriert. In ca 53% aller gemessenen iOS-MAC-Adressen ist das lokale Bit gesetzt, was bei zufallsgenerierten Adressen dem erwarteten Wert von 50% nahekommt. Alle iPhones verwenden, wenn sie mit einem Access-Point verbunden sind, ihre Korrekte MAC-Adresse. Weiterhin haben das iPhone 8 sowie das iPhone X mit der iOS-Version 12 in sämtlichen MAC-Adressen das lokale Bit gesetzt.

#### Vergleich mit den MAC-Adressen der Vorarbeit

In der Vorarbeit wurde eine Tabelle mit den häufigsten auftretenden OUI's von MAC-Adressen erstellt. Die in dieser Arbeit gemessenen Adressen wurde mit der Tabelle der Vorarbeit verglichen, um allenfalls Muster zu erkennen. Die OUIs der Vorarbeit sind in der Tabelle 16 abgebildet. CSV-Listen mit den MAC-Adressen aus den Messungen sind im Repository im Ordner "Experimente" zu finden.

00:B5:D0	04:79:70	04:F0:21
04:FE:A1	60:F1:89	7C:0B:C6
80:00:0B	84:98:66	9C:E0:63
B8:27:EB	E0:9D:31	EC:9B:F3
F0:42:1C		

Tabelle 16: Häufigste OUIs in der Vorarbeit

Es wurden keine Übereinstimmungen der MAC-Adressen aus den Messungen mit den OUIs der Vorarbeit gefunden.

## 9 Android-Messungen

Nachfolgend sind die Messungen auf Mobilgeräten mit Android-Betriebssystemen und daraus gewonnene Erkenntnisse beschrieben.

Gesamthaft wurden 70 Einzelmessungen in 28 Stunden und 40 Minuten durchgeführt.

### 9.1 Versuchsdurchführung

Die im Unterabschnitt 7.3 genannten Messungen wurden auf insgesamt neun Android-Geräten durchgeführt, welche in der Tabelle 17 aufgeführt werden.

Gerätetyp	Android-Version	Besitzer
Samsung A51	Android 10	Janik Schlatter
Fairphone 3+	Android 10	Mike Schmid
Samsung Galaxy S8	Android 9	IFS
Samsung Galaxy S8	Android 9	IFS
Samsung Galaxy S8	Android 9	Janik Schlatter
Samsung Galaxy S8	Android 9	Jenny Bösch
Samsung Galaxy S9	Android 10	IFS
Samsung Galaxy S20+	Android 10	IFS
Google Pixel 3	Android 11	IFS

Tabelle 17: Gemessene Android-Geräte

Die Messungen auf allen Geräten ausser den vier Galaxy S8 Geräten wurden komplett durchgeführt. Die S8 mit Android 9 verschleiern ihre MAC-Adresse in Probe-Requests nicht, weswegen auf diesen vier Geräten nur 20-minütige Aktivtests durchgeführt wurden.

Weiterhin stand für die zweite Woche in den Android-Messungen der WaveXpert, ein WLAN-Messgerät mit dem mehrere Kanäle gleichzeitig gemessen werden können, zur Verfügung. Mit dem WaveXpert wurden die Messungen mit dem Fairphone 3+ und Messungen mit mehreren parallel laufenden Geräten durchgeführt. Diese parallelen Messungen dienen der Verifikation eines Prototypen mit Daten, die ein realistischeres Umfeld darstellen, deren Parameter (Anzahl Geräte, Laufzeit, welche Geräte verbunden sind, etc.) aber beeinflusst werden können.

## 9.2 Ergebnisse

Die folgenden Messergebnisse konnten in den Messungen produziert werden.

### Probe-Requests

In der Tabelle 18 ist ersichtlich, wie viele Probe Requests pro Messung aufgezeichnet wurden, wie viele davon in Bursts gruppiert waren, die minimale, durchschnittliche und maximale Burstgrösse sowie eine Schätzung der nicht aufgezeichneten Frames und die durchschnittliche Zwischenankunftszeit zwischen zwei Bursts.

Gerät	Android Version	Anzahl Probe-Requests	Anzahl Bursts	min. Burstgrösse	avg. Burstgrösse	max. Burstgrösse	Verpasste Frames	Zwischenankunftszeit
Samsung A51	10	412	61	2	6,409	12	270	276,04 s
Samsung Galaxy S9	10	384	125	1	2,897	7	632	211,88 s
Samsung Galaxy S20+	10	489	303	1	1,889	2	161	118,00 s
Google Pixel 3	11	5048	960	1	4,415	15	2076	32,99 s
Samsung Galaxy S8	9	44	12	2	3,667	6	32	94,27 s
Samsung Galaxy S8	9	60	11	2	5,455	10	64	101,21 s
Samsung Galaxy S8	9	149	20	1	7,450	13	149	55,78 s
Samsung Galaxy S8	9	101	14	4	7,214	11	200	80,29 s
Fairphone 3+	10	2329	386	1	10,283	25	744	93,69 s
TOTAL		9016	1892	1	5,520	25	4328	118,24 s

Tabelle 18: Ergebnisse der Android-Messungen

Die genauen Auswertungen der Messergebnisse finden sich im Anhang 24. Die Daten sind auf dem Repository im Ordner "Experimente" in den jeweiligen Unterordnern zu finden.

Nachfolgend sind in den Abbildungen die Graphische Auswertung nach Gesamtzahl (Abbildung 20) der Messwerte und die Messergebnisse nach Kategorie für die einzelnen Android-Geräte (Abbildungen 21 bis 26) dargestellt.

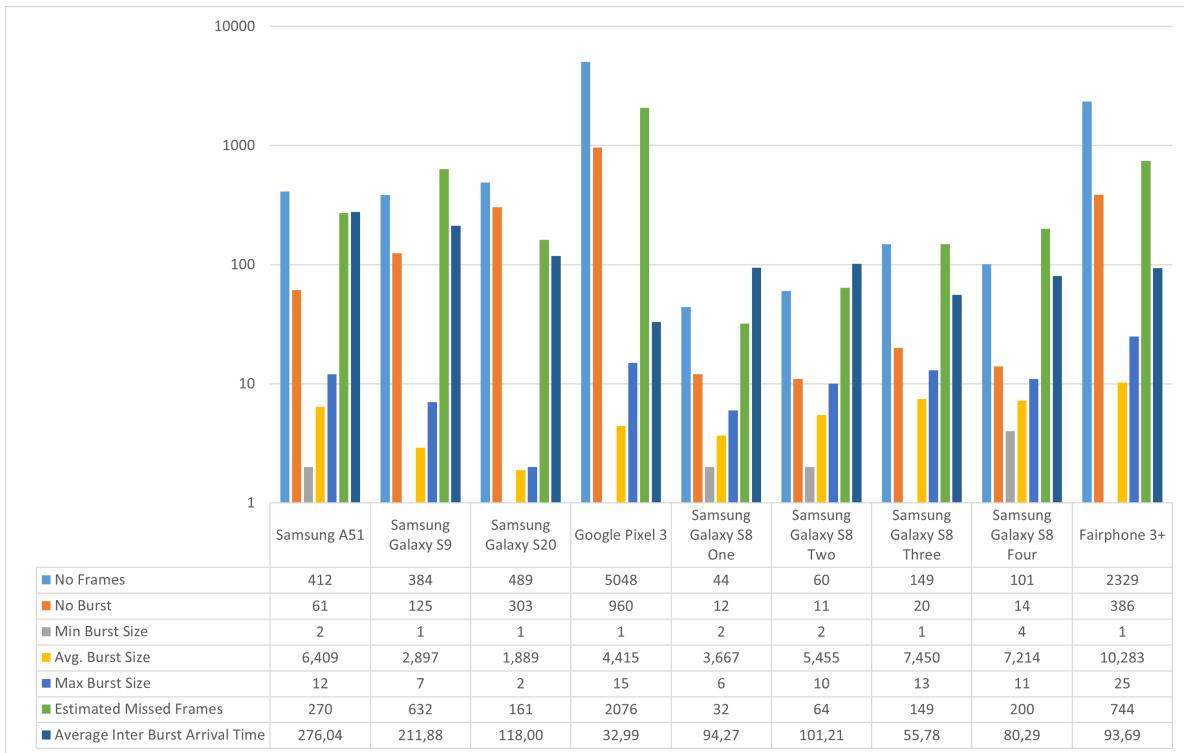


Abbildung 20: Gesamtergebnis der Android-Messungen

## 9. ANDROID-MESSUNGEN

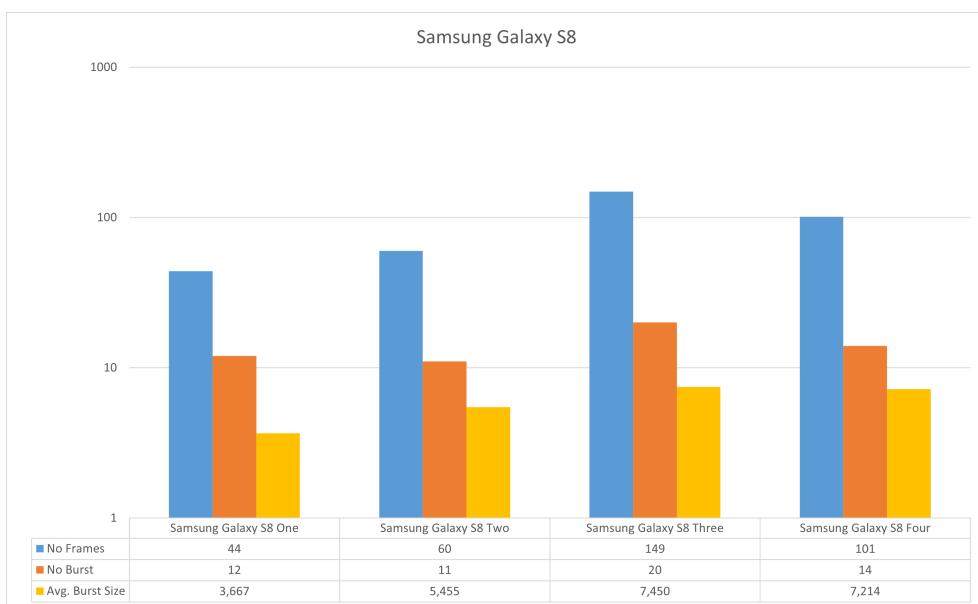


Abbildung 21: Messergebnisse der vier Samsung Galaxy S8-Geräte mit Android 9

In den Messungen der Samsung Galaxy S8 ist aufgefallen, dass diese bei Probe-Requests ihre echte Gerätemacadresse verwenden. Dieses Verhalten ist bei allen vier Geräten ersichtlich und lässt darauf schliessen, dass alle Samsung Galaxy Geräte der S8-Generation mit Android die MAC-Adresse nicht verschleiern. (Sofern nicht über den Developermodus die seit Android 6 verfügbare Option gesetzt wurde, die MAC-Adressen zu randomisieren)

## 9. ANDROID-MESSUNGEN

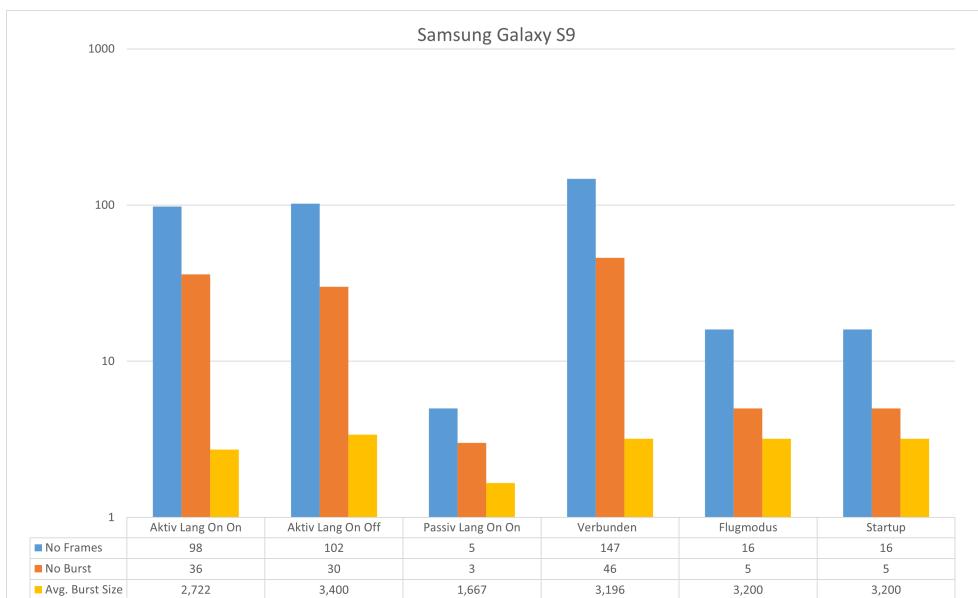


Abbildung 22: Messergebnisse Samsung Galaxy S9 - Android 10

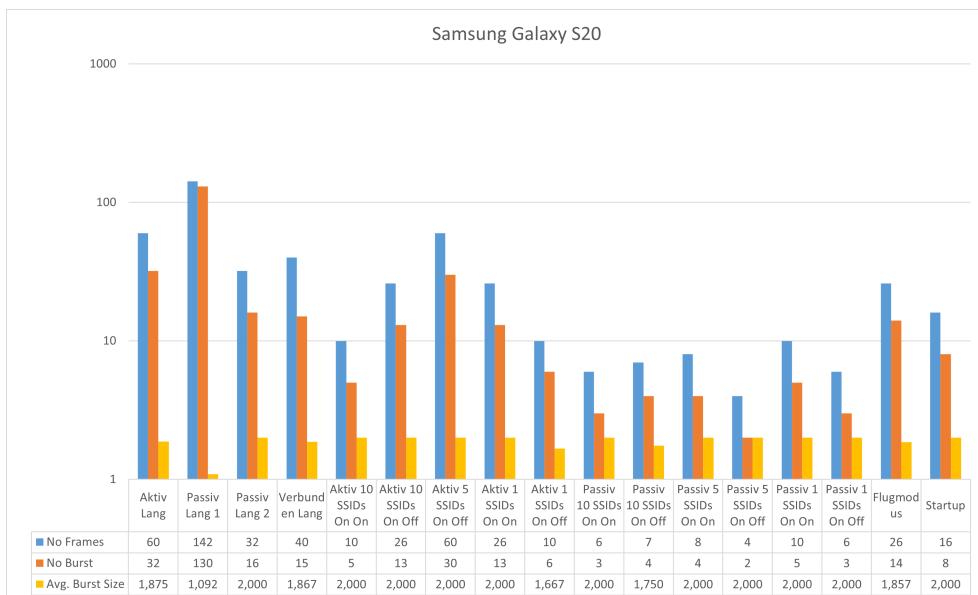


Abbildung 23: Messergebnisse Samsung Galaxy S20 - Android 10

## 9. ANDROID-MESSUNGEN

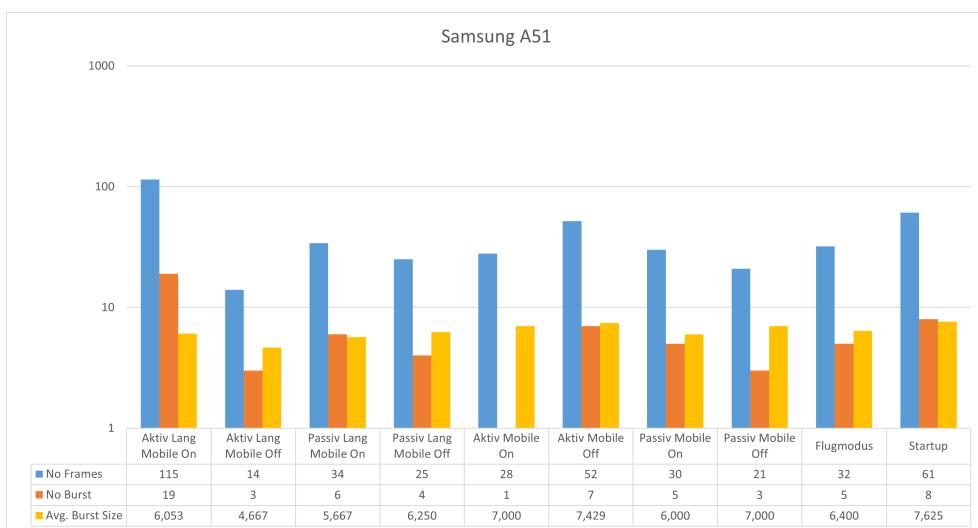


Abbildung 24: Messergebnisse Samsung A51 - Android 10

Besonders interessant bei der Messung des A51 ist die Tatsache, dass das Gerät für alle Messungen immer die MAC-Adresse "02:00:00:00:00:00" verwendet hat. Lediglich in den Startup- und Flugmodus-Messungen sind weitere Probe-Requests mit zufällig generierten MAC-Adressen versendet worden.

## 9. ANDROID-MESSUNGEN

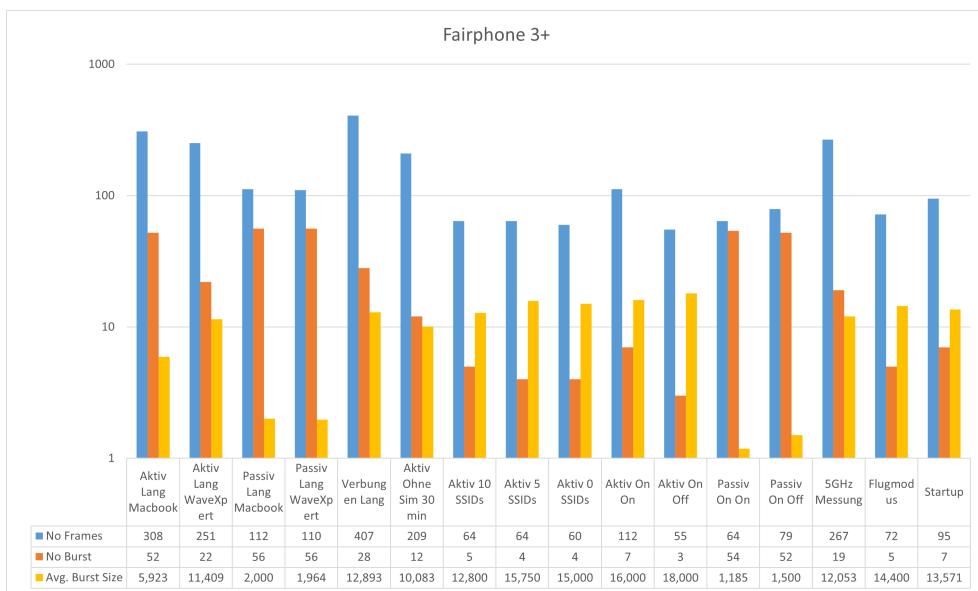


Abbildung 25: Messergebnisse Fairphone 3+ - Android 10

Das Fairphone 3+ fällt dadurch auf, dass für die passiven Probe-Requests eine MAC-Adresse mit Google-OUI und zufälliger NIC verwendet wird. Weiterhin ist die Zwischenankunftszeit bei passiven Probe-Request Bursts immer um die 63s.

Die Messungen mit dem Fairphone wurden alle mit dem WaveXpert durchgeführt und es ist in den aufgezeichneten Ergebnissen sehr gut ersichtlich, dass das Fairphone jeweils einen Burst Probes auf einem Kanal absendet, danach den Kanal wechselt und dort die nächste Gruppe Probe-Requests aussendet.

## 9. ANDROID-MESSUNGEN

---

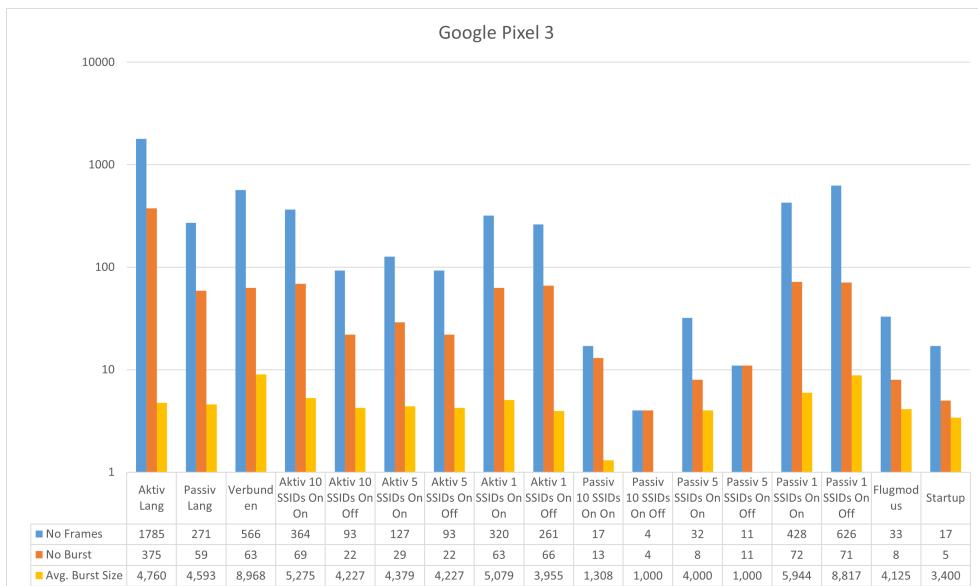


Abbildung 26: Messergebnisse Google Pixel 3 - Android 11

### Erkenntnisse aus den Android-Messungen

In total 9016 aufgezeichneten Probe-Requests sind im Schnitt jeweils 5.52 Frames pro Burst mit einer durchschnittlichen Zwischenankunftszeit von 118.24s gemessen worden. Die Bursts beinhalten zwischen einem und 25 Frames.

Mit 4328 geschätzten verpassten Frames haben Android-Geräte weniger verpasste Frames relativ zu den aufgezeichneten Frames, was in den Messungen mit dem WaveXpert darauf zurückgeführt werden kann, dass Android Geräte die Frames pro Kanal gruppiert aussenden. Anders ausgedrückt, ein Android-Gerät sendet zuerst einige Frames auf einem Kanal aus, wechselt dann den Kanal und sendet die nächste Gruppe. Der Vorgang wiederholt sich bei jedem Burst.

Bei genauerer Betrachtung der pcap oder JSON-Dateien zu den einzelnen Messungen fällt auf, dass Android-Geräte IE-Felder gem. den Tabellen [19](#) und [20](#) beinhalten.

Tag-NR Tag-Name	0 SSID	1 Supported Rates	50 Ex. Supp. Rates	3 DS Params Set
Geräte	alle	alle	alle	alle

Tabelle 19: IE-Felder, die in allen Messungen vorkommen

Die einzige Ausnahme davon ist das Fairphone 3 welches in den Passivmessungen den DS Parameter Set Tag nicht verwendet.

Der in den iOS-Messungen gängige Tag Interworking (107) wird bei Android-Geräten nicht verwendet. Dieser Tag kann somit in einem Fingerprinting für die Unterscheidung von Android- und iOS-Geräten verwendet werden.

Weiterhin verwendet das Google Pixel 3 die HT Capabilities nur im verbundenen Zustand und die Extended Capabilities gar nicht. Das Fairphone 3 verwendet die Extended Capabilities auch nie und die HT-Capabilities nur im Passiven Zustand.

## 9. ANDROID-MESSUNGEN

---

	<b>Microsoft Corp.</b>	<b>Broadcom</b>	<b>Epigram, Inc</b>	<b>Wi-Fi-Alliance</b>
A51	x			
Galaxy S8 One	x	x	x	
Galaxy S8 Two	x	x	x	
Galaxy S8 Three	x	x	x	
Galaxy S8 Four	x	x	x	
Galaxy S9	x	x	x	
Galaxy S20+	x	x	x	x
Google Pixel 3	x			x
Fairphone 3+	x			

Tabelle 20: Herstellerspezifische Felder (Vendor Specific - 221)

Auch der Vendor Specific Tag mit der Apple-OUI wird in Android-Geräten nie verwendet und kann für ein Fingerprinting verwendet werden.

In allen Android Geräten ist das Local Bit gesetzt, was in einem Fingerprinting für eine zusätzliche Unterscheidung von Android und iOS Geräten genutzt werden kann.

Das Samsung A51, das Fairphone 3+ und die Galaxy S8 haben über mehrere Bursts hinweg aufsteigende Sequenznummern.

### Vergleich mit den MAC-Adressen der Vorarbeit

In der Vorarbeit wurde eine Tabelle mit den häufigsten auftretenden OUI's von MAC-Adressen erstellt. Die in dieser Arbeit gemessenen Adressen wurde mit der Tabelle der Vorarbeit verglichen, um allenfalls Muster zu erkennen. Die OUIs der Vorarbeit sind in der Tabelle 16 abgebildet. CSV-Listen mit den MAC-Adressen aus den Messungen sind im Repository im Ordner "Experimente" zu finden.

Es wurden keine Übereinstimmungen der MAC-Adressen aus den Messungen mit den OUIs der Vorarbeit gefunden.

## 10 Schlussfolgerungen

Anhand der in den Abschnitten 8 und 9 genannten Resultate lassen sich nun mehrere Verfahren für eine Unterscheidung von Mobilgeräten herleiten.

Hier gilt zu erwähnen, dass die durchgeführten Messungen in dieser Bachelorarbeit statistisch nicht signifikant sind, da lediglich für das Samsung Galaxy S8 Tests auf mehr als zwei unterschiedlichen Geräten durchgeführt wurden. Dies bedeutet, dass eine Beobachtung, die spezifisch auf einem Gerät gemacht wurde, (z.B. Die hauptsächliche Verwendung der MAC-Adresse "02:00:00:00:00:00" auf dem Samsung A51) nicht für alle Geräte dieses Typs gelten müssen. Verfahren, die ein Mobilgerät aufgrund der spezifischen Eigenschaften dieses Geräts erkennen, können in der Praxis daran scheitern, dass das zu messende Gerät sich anders verhält als andere Geräte der selben Bauart.

Trotzdem wurden einige Beobachtungen gemacht, die es in Kombination mit den im Kapitel I recherchierten Geräteverhalten ermöglichen, Mobilgeräte unterschiedlicher Bauart voneinander zu unterscheiden.

### 10.1 Generelle Ansätze für ein Fingerprinting anhand ausgesendeter Probe-Requests

Allgemein kann ein Mobilgerät anhand einer oder mehrerer Eigenschaften von anderen Geräten eindeutig unterschieden werden. Kann diese Unterscheidung mehrmals nacheinander durchgeführt werden, sind diese Eigenschaften oder eine Kombination davon für dieses Gerät einzigartig und dienen somit als Fingerabdruck. Bevor die MAC-Adresse von Mobilgeräten zufallsgeneriert wurde, konnte diese Adresse als Fingerprint genutzt werden. Die Abbildung 27 zeigt dabei konzeptionell, wie solch ein Fingerprinting durchgeführt werden könnte.

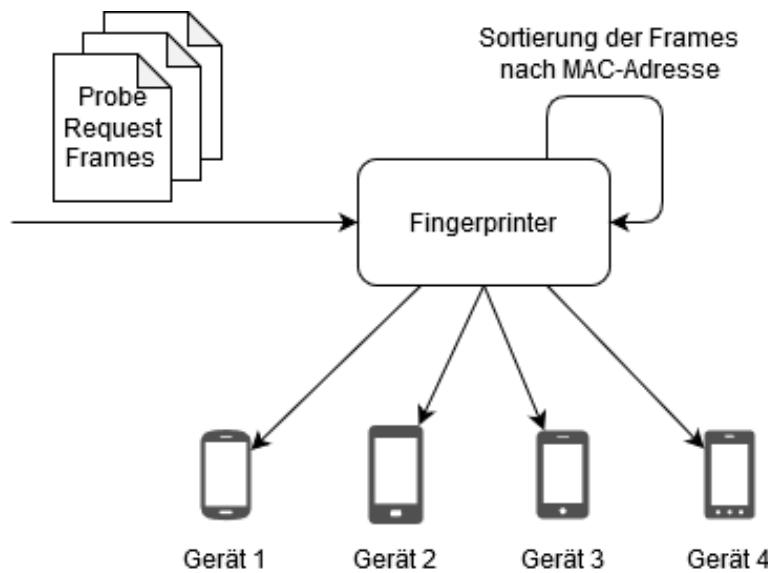


Abbildung 27: Fingerprinting vor iOS 8 / Android 9

Da die MAC-Adresse in modernen Geräten mit aktuellen Betriebssystemen aber zufallsgeneriert wird, muss dieses Verfahren angepasst werden. Ziel ist, Eigenschaften zu finden, um Geräte daran zu unterscheiden. Dabei gibt es verschiedene Ansätze: Machine Learning-Verfahren versuchen, automatisiert aus Millionen von Probe-Requests diese Eigenschaften zu finden und Geräte dementsprechend zu unterscheiden.

Ein weiterer Ansatz ist es, automatisiert nach vorgegebenen Filterregeln Geräte anhand bekannter Eigenschaften zu sortieren. Diese Filterung kann in mehreren Teilschritten vorgenommen werden. Zuerst wird anhand einer Eigenschaft sämtliche ankommende oder verfügbare Probe-Requests in diverse Gruppen unterteilt. Jede dieser Gruppen kann danach mit weiteren Filtern in Untergruppen unterteilt werden. Sind alle Filtervorgänge abgeschlossen, sollten sämtliche Frames einer Gruppe zu einem eindeutigen Mobilgerät gehören. Der Ansatz ist in der Abbildung 28 konzeptionell dargestellt.

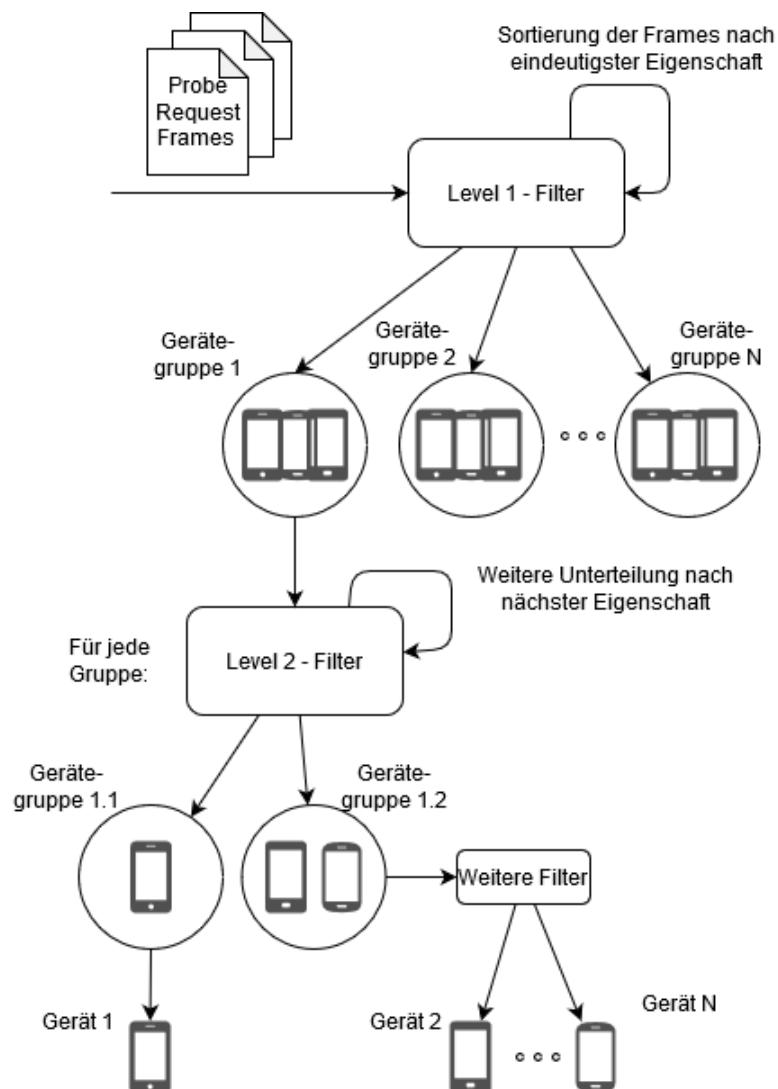


Abbildung 28: Filterbasiertes, hierarchische Geräteunterscheidung

## 10. SCHLUSSFOLGERUNGEN

Ein Vorteil eines filterbasierten Ansatzes ist, dass neue Erkenntnisse oder Änderungen des Geräteverhaltens mit geringem Aufwand implementiert werden können, indem die Filterregeln dementsprechend angepasst werden. Weiterhin muss im Gegensatz zu einem Machine-Learning-Ansatz nicht für jede Änderung ein Training des Algorithmus durchgeführt werden (von selbstlernenden Algorithmen abgesehen). Nachteile sind der Aufwand, der benötigt wird, um an die neuen unterscheidbaren Eigenschaften zu kommen und dass ein Filter-Programm regelmäßig an die Eigenschaften angepasst werden muss. Auch hier können selbstlernende Algorithmen eingesetzt werden, um den Aufwand zu verringern.

## 10.2 Spezifische Ansätze für die Geräteunterscheidung

Im Verlaufe der Bachelorarbeit sind mehrere Ansätze entstanden, wie eine Unterscheidung von Mobilgeräten durchgeführt werden könnte. Nachfolgend sind diese Ansätze, deren Vorgehen und die Umsetzbarkeit beschrieben.

### Ansatz 1: Zeitbasierte Auswertung von Frames

Unabhängig von den Informationen, die in ausgesendeten Probe-Requests enthalten sind, kann allein aufgrund von zeitlichen Informationen gefiltert werden. Ein Mobilgerät sendet jeweils einen Burst von Probe-Requests mit der gleichen MAC-Adresse aus. Wird innerhalb der Zeit des Bursts ein Frame aufgezeichnet, welches eine andere MAC-Adresse hat, wird dieses von einem anderen Gerät ausgesendet.

**Vorgehen** Wenn ankommende Frames in Bursts gruppiert werden, können Frames, die im selben Zeitraum aufgezeichnet werden, davon unterschieden werden. Allein anhand dieser Information kann eine ungefähre Anzahl von Mobilgeräten im Empfangsbereich eines Access-Points evaluiert werden. Die Abbildung 29 beschreibt das Verfahren.

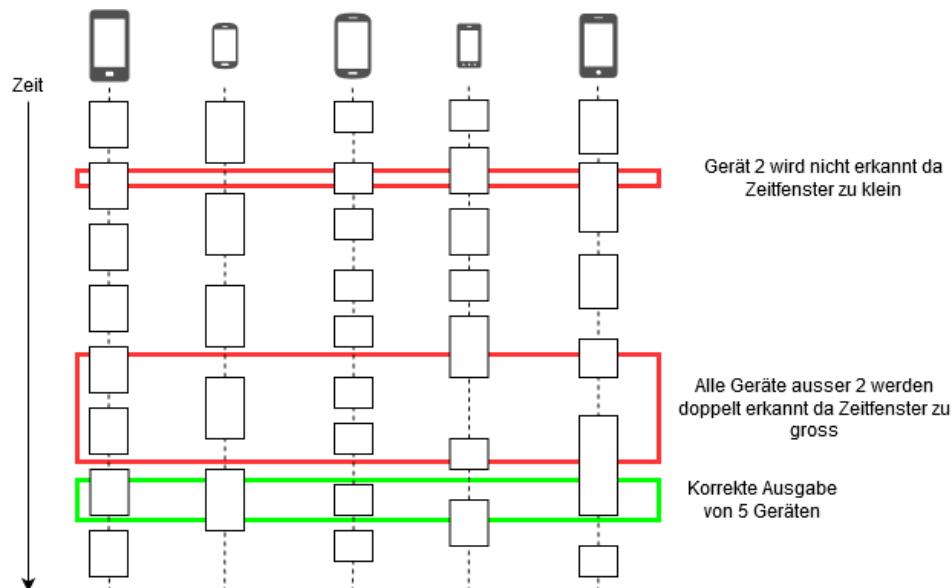


Abbildung 29: Zeitbasierte Filterung von ausgesendeten Probe-Requests

Die grösste Schwierigkeit ist dabei, dass das Zeitfenster derart gewählt wird, dass nicht zwei Bursts eines Geräts als zwei separate Geräte fehlinterpretiert werden, oder dass ein Gerät nicht erkannt wird, weil innerhalb des Zeitfensters gerade keine Probe-Requests gesendet wurden. Die Genauigkeit der Messung kann verbessert werden, indem mehrere Zeitfenster zu verschiedenen Zeitpunkten ausgewertet werden und ein Mittelwert aus den Resultaten gebildet wird. Zudem kann eine Auswertung der Burst-Zwischenankunftszeiten zur Laufzeit dazu verwendet werden, das Zeitfenster dynamisch an die Umgebung anzupassen.

**Umsetzbarkeit** Dieser Ansatz lässt sich mit wenig Aufwand umsetzen, vorausgesetzt, die Probe-Requests werden bereits durch eine Vorfilterung in Bursts aufgeteilt. Weiterhin ist dieser Ansatz gegenüber einer Veränderung von Geräteverhalten robust, da nur die Zwischenankunftszeiten der Bursts für die Unterteilung benötigt werden. Allerdings ist die Genauigkeit dieser Filterregel eher gering, da die Unterteilung in erster Linie von der Wahl der Dauer des Zeitfensters abhängt. Die Messergebnisse zeigen, dass ein Zeitbasierter Ansatz umsetzbar ist, da aber die verschiedenen getesteten Geräte keine einheitlichen Burstgrößen und -Zwischenankunftszeiten haben, muss die optimale Grösse des Zeitfensters experimentell ermittelt werden.

### Ansatz 2 - Auswertung der Information Element Felder

Dieser Ansatz beinhaltet zwei Vorgehen: Zum einen kann aufgrund vorhandener IE-Felder ein Mobilgerät von einem anderen unterschieden werden, wenn sie nicht dieselben Felder in ihren Probe-Requests verwenden. Zum anderen können Parameter in diesen Feldern dazu verwendet werden, Geräte zu unterscheiden.

Der Ansatz wurde auch schon in verwandten Arbeiten angewendet und die Messergebnisse lassen darauf schliessen, dass er auch weiterhin für ein Fingerprinting verwendet werden kann.

**Vorgehen** Eintreffende Probe-Request werden nach ihren IE-Feldern sortiert und in distinkte Gruppen unterteilt. Im Idealfall befindet sich pro Gruppe nur ein einzelnes Mobilgerät. Da gemäss unseren Messergebnissen aber mehrere Geräte die selben IE-Felder in ihren Probe-Requests verwenden und die Parameter in diesen Feldern auch identisch sein können, werden aber in jeder Gruppe mehrere Geräte auftreten. Ein weiteres Problem ist die Tatsache, dass ein Mobilgerät in mehreren Bursts nicht die selben IE-Felder verwenden muss. Somit ist es möglich, dass Bursts eines Geräts als mehrere Mobilgeräte erkannt werden.

**Umsetzbarkeit** Das Auslesen und Auswerten von Information Element Feldern aus Probe-Requests kann mit geringem Aufwand durchgeführt werden.

### Ansatz 3 - Filtern von nichtrandomisierten MAC-Adressen

Wird die gleiche MAC-Adresse über mehrere Bursts erkannt, kann davon ausgegangen werden, dass das Gerät die MAC-Adresse nicht randomisiert. In diesem Fall kann eine Filterung gemäss der Abbildung 27 durchgeführt und die MAC-Adresse des Geräts als Fingerabdruck verwendet werden.

**Vorgehen** Parallel zu anderen Filtern kann die wiederholte Verwendung von MAC-Adressen über mehrere Bursts erkannt werden. Weitere Bursts mit dieser Adresse können direkt mit dem Fingerabdruck versehen werden.

**Umsetzbarkeit** Wenn bereits eine Gruppierung von Probe-Requests nach Bursts vorgenommen wurde, benötigt die Umsetzung dieses Verfahrens nur geringen Aufwand.

### Ansatz 4 - Filterung zur Laufzeit anhand Burst-Zwischenankunftszeiten

Wenn für sämtliche Gerätetypen und Betriebssysteme die Zwischenankunftszeiten bekannt ist, und diese sich deterministisch verhalten, ist es möglich, eintreffende Bursts anhand dieser Zeiten einem bekannten Gerät zuzuordnen.

**Vorgehen** Angenommen, man hat ein Mobilgerät, welches immer 30 Sekunden nach dem letzten Probe-Request einen neuen Burst aussendet. Somit spielt es keine Rolle, ob die MAC-Adresse in jedem Burst neu zufallsgeneriert wird, da man nach einer Messung nur die 30s wartet und den neuen Burst für den Fingerprint aktualisiert.

Dieses Vorgehen hat mehrere Schwächen:

- Falls ein Mobilgerät die Bursts nicht immer in gleichbleibenden Abständen aussendet, lässt sich das Verfahren nicht anwenden.
- Wenn zwei Mobilgeräte dieselbe Zwischenankunftszeit haben, können diese nicht unterschieden werden.
- Wenn ein Mobilgerät eine Verzögerung von 30s und ein weiteres Mobilgerät eine Verzögerung von 15s haben, kann man diese Geräte alle 30 Sekunden nicht unterscheiden.
- Wenn zwei Probe-Requests von unterschiedlichen Mobilgeräten zur gleichen Zeit aufgezeichnet werden, lassen sich die Mobilgeräte nicht unterscheiden

**Umsetzbarkeit** In den Versuchen hat sich herausgestellt, dass die meisten Geräte die Bursts in zufälligen Zeitintervallen aussenden. Somit lässt sich dieses Verfahren nicht generell umsetzen. In Kombination mit anderen Verfahren kann dieser Ansatz aber dazu verwendet werden die Genauigkeit dieser Verfahren zu erhöhen. Dazu muss zur Laufzeit ausgewertet werden, ob die Zwischenankunftszeiten von Bursts eines Gerätes deterministisch sind, bevor künftige Probe-Requests danach sortiert werden.

### Ansatz 5 - Bitweise MAC-Header-Analyse

Im Paper "Noncooperative 802.11 MAC Layer Fingerprinting and Tracking of Mobile Devices" aus dem Jahr 2017 wird ein Verfahren beschrieben, mit dem ein Fingerprinting und Tracking bewerkstelligt werden könnte.

**Verfahren** Prinzipiell wird jedes Bit des Probe-Request MAC-Headers mittels der Berechnung der Entropie über sämtlichen vergangenen Messungen gewichtet. Neu auftretende Felder oder selten verwendete Felder haben eine hohe Gewichtung und werden für ein Fingerprinting direkt verwendet. Felder, die in einer Vielzahl von Probe-Requests vorkommen, werden für den Fingerprint ignoriert.

**Umsetzbarkeit** Im Vergleich mit den bisher genannten Verfahren ist dieser Ansatz der Aufwändigste und wird voraussichtlich im Rahmen dieser Arbeit nicht umgesetzt.

### Ansatz 6 - Filterung nach Sequenznummer

Für Geräte, deren Sequenznummer nicht pro Burst zufallsgeneriert wird, können Bursts daran erkannt werden, dass die Sequenznummer in neueren Frames inkrementiert wird.

**Verfahren** Wird erkannt, dass ein Mobilgerät in mehreren aufeinanderfolgenden Bursts die Sequenznummer nicht zufallsgeneriert, kann diese Erkenntnis dafür genutzt werden, eine Unterscheidung von Mobilgeräten genauer zu machen. Wenn Beispielsweise mit einem anderen Verfahren die Unterscheidung von zwei Bursts nicht möglich ist, aber erkannt wurde, dass Gerät A die Sequenznummer nicht randomisiert (unabhängig ob B ebenfalls die Sequenznummer nicht randomisiert), kann man denjenigen Burst auswählen, dessen Sequenznummer näher an der des vorhergehenden Bursts ist.

**Umsetzbarkeit** Die Versuche haben gezeigt, dass mehrere Geräte die Sequenznummern nicht für neue Bursts zufallsgenerieren. In Kombination

mit anderen Verfahren bietet diese Lösung eine Möglichkeit die Genauigkeit einer Unterscheidung zu erhöhen.

### **Ansatz 7 - Filterung nach Framelänge**

Wenn ein Gerät bei den Probe-Requests immer die gleiche Frame Länge hat, können die Probe-Requests dieses Gerätes daran erkannt und von Probes von anderen Geräten unterschieden werden.

**Verfahren** Dieses Verfahren ist vom Prinzip her ähnlich wie die Erkennung der IE Felder. Es ist allerdings einfacher, ein Frame anhand der Länge, die in Wireshark spezifisch ausgegeben wird, zu kategorisieren.

**Umsetzbarkeit** In den Messungen wurde erkannt, dass die Frame-Länge je nach dem messenden Gerät unterschiedlich sein kann, da abhängig von der Netzwerkkarte unterschiedliche Radiotap Header-Informationen aufgezeichnet werden. Deshalb wird dieses Verfahren nicht umgesetzt und stattdessen die Filterung nach IE-Feldern implementiert.

### **Ansatz 8 - Filtern nach Local-Bit**

Wenn ein Mobilgerät für sämtliche Frames in der zufallsgeneriereren MAC-Adresse das Local-Bit immer setzt, kann dieses Gerät von einem anderen Gerät, welches das Local-Bit auch zufällig setzt, unterschieden werden.

**Verfahren** Ähnlich wie in den Verfahren 3 und 6 kann zur Laufzeit evaluiert werden, ob in einer Gerätegruppe in den Frames der Bursts das Local Bit immer gesetzt ist. Künftig aufgezeichnete Frames können danach herausgefiltert werden.

**Umsetzbarkeit** Auch dieses Verfahren kann in Kombination mit anderen Verfahren dazu beitragen, die Genauigkeit einer Unterscheidung zu verbessern.

## **Teil III**

# **Prototyp**

## 11 Prototyp-Einführung

In den folgenden Abschnitten wird die Funktionsweise des Prototyps beschrieben. Dabei handelt es sich in erster Linie um ein Proof-of-Concept und nicht um eine vollumfängliche Softwarelösung. Der Prototyp entspricht dabei der im Abschnitt 10 genannten Lösung mit hierarchisch implementierten Filterregeln, die ankommende Probe-Requests in jeder Hierarchiestufe weiter unterteilen. Als Ergebnis entstehen distinkt voneinander unterscheidbare Gruppen von Probe-Requests, die potentiell zu einzelnen Geräten gehören.

Im Prototyp werden prinzipiell Prozessschritte gemäss folgender Auflistung vorgenommen.

- Preprocessing
- Vorfilterung
- Filterung

Bei der Programmierung der Filter wurde darauf geachtet, dass der Input und der Output in einem einheitlichen Format gehalten wird. Dieses Vorgehen erlaubt es dem Anwender, die einzelnen Verfahren in der Reihenfolge auszutauschen und die Ergebnisse untereinander zu vergleichen und somit eine optimale Reihenfolge festzulegen. Das gewählte Format ist hierbei ein Python-Dictionary (nachfolgend Dictionary genannt), welcher jeweils als Schlüssel-Werte-Paar eine Liste von Bursts beinhaltet.

### 11.1 Preprocessing

Im Preprocessing werden zuerst die einzelnen Probe-Requests aus den JSON-Dateien extrahiert und in einzelnen Frame-Klassen instanziert. Danach werden die Frames anhand der MAC-Adresse in die einzelnen Bursts klassifiziert, in Burst-Klassen eingeteilt und als Python-Liste (nachfolgend Liste genannt) gespeichert. Die Liste ist dabei aufsteigend nach der Ankunftszeit der Bursts sortiert. Ein Burst-Objekt hat jeweils die in den Frames verwendete MAC-Adresse, die Ankunftszeit des ersten und des letzten Frames, einen Boolean, ob das Lokale Bit gesetzt ist und eine Liste von Frames als Parameter. Auch die Liste der Frames ist aufsteigend nach der Ankunftszeit der Frames sortiert um in weiteren Filterschritten anhand der Ankunftszeit weiter filtern zu können.

### 11.2 Vorfilterung

Die Vorfilterung dient dem Zweck, das Datenset von Bursts zu säubern, die nicht weiter gefiltert werden müssen. Wird in einem Filterforgang festgestellt, dass ein Gerät seine wahre MAC-Adresse verwendet, kann diese Adresse in einem Set gespeichert werden und neue ankommende Probe-Requests können in der Vorfilterung aus dem Datenset entfernt werden, wenn sie eine der MAC-Adressen im Set verwenden. Es können zwei Fälle auftreten, in denen Geräte ihre wahre MAC-Adresse verwenden. Erstens, wenn das Gerät seine Adresse nicht zufallsgeneriert und zweitens, wenn das Gerät mit einem Acess-Point verbunden ist und die randomisierung der MAC-Adresse im verbundenen Zustand nicht eingestellt hat. Der erste Fall tritt bei älteren Geräten auf, deren Betriebssystem eine randomisierung der MAC nicht zulässt oder wenn auf dem Gerät die Einstellung, Adressen zufällig zu generieren, nicht eingestellt ist. Beide Fälle lassen sich daran erkennen, dass ein Gerät über mehrere Bursts hinweg die selbe MAC-Adresse verwendet. Es gibt dabei Geräte, die im verbundenen Zustand im SSID-Tag die MAC-Adresse des Netzwerks verwenden.

### 11.3 Filterung

Anhand den im Abschnitt 10 genannten Ansätze wurde versucht, zwei Filterregeln zu implementierten. Ziel dieser Filter ist es, Bursts in Gruppen zu unterteilen und diese Gruppen so lang weiter zu unterteilen bis potentiell pro Gruppe nur Bursts von einem Gerät existieren.

#### Filtern nach IE-Feldern

Zuerst werden die Bursts nach den verwendeten Information-Element-Feldern klassifiziert. In den Messungen wurde festgestellt, dass einzelne Geräte für Probe-Requests merhheitlich die selben IE-Felder verwenden. Die verwendeten IE-Felder werden innerhalb jeder der Listen, welche von der Vorfilterung erstellt wurde, von Burst zu Burst miteinander verglichen. Die Bursts werden dann anhand der IE-Felder in weitere Listen unterteilt und zurückgegeben.

## 11.4 Verwerfen des Ansatzes zur Zeitbasierten Auswertung

Es wurde versucht, das im Unterabschnitt 10.2 genannte Verfahren für die zeitbasierte Auswertung im Prototypen umzusetzen. Dabei fiel auf, dass die Dauer eines Bursts (Zeit zwischen dem ersten und letzten Frame im Burst) zu kurz ist, um das Verfahren wie geplant anzuwenden.

Selbst Bursts mit mehr als zehn Frames haben eine Dauer von weniger als einer Sekunde. Kombiniert mit der unvorhersehbaren Zwischenankunftszeit der Bursts ist es unmöglich, ein Zeitfenster zu finden, welches eine genaue Zählung der Mobilgeräte ermöglicht.

In einer zusätzlichen Auswertung der Messungen der vier Samsung Galaxy S8 kann aufgezeigt werden, dass sich selbst identische Gerätetypen mit der gleichen Betriebssystemversion sehr unterschiedlich verhalten. Das Verhalten wird in der Abbildung 30 dargestellt.

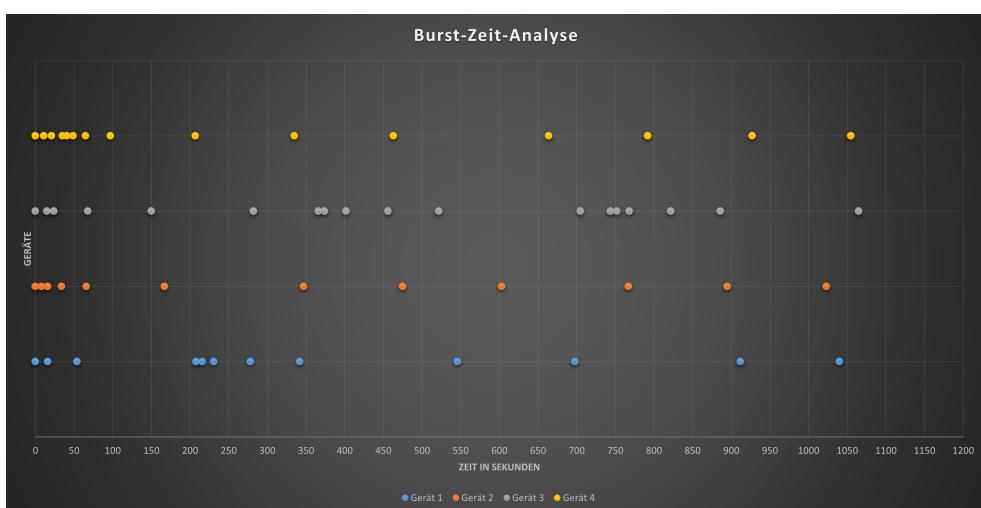


Abbildung 30: Auswertung der Bursts von vier Samsung Galaxy S8 Geräten.

Man kann sehr schön erkennen, dass es keine Regelmässigkeit zwischen allen vier Geräten gibt. Die Geräte zwei (Orange) und vier (Gelb) verhalten sich ähnlich, dadurch dass sie zu Beginn der Messung einige Probe-Requests in kurzer Abfolge aussenden und danach die Zwischenankunftszeiten grösser werden. Das Gerät drei (Grau) scheint die Zwischenankunftszeit der Probe-Requests jeweils mit jedem Burst zu erhöhen bis nach ca. 350 Sekunden wieder ein Reset durchgeführt wird. Das Gerät eins (Blau) verhält sich zu Beginn ähnlich wie das Gerät drei, wechselt aber zum Zeitpunkt 550 in einen regelmässigen Rhythmus.

Wenn man nun als Beispiel den Zeitpunkt 350 betrachtet, kann man sehen, dass kurz davor oder danach alle Mobilgeräte Probe-Requests ausgesendet haben. Mit einem angenommenen Zeitfenster von 50 Sekunden werden nun alle Bursts zwischen den Zeitpunkten 325 und 375 erfasst und es muss entschieden werden, wie viele Geräte sich im Datenset befinden. Insgesamt werden im Zeitfenster fünf Bursts erkannt. Jeweils ein Burst gehört zu den Geräten eins, zwei und vier und zwei Bursts gehören zum Gerät drei. Nun kann dieser Burst aber nicht eindeutig zum Gerät drei zugeordnet werden, da man nicht sagen kann, ob die zwei Bursts zu einem Gerät gehören oder von zwei separaten Geräten ausgesendet wurden ohne bereits zu wissen, welches Gerät welche Bursts ausgesendet hat.

### 11.5 Konzept: Säuberung des Datensets

Nachdem die Unterteilungen nach IE-Feldern und Ankunftszeiten vorgenommen wurde, kann eine weitere Säuberung vorgenommen werden, indem die Sequenznummern und die Local Bits angesehen werden. Die Säuberung des Datensetzes konnte im Prototypen aus Zeitgründen nicht umgesetzt werden.

#### Sequenznummern

Jede Gruppe besteht aus einer Liste von Bursts. Wenn man durch die einzelnen Bursts iteriert, kann festgestellt werden, ob die Sequenznummern aufsteigend sind, oder ob sie pro Burst zufällig generiert werden. Ist nun in der Mehrheit der Bursts die Sequenznummer aufsteigend und in einem einzelnen Burst nicht, ist die Chance gross, dass dieser Burst falsch klassifiziert wurde und der Burst kann aus der Liste entfernt werden.

#### Local Bit

In den Experimenten im Abschnitt 9 hat sich herausgestellt, dass sämtliche Android-Geräte das Local Bit in allen verwendeten MAC-Adressen setzen. Bei iOS-Geräten wird auch das Local Bit zufallsgeneriert was dazu führt, dass das Bit in ca. 50% aller Fälle gesetzt ist. Da nun iOS-Geräte daran erkannt werden können, dass sie in den IE-Feldern den Interworking- und Apple-Vendor-Tag verwenden, kann man in der Liste von Bursts eines als Android-Gerät erkannten Geräts diejenigen Bursts entfernen, bei denen das Local Bit nicht gesetzt ist.

## 12 Software-Entwicklung

In den folgenden Unterabschnitten wird die Architektur und Funktionsweise des erarbeiteten Prototyps erklärt. Der Prototyp ist in Python geschrieben und besteht aus fünf Klassen gemäss der Abbildung 31.

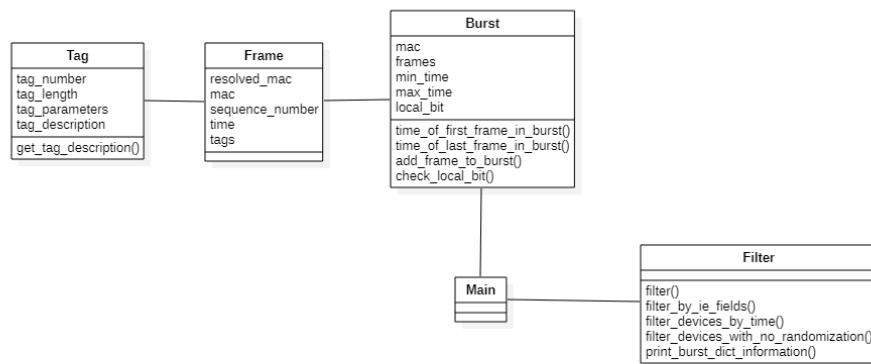


Abbildung 31: Klassendiagramm des Prototyps

### 12.1 Beschreibung der Klassen

#### Main-Klasse

Die Main-Klasse ist der Einstiegspunkt in das Programm. Darin werden Probe-Requests aus einer angegebenem JSON-Datei ausgelesen, decodiert und in Frame-Instanzen gespeichert. Diese Frame-Instanzen werden dann nach ihrer MAC-Adresse sortiert und in Burst-Instanzen gespeichert. Nachdem die Probe-Requests durch das Preprocessing in Bursts gruppiert wurden, wird eine Filterklasse instanziert und kann auf die Bursts angewandt werden.

### Burst-Klasse

Ein Burst besteht aus mehreren Probe-Request-Frames mit der gleichen MAC-Adresse. Die Burst Klasse beinhaltet einen bis mehrere Frames in einer Liste, welche aufsteigend nach der Ankunftszeit der Frames sortiert ist. Die Tabelle 21 beschreibt die Klassenvariablen.

Variable	Beschreibung
mac	Die MAC-Adresse der Frames im Burst.
frames	Eine Liste mit Frame-Objekten, die im Burst vorkommen.
min_time	Ankunftszeit des ersten Frame im Burst.
max_time	Ankunftszeit des letzten Frame im Burst.
local_bit	Indikator, ob das lokale Bit gesetzt ist.

Tabelle 21: Klassenvariablen der Burst-Klasse

Die Klasse beinhaltet vier Klassenmethoden gemäss der Tabelle 22.

Methode	Beschreibung
time_of_first_frame_in_burst	Evaluiert die Ankunftszeit des ersten Frame im Burst.
time_of_last_frame_in_burst	Evaluiert die Ankunftszeit des letzten Frame im Burst.
add_frame_to_burst	Fügt ein weiteres Frame zum Burst hinzu.
check_local_bit	Evaluiert anhand der MAC-Adresse, ob das local Bit gesetzt ist.

Tabelle 22: Klassenmethoden der Burst-Klasse

### Frame-Klasse

Die Frame-Klasse beinhaltet jeweils einen einzelnen Probe-Request-Frame, welcher in den Messungen aufgezeichnet wurde. Die Tabelle 23 beschreibt die Klassenvariablen. Die Frame-Klasse beinhaltet keine Klassenmethoden.

Variable	Beschreibung
resolved_mac	Die MAC-Adresse aufgelöst nach Hersteller, falls eine Auflösung durch Wireshark vorgenommen wurde.
mac	Die MAC-Adresse des Frame.
sequence_number	Die Sequenznummer des Frame.
time	Ankunftszeit des Frame.
tags	Liste mit Tag-Objekten, welche die Information-Element-Felder repräsentieren.

Tabelle 23: Klassenvariablen der Frame-Klasse

### Tag-Klasse

Tags werden verwendet, um die Information-Element-Felder zu repräsentieren. Die verwendeten Klassenvariablen sind in der Tabelle 24 dargestellt.

Variable	Beschreibung
tag_number	Nummer des IE-Felds gemäss IEEE 802.11
tag_length	Grösse des Information-Element-Felds
tag_parameters	Liste mit weiteren Informationen im IE-Feld.
tag_description	Bezeichnung des Information-Element-Felds.

Tabelle 24: Klassenvariablen der Tag-Klasse

Die Tag-Klasse beinhaltet die Klassenmethode "get\_tag\_description", welche aus einem Dictionary die Bezeichnung der Information-Element-Felder ausliest.

### Filter-Klasse

In der Filter-Klasse sind die einzelnen Filtermethoden umgesetzt, welche für die Klassifikation der Probe-Requests verwendet werden. Die Methoden nehmen als Input jeweils einen Dictionary mit Burst-Klassen entgegen und geben als Output ebenfalls einen Dictionary mit Burst-Klassen zurück. Die Eigenschaft, dass der Input und Output der Methoden in identischem Format gehalten wird, erlaubt eine beliebige Reihenfolge der Methodenaufrufe. Die einzelnen Filtermethoden sind in der Tabelle 25 beschrieben.

Methode	Beschreibung
filter	Führt alle Filtermethoden der Filterklasse aus.
filter_by_ie_fields	Klassifiziert Frames anhand der Information-Element-Felder.
filter_devices_by_time	Klassifiziert Frames nach ihrer Ankunftszeit.
filter_devices_with_no_randomization	Erkennt, ob in mehreren Bursts die selbe MAC-Adresse verwendet wurde und fasst diese in einer Liste zusammen.
print_burst_information	Methode für die Ausgabe der Filterresultate.

Tabelle 25: Klassenmethoden der Filter-Klasse

Die Filterung nach Ankunftszeit kann, wie im Unterabschnitt 11.4 beschrieben, nicht erfolgreich durchgeführt werden. Die Methode wurde der Vollständigkeit halber im Prototyp belassen, und kann verwendet werden, um aufzuzeigen, dass eine Filterung nach Ankunftszeit keine erfolgreiche Klassifikation durchführt.

## 13 Prototyp-Verifikation

Um die Funktionalität des Prototyps zu verifizieren, wurden insgesamt sechs separate Messungen durchgeführt. Diese Messungen beinhalten - abgesehen von einer Messung - alle mehrere Mobilgeräte. Die Messparameter und der Verwendungszweck sind in den nachfolgenden Unterabschnitten beschrieben.

### 13.1 Messungen für die Verifikation

#### Fairphone Gemischt Aktiv/Passiv

In der 30-Minütigen Messung wurde in einem fünf-Minuten-Intervall zwischen aktivem und passivem Verhalten auf dem Fairphone 3+ gewechselt. Mit diesen Messdaten wurde überprüft, ob der Prototyp in der Lage ist, Geräte zu erkennen, die während der Probe-Request-Aufzeichnung zwischen Aktiv/Passiv wechseln.

Die Prototyp-Verifikation hat gezeigt, dass der Prototyp Geräte, die unterschiedliche IE-Felder im aktiven oder passiven Modus verwenden, als zwei separate Geräte klassifiziert. Eine Auswertung der verwendeten IE-Felder hat gezeigt, dass außer dem Fairphone 3+ alle Geräte mehrheitlich die gleichen IE-Felder im aktiven wie auch im passiven Modus verwenden.

Die Abbildung 32 zeigt das Resultat des Prototypen für die Messung.

```
MAC | nof Bursts | Tag Numbers
3a:3d:93:88:a9:b7 | 19 | ['0', '1', '50', '45', '127', '221']
92:c1:f9:bd:25:b0 | 10 | ['0', '1', '50', '3', '45', '221']
Number of Devices: 2
```

Abbildung 32: Ausgabe des Prototyps für die Messung

### Fairphone iPhone Aktiv

In der 30-Minütigen Messung wurden Probe-Requests des iPhone X und des Fairphone 3+ im aktiven Modus aufgezeichnet. Mit den Messdaten wurde überprüft, ob der Prototyp zwischen einem Android- und einem iOS-Gerät unterscheiden kann.

Der Prototyp konnte in dieser Messung das Fairphone 3+ vom iPhone X unterscheiden, wie die Abbildung 33 zeigt.

```
MAC | nof Bursts | Tag Numbers
06:83:7e:4b:fd:ef | 13 | ['0', '1', '50', '3', '45', '221']
f6:65:54:75:a8:21 | 29 | ['0', '1', '50', '3', '45', '127', '107']
Number of Devices: 2
```

Abbildung 33: Ausgabe des Prototyp für die Messung

### Fairphone iPhone Passiv

In der 30-Minütigen Messung wurden Probe-Requests des iPhone X und des Fairphone 3+ im passiven Modus aufgezeichnet. Mit den Messdaten wurde überprüft, ob der Prototyp zwischen einem Android- und einem iOS-Gerät unterscheiden kann.

In der Passivmessung ist nur ein Burst des Fairphone aufgetreten und der Prototyp hat diese falsch als Störeinfluss klassifiziert und nicht beachtet. Man kann in der Abbildung 34 sehen, dass nur ein Gerät erkannt wird.

```
MAC | nof Bursts | Tag Numbers
07:c6:48:a7:fb:d8 | 42 | ['0', '1', '50', '3', '45', '127', '107']
Number of Devices: 1
```

Abbildung 34: Ausgabe des Prototyp für die Messung

Man kann im Prototyp einstellen, wieviele Bursts in einer Liste vorkommen müssen, damit die Liste als separates Gerät gespeichert wird. Man sieht in Abbildung 35 das Resultat, wenn ein Burst ausreicht, damit die Liste als Gerät akzeptiert wird.

```
MAC | nof Bursts | Tag Numbers
07:c6:48:a7:fb:d8 | 42 | ['0', '1', '50', '3', '45', '127', '107']
c6:06:45:eb:67:37 | 1 | ['0', '1', '50', '3', '45', '221']
Number of Devices: 2
```

Abbildung 35: Ausgabe des Prototyp für die Messung

### Pixel iPhone S20 Aktiv

In der 60-Minütigen Messung wurden Probe-Requests des Google Pixel 3, des iPhone X und des Samsung Galaxy S20+ im aktiven Modus aufgezeichnet. Mit den Messdaten wurde überprüft, ob der Prototyp zwischen drei Geräten unterscheiden kann.

Der Prototyp kann in der Messung erfolgreich die drei Geräte klassifizieren. Die Abbildung 36 zeigt die Ausgabe des Prototyps.

```
MAC | nof Bursts | Tag Numbers
c6:ca:bcc:cb:a5:87 | 25 | ['0', '1', '50', '3', '45', '127', '107']
Be:78:7a:21:cfc:41 | 12 | ['0', '1', '50', '3', '45', '127', '255', '255', '221', '221', '221', '221']
c6:ed:f0:f7:8f:cb | 10 | ['0', '1', '50', '3', '221']
Number of Devices: 3
```

Abbildung 36: Ausgabe des Prototyp für die Messung

### Pixel iPhone S20 Passiv

In der 60-Minütigen Messung wurden Probe-Requests des Google Pixel 3, des iPhone X und des Samsung Galaxy S20+ im passiven Modus aufgezeichnet. Mit den Messdaten wurde überprüft, ob der Prototyp zwischen drei Geräten unterscheiden kann.

Auch in der Passivmessung wurden alle drei Geräte korrekt klassifiziert, wie in der Abbildung 37 ersichtlich.

```
MAC | nof Bursts | Tag Numbers
ae:d1:ab:8a:21:3a | 8 | ['0', '1', '50', '3', '45', '127', '255', '255', '221', '221', '221', '221']
0e:b6:dd:e0:9e:8e | 12 | ['0', '1', '50', '3', '221']
bb:ed:68:d7:05:97 | 61 | ['0', '1', '50', '3', '45', '127', '107']
Number of Devices: 3
```

Abbildung 37: Ausgabe des Prototyp für die Messung

### Sechs Mobilgeräte Gemischtes Verhalten

In dieser Messung wurden 20 Minuten lang Probe-Requests von sechs verschiedenen Mobilgeräten aufgezeichnet. Dabei wechseln die Mobilgeräte zwischen Aktiv/Passiv, verbinden sich zwischendurch mit einem WLAN oder werden in den Flugmodus geschaltet. Die Messung soll in einem kleineren Rahmen das Verhalten von Passagieren in einem Zug simulieren und dient der Verifikation des Prototyps mit realistischen Messdaten.

Der Prototyp erkennt acht verschiedene Geräte in der Messung. Dies kann damit erklärt werden, dass ein Gerät je nachdem, ob es Aktiv oder Passiv ist, separat erkannt und klassifiziert wird, wie bereits mit der Messung des Fairphone 3+ gezeigt. In der Abbildung 38 ist die Ausgabe des Prototyp ersichtlich.

```

MAC | nof Bursts | Tag Numbers
da:de:1a:77:7a:e2 | 38 | ['0', '1', '50', '3', '221']
36:e3:f4:9f:fc:ef | 25 | ['0', '1', '50', '3', '45', '127', '255', '221', '221', '221']
1d:33:b9:dd:ec:77 | 17 | ['0', '1', '50', '3', '45', '127']
Google_d7:4e:4a | 39 | ['0', '1', '50', '3', '45', '221', '127']
Apple_57:14:58 | 8 | ['0', '1', '50', '3', '45', '127', '221', '221', '221']
2a:c6:38:1e:89:8a | 18 | ['0', '1', '50', '3', '45', '127', '221', '221', '221', '221']
HTC_1ed:1fe | 2 | ['0', '1', '50', '45', '191', '221', '3', '127']
de:0e:e5:6f:fd:01 | 3 | ['0', '1', '50', '3', '45', '127', '191', '221', '255', '221', '221']
Number of Devices: 8

```

Abbildung 38: Ausgabe des Prototyp für die Messung

Weiterhin ist es nicht ausgeschlossen, dass ein Gerät von ausserhalb der Messkammer aufgezeichnet wurde, da in den Versuchen kein HTC-Gerät verwendet wurde.

## **Teil IV**

## **Abschluss**

## 14 Weiterführende Arbeiten

### 14.1 Entwicklung einer Zähleinrichtung

Der in dieser Arbeit entwickelte Prototyp ist darauf ausgelegt, zuvor durchgeführte Wireshark-Messungen auszuwerten. Damit Messungen in der Praxis, beispielsweise in einem Zug oder Tram durchgeführt werden können, muss der Prototyp dahingehend erweitert werden, dass Probe-Requests kontinuierlich erfasst und ausgewertet werden können. Dazu wurden ein Architekturvorschlag erarbeitet, der in einer künftigen Arbeit als Anhaltspunkt dienen kann.

#### Architekturvorschlag

Eine Möglichkeit, wie die Architektur einer Zähleinrichtung aufgebaut werden könnte, wird in der Abbildung 39 dargestellt.

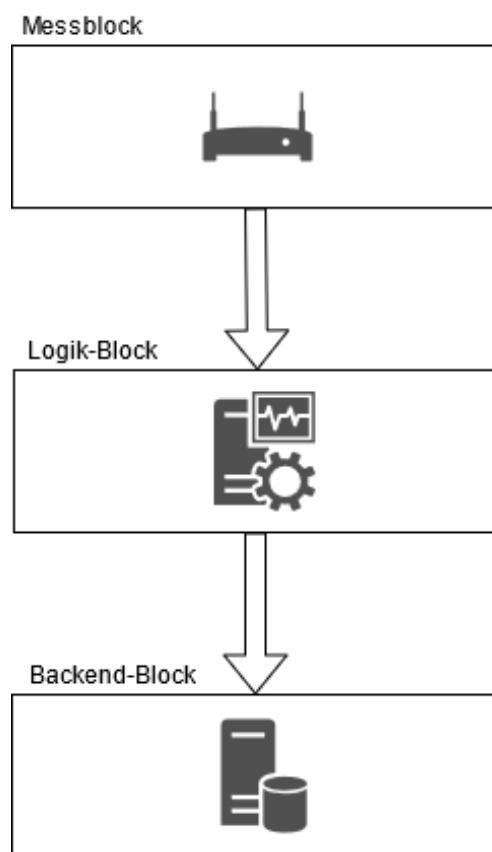


Abbildung 39: Architekturvorschlag für eine Zähleinrichtung

Im Messblock befinden sich ein oder mehrere Access-Points oder Messantennen, welche ankommende Probe-Requests erfassen und an den Logik-Block weiterleiten. Der Logik-Block ist für die Sortierung und Filterung der ankommenden Probe-Requests verantwortlich. Die Resultate werden dann an einen Backend-Block weitergeleitet, welcher für eine Persistierung der Ergebnisse und allfällige Auswertungen verantwortlich ist. Falls die Architektur mit einem IOT-Ansatz umgesetzt wird, kann durch eine Vorfilterung im Messblock die benötigte Bandbreite für die Übertragung reduziert werden. Weiterhin muss in solch einem Ansatz evaluiert werden, ob sich der Logik-Block vor Ort in Form eines Gateway befindet oder auf dem Server realisiert wird.

Ein weiterer möglicher Ansatz für eine Zähleinrichtung ist die Verwendung von Raspberry Pi als abgeschlossenes Zählsystem. Es ist möglich mit einem - oder mehreren - Raspberry Pi mit WLAN-Dongles ein Messsystem zu entwickeln, welches Probe-Requests erfassen und filtern kann. Eine Möglichkeit ist die Verwendung eines der Python-Module "Pyshark"<sup>1</sup> oder "Scapy"<sup>2</sup> welche beide in der Lage sind, WLAN-Daten zu erfassen und zu filtern.

### 14.2 Erweiterung der Filterung - Algorithmen

Eine zusätzliche Möglichkeit, den Prototypen zu verbessern, ist, die Algorithmen zu erweitern. Die Filterregeln können beispielsweise durch Machine-Learning-Algorithmen dahingehend angepasst werden, dass der Prototyp selbstständig und kontinuierlich die Klassifikation der Probe-Requests verbessert.

Dazu müssten weitere Messungen durchgeführt werden, in denen mehrere Geräte zur selben Zeit mit verschiedenen Einstellungen in Betrieb sind. Dieses Vorgehen würde einen Supervised-Learning-Ansatz ermöglichen. Alternativ kann auch mit Datensets aus der Praxis gearbeitet werden, welche mit einem Unsupervised-Learning-Ansatz einen Algorithmus trainieren.

Werden mehrere Messantennen für die Aufzeichnung von Probe-Requests verwendet, können zusätzliche Informationen bei den Messdaten hinzukommen. Beispielsweise ist die Information, welche Antenne den Probe-Request aufgezeichnet hat, in einer Filterung verwendbar. Allerdings muss darauf geachtet werden, dass Frames von einem Gerät, die von mehreren Antennen gleichzeitig aufgezeichnet werden, nicht als separate Frames interpretiert werden.

---

<sup>1</sup>Dokumentation: <https://kminewt.github.io/pyshark/>

<sup>2</sup>Dokumentation: <https://scapy.readthedocs.io/en/latest/introduction.html>

### 14.3 Erweiterung der Filterung - Messungen

In dieser Arbeit wurden zwölf verschiedene Geräte mit insgesamt dreizehn Betriebssystemversionen getestet. Die neueste Android-Version 11 konnte nur auf einem Gerät getestet werden, da die gerätespezifische Android-11-Version auf keinem der getesteten Geräte zur Verfügung stand. Weitere Messungen auf diesen Gerätetypen mit der Android Version 11 können neue Erkenntnisse generieren, mit denen der Prototyp weiter verbessert werden kann oder die eine Unterscheidung von Geräten komplett unmöglich machen.

Weiterhin können andere moderne Geräte getestet werden, um den Katalog von Geräteverhalten zu erweitern und den Prototypen zu verbessern.

### 14.4 Erweiterung der Aufzeichnungsmethoden

Im Rahmen dieser Arbeit wurde das Verhalten der Mobilgeräte beim Aus-senden von Probe-Requests erfasst. Künftige Arbeiten könnten neben Probe-Requests auch weitere Informationen von Mobilgeräten aufzeichnen. Eine mögliche Informationsquelle wären Bluetooth-Signale oder andere Frame-Typen aus der Sicherungsschicht.

Zudem gibt es im Forschungsfeld der Indoor-Lokalisierung den Ansatz, die Position eines Mobilgeräts anhand der Signalstärke zu triangulieren.

## 15 Abschliessendes Urteil

In dieser Arbeit wurden Messungen durchgeführt, um das Probing-Verhalten von modernen Mobilgeräten und aktuellen Betriebssystemen zu ermitteln. Anhand der Erkenntnisse aus diesen Messungen wurde ein Prototyp entwickelt, welcher aufgrund von aufgezeichneten Probe-Requests die Anzahl Mobilgeräte im Empfangsbereich berechnet. Dazu wurden Filterregeln in einem hierarchischen Filtervorgang implementiert. Der Prototyp wurde an dafür aufgezeichneten Messungen mit mehreren parallell laufenden Geräten validiert und konnte die Anzahl Geräte zum Teil korrekt ausgeben. Eine Validierung mit Messdaten aus der Praxis konnte nicht durchgeführt werden, da kein Datenset zur Verfügung stand, welches mehrere Mobilgeräte beinhaltet aber auch die Gesamtzahl von Mobilgeräten erfasst hat.

Die in den Messungen aufgezeichneten Probe-Requests beinhalten zu wenige voneinander unterscheidbare Parameter, dass damit Geräte mit einem Fingerabdruck versehen werden können. Fingerprinting von Mobilgeräten anhand von Probe-Requests wird mit neu erscheinenden Betriebssystemversionen immer schwieriger. Es ist schon nicht mehr möglich iPhones mit iOS-14 voneinander zu unterscheiden und neuere Android-Versionen wird die Randomisierung von Probe-Requests eher verbessern, als sie zu verschlechtern.

Somit sind Probe-Requests künftig nicht mehr für Fingerprinting nutzbar. Um bestehende Verfahren weiterhin nutzen zu können, müssen diese weiterentwickelt oder durch andere Verfahren ergänzt werden.

## **Teil V**

# **Projektmanagement**

## 16 Changelog

Date	Version	Changes	Autor
16.09.2020	0.1	Erstfassung	Mike Schmid
22.09.2020	0.2	Korrekturen	Mike Schmid
29.09.2020	0.3	Anpassungen Risikoanalyse	Mike Schmid
08.11.2020	0.4	Anpassung Iterationen und Meilensteine	Mike Schmid
15.12.2020	0.5	Korrektur Abgabedatum vom 08.11 auf den 15.01.2021	Mike Schmid
03.01.2021	1.0	Finalisieren der Projektdokumentation	Janik Schlatter & Mike Schmid

Tabelle 26: Changelog Projektmanagement

## 17 Einführung

### Zweck

Dieser Teil der Dokumentation beschreibt die Projektplanung und bietet eine Übersicht über den Verlauf der Bachelorarbeit "Mobile Fingerprinting". Im Detail wird auf die Planung, Organisation und den Projektaufbau und -ablauf eingegangen.

### Gültigkeit

Der Gültigkeitsbereich ist auf die Dauer der Durchführung der Bachelorarbeit "Mobile Fingerprinting" im Herbstsemester 2020 an der Fachhochschule OST beschränkt.

### Sprache

Die allgemeine Projektsprache (Dokumentation, Protokolle, etc.) wird in deutscher Schriftsprache verfasst. Der Code, das GitHub-Repository und die Versionskontrolle wird in englischer Sprache geschrieben.

### Referenzen

Alle Dokumente werden auf dem GitHub-Repository <sup>3</sup> abgelegt und verwaltet.

### Vorarbeit

Im Vorjahr wurde eine Studienarbeit mit dem Namen "Passenger Tracking" durchgeführt, die versucht hat, mittels Machine Learning ein Programm zu entwickeln, welches Passagiere im öffentlichen Verkehr anhand der von den Mobilgeräten ausgesendeten Probe-Requests erkennen kann. Dazu wurden den Studierenden 170 Millionen gesammelte WLAN-Daten vom Industriepartner für das Training des Algorithmus zur Verfügung gestellt. Der Prototyp konnte die Anzahl Geräte in einem Bus mittels eines Clustering Algorithmus mit durchschnittlich 94% Genauigkeit ermitteln.

Diese Bachelorarbeit soll sich im Gegensatz zur Vorarbeit direkt mit dem Verhalten der Mobilgeräte auseinandersetzen. Wie wird die MAC-Address-Randomisierung von verschiedenen Herstellern mit unterschiedlichen Betriebssystemversionen durchgeführt und wie kann man die Unterschiede dafür nutzen, ein Gerät eindeutig zu identifizieren und zu verfolgen?

---

<sup>3</sup> <https://github.com/EkoGuandor229/BA-MobileFingerprinting>

## 18 Projektübersicht

Auszug aus der Aufgabenstellung:

"Smartphones senden regelmäßig WLAN-Probe-Requests, um Access Points zu finden. In der Vergangenheit konnte durch das Auslesen der MAC-Adresse aus diesen Requests ein Gerät über längere Zeit verfolgt werden.

Seit einigen Jahren wird diese Verfolgung durch die Verwendung von anonymisierten MAC-Adressen erschwert. Weitere Datenfelder in diesen Requests können für das Tracken aber weiterhin von Nutzen sein.

In einer früheren Arbeit wurde anhand bestehender Daten eines Industriepartners ein Top-Down-Ansatz mittels Machine Learning erarbeitet, der aber nicht zu genügend genauen Resultaten führte."

### Zweck und Ziel

Das Verschleiern von MAC-Adressen durch Randomisierung ist seit einigen Jahren eine gängige Praxis, um die Verfolgung von Benutzern durch Organisationen oder Einzelpersonen zu verhindern. Dabei spielt es keine Rolle, ob die Verfolgung für statistische Zwecke, für personalisierte Werbung oder mit böswilligen Absichten geschieht.

Diese Arbeit befasst sich damit, wie diese Verschleierung von einzelnen Anbietern von Mobilgeräten durchgeführt wird und ob sich die Umsetzung in unterschiedlichen Betriebssystemversionen unterscheidet. Durch Experimente soll herausgefunden werden, wie die Verschleierung in modernen Geräten in verschiedenen Versionen der Betriebssysteme umgesetzt wird. Weiterhin soll ein Prototyp entwickelt werden, der die gewonnenen Erkenntnisse umsetzt und die Verfolgung eines Gerätes über mehrere WLAN-Access-Points ermöglicht. Die für das Training und die Verifizierung des Prototyps benötigten Daten sollen in den Experimenten erfasst und aufbereitet werden.

### **Lieferumfang**

- Auswertung, wie die Randomisierung bei MAC-Adressen von verschiedenen Anbietern, Mobilgeräten und Betriebssystemen umgesetzt wird und welche weiteren Eigenschaften/Attribute von Probe-Requests sonst noch zur Erkennung verwendet werden können.
- Proof-of-Concept Prototyp, der die Verfolgung eines zuvor kategorisierten Mobilgerätes (potentiell über mehrere WLAN-Access-Points) ermöglicht, falls möglich.
- Source-Code des Prototyps
- Protokolle, Dokumentation und gewonnene Daten der Experimente
- Sitzungsprotokolle
- Bericht der Bachelorarbeit gem. den Vorgaben der Hochschule und des Projektbetreuers
- Abstract der Arbeit für das Online-Tool der Hochschule
- Plakat mit den wichtigsten Eckpunkten für die Bachelorarbeitspräsentation
- Eigenständigkeitserklärung
- Einverständniserklärung für die Publikation
- Vereinbarung über Urheber- und Nutzungsrechte

### **Annahmen und Einschränkungen**

Für die Bachelorarbeit sind 12 ECTS vorgesehen. Pro Person fällt somit ein Arbeitsaufwand von 360 Stunden, für die Arbeit gesamthaft 720 Stunden an. Die Bachelorarbeit muss bis zum 15.01.2021 abgegeben werden.

## 19 Projektorganisation

### Projektmitglieder

Mike Schmid



Janik Schlatter



### Externe Schnittstellen

Prof. Beat Stettler Betreuer	Martin Willi Experte	Claudio Fuchs Gegenleser
---------------------------------	-------------------------	-----------------------------

## 20 Management

### Eckdaten

Die Bachelorarbeit startet am 14.09.2020 und endet am 15.01.2021. In diesem Zeitraum werden die 360 Stunden Arbeit der einzelnen Projektmitglieder erbracht. Zu Beginn der Arbeit wurde davon ausgegangen, dass die Arbeit in der zweiten Januarwoche am 08.01.2021 abgegeben werden muss. Die Arbeit wurde auf dieses Abgabedatum hin geplant und durchgeführt. Die zusätzliche Woche wird als Reservezeit für allfällige Nachbesserungen betrachtet.

Gesamthaft stehen in diesem Zeitraum - bis am 08.01.2021 - 17 Wochen zur Verfügung, was in einem Wochendurchschnitt von 21.18 Stunden pro Person entspricht. Um die Planung zu vereinfachen, wurde die zu leistende Zeit auf 21h/Person und Woche festgelegt. Die Übrigen drei Stunden werden in der Abschlussphase verwendet, um die Dokumentation fertigzustellen.

---

Projektdauer	17 Wochen
Anzahl Projektmitglieder	2
Arbeitszeit für Projekt pro Woche und Mitglied	21 Stunden
Total Stunden pro Mitglied	360 Stunden
Arbeitsstunden Total	720 Stunden
Projektstart	14.09.2020
Projektende (geplant)	08.01.2021

---

Die folgenden Absenzen sind beim Projektbeginn bekannt und eingeplant:

---

Mitglied	Von	Bis
Janik Schlatter	22.10.2020	30.10.2020
Beide	24.12.2020	27.12.2020

---

Jedes Projektmitglied ist selbst dafür verantwortlich, die Abwesenheiten entsprechend zu kompensieren.

## Zeitliche Planung

Die 17 Wochen des Projekts werden in vier Phasen unterteilt:

- Initialisierung
- Recherche
- Experimente, Implementation & Evaluation
- Abschluss

In der Abbildung 40 sind die einzelnen Phasen ersichtlich. Da in der dritten Phase die Implementation davon abhängt, dass durch die Experimente statistisch relevante Erkenntnisse gewonnen werden, welche für einen Prototypen genutzt werden können, vereinheitlicht die dritte Phase die Experimente, Evaluation und das Prototyping.

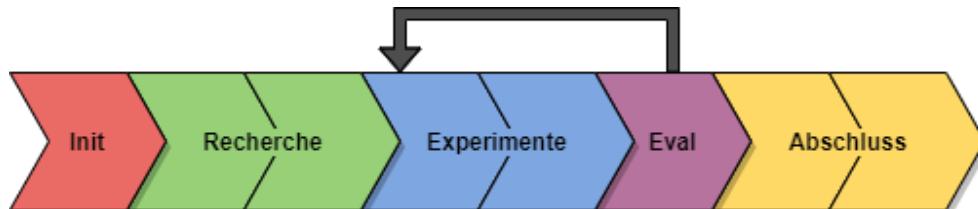


Abbildung 40: Projektphasen

## Iterationen

Eine Iteration beträgt in der zweiten Phase zwei und in der dritten Phase drei Wochen. Jeweils im nächsten Meeting nach einer Iteration werden die gewonnenen Erkenntnisse und das weitere Vorgehen besprochen. In der Tabelle 27 sind die einzelnen Iterationen und deren Tätigkeiten beschrieben.

<b>Iteration</b>	<b>Inhalt</b>	<b>Start</b>	<b>Ende</b>
Initialisierung	Projektstart und Kick-Off Meeting	14.09.2020	20.09.2020
Research	Wissensaufbau, Dokumentenstudium, Paperstudium und vorbereiten erstes Experiment	21.09.2020	04.10.2020
Experiment 1	Planen, durchführen und auswerten erstes Experiment	05.10.2020	25.10.2020
Experiment 2	Planen, durchführen und auswerten zweites Experiment	26.10.2020	29.11.2020
Prototype 1	Implementation erster Prototyp für Klassifizierung eines Mobilgeräts	30.11.2020	13.12.2020
Experiment/Prototype	Iterativ weitere Experimente oder Weiterentwicklung Prototyp	14.12.2020	27.12.2020
Abschluss	Fertigstellen Dokumentation, Aufbereiten der Ergebnisse für Abgabe	28.12.2020	08.01.2021

Tabelle 27: Projektiterationen

## Meilensteine

In der Bachelorarbeit wurden Meilensteine gemäss der Tabelle 28 festgelegt.

Meilenstein	Beschreibung	Termin	Aufwand
M1	Projektplan	20.09.2020	42h (1 Woche)
M2	Recherche	04.10.2020	84h (2 Wochen)
M3	Experimente 1 & 2	29.11.2020	336h (8 Wochen)
M4	Prototyp 1	29.11.2020	84h (2 Wochen)
M5	Experiment 3 & Prototyp 2	27.12.2020	84h (2 Wochen)
M6	Schlussabgabe	08.01.2021	84h (2 Wochen)

Tabelle 28: Meilensteine

## Meetings und Informationsfluss

Die Projektmitglieder haben drei Tage - Dienstag, Mittwoch und Donnerstag - an denen sie jeweils gemeinsam an der Bachelorarbeit arbeiten. An jedem Tag stehen mindestens sechs Lektionen für die Bearbeitung der anstehenden Arbeiten zur Verfügung.

Besprechungen zwischen den Projektteilnehmern und dem Betreuer finden einmal alle ein bis zwei Wochen statt, um sich über den Stand der Arbeit und gewonnene Erkenntnisse auszutauschen. Meetings können vor Ort an der Hochschule in einem dafür geeigneten Raum oder über Microsoft Teams durchgeführt werden.

Die in den Sitzungen besprochenen Pendenzen und Entscheidungen können aus den Sitzungsprotokollen entnommen werden die sich im Anhang 24 befinden.

## 21 Risikomanagement

Mögliche Risiken wurden während der Bachelorarbeit fortlaufend evaluiert und angepasst, um auf unvorhergesehene Ereignisse möglichst verhältnismässig reagieren zu können.

### Risiken

Eine Risikoanalyse mit gewichtetem Schaden und Informationen, wie mit diesen Risiken umgegangen wird, ist im Dokument «TechnischeRisiken.xlsx» zu finden. Die Abbildung 41 zeigt die Risikomatrix zum Beginn der Bachelorarbeit am 14.09.2020.

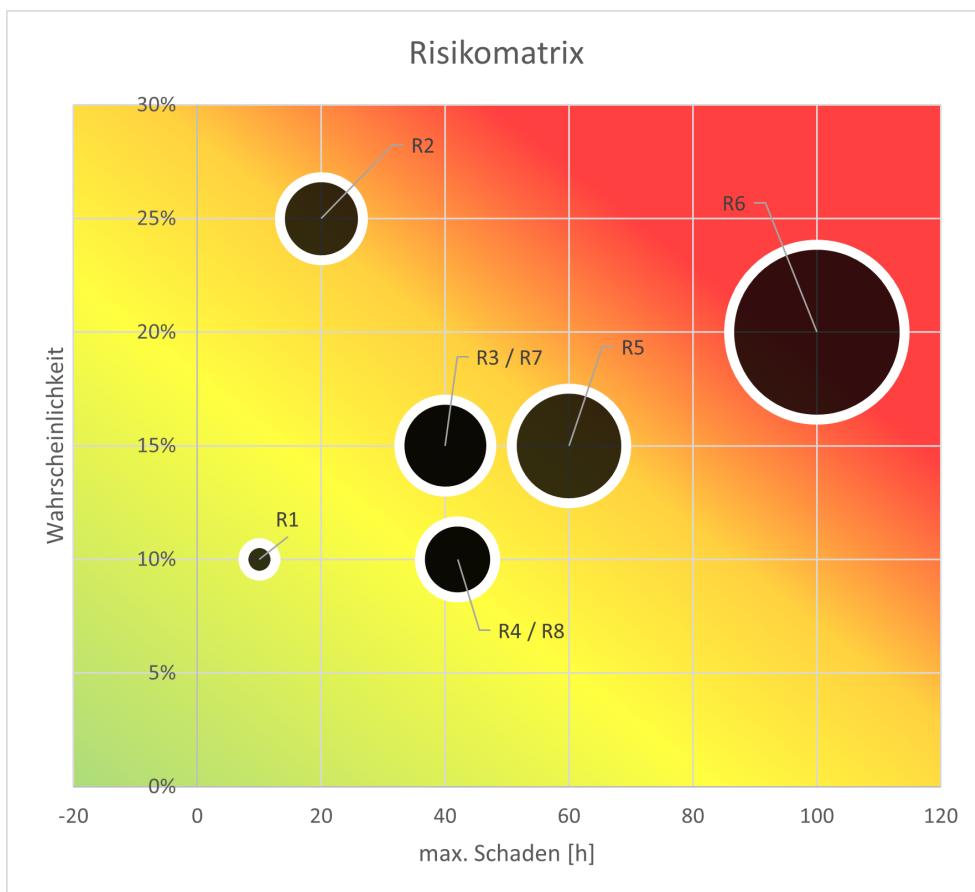


Abbildung 41: Risikomatrix, die Grösse der Blasen entspricht dem gewichteten Schaden

## 21. RISIKOMANAGEMENT

---

R1	Testgeräte	R5	Format Versuchsdaten
R2	Testvorbereitung	R6	Gerätemerkmale
R3	Testdurchführung	R7	Geräteverhalten
R4	Testauswertung	R8	Hardwareverhalten

Tabelle 29: Risiken der Risikomatrix

## Risikoentwicklung und Risikoüberwachung

Nachfolgend sind in den Tabellen 30 die Änderungen der Risiken im Verlauf der Bachelorarbeit dokumentiert.

Risiko	Änderung	Begründung
R6 Gerätemerkmale	Neues Risiko	Hinzufügen Hardwarerisiken
R7 Geräteverhalten	Neues Risiko	Hinzufügen Hardwarerisiken
R8 Geräteverhalten	Neues Risiko	Hinzufügen Hardwarerisiken
R1 Testgeräte	Schaden: 10h → 5h Wahrscheinlichkeit: 10% → 5%	Eingetretenes Risiko
R2 Testvorbereitung	Schaden 20h → 10h Wahrscheinlichkeit: 25% → 5%	Vorbeugung durch Absprache mit Betreuer
R3 Testdurchführung	Schaden: 40h → 30h Wahrscheinlichkeit: 15% → 10%	Eingetretenes Risiko
R4 Testauswertung	Wahrscheinlichkeit: 10% → 5%	Vorbeugung
R5 Format Versuchsdaten	Wahrscheinlichkeit: 15% → 5%	Vorbeugung
R6 Gerätemerkmale	Schaden: 100h → 80h	Eingetretenes Risiko
R7 Geräteverhalten	Schaden: 40h → 30h	Eingetretenes Risiko

Tabelle 30: Änderungen der Risikoanalyse

### **Eingetretene Risiken**

In der Aufzählung 42 sind die eingetretenen Risiken aufgeführt. Risiken aus der Analyse, die nicht in der Aufzählung vorkommen, sind im Projektverlauf nicht eingetreten oder konnten durch vorbeugende Massnahmen verhindert werden.

- R1 - Testgeräte. Die vorhandene Hardware konnte nicht zur Aufzeichnung der Probe-Requests verwendet werden. Es wurde Frühzeitig ein Ersatzlaptop organisiert. Zeitaufwand: ca. 2 Stunden
- R3 - Testdurchführung. Es mussten zwei zusätzliche Messungen durchgeführt werden, da der ursprüngliche Testplan die benötigten Daten nicht abdeckte. Zeitaufwand: ca. 4 Stunden
- R6 - Gerätemerkmale. Die iOS-Geräte liefern zu wenige eindeutige Merkmale, als dass damit ein Fingerprinting durchgeführt werden könnte. Es wird in der Dokumentation darauf eingegangen, sonst aber keine Massnahmen ergriffen.
- R7 - Geräteverhalten. Die Samsung Galaxy S8 verhalten sich im Timing alle unterschiedlich. Die iPhone X haben Unterschiede im Verhalten. In der Analyse wurde auf die Unterschiede eingegangen. Kein Zeitverlust.

Abbildung 42: Eingetretene Risiken

## 22 Arbeitspakete

Die Arbeitspakete in der Bachelorarbeit werden mittels eines Excel-Dokuments verwaltet. Zu Beginn eines Meilensteins werden, wo sinnvoll, Arbeitspakete erfasst, deren Zeitaufwand geschätzt und zugeteilt.

Ein Arbeitspaket beinhaltet:

- Titel
- Beschreibung
- Sub-Tasks
- Definition of Done
- Geschätzter Aufwand
- Tatsächlich benötigter Aufwand

### Auswertung der Arbeitspakete

#### Phase 1: Initialisierung

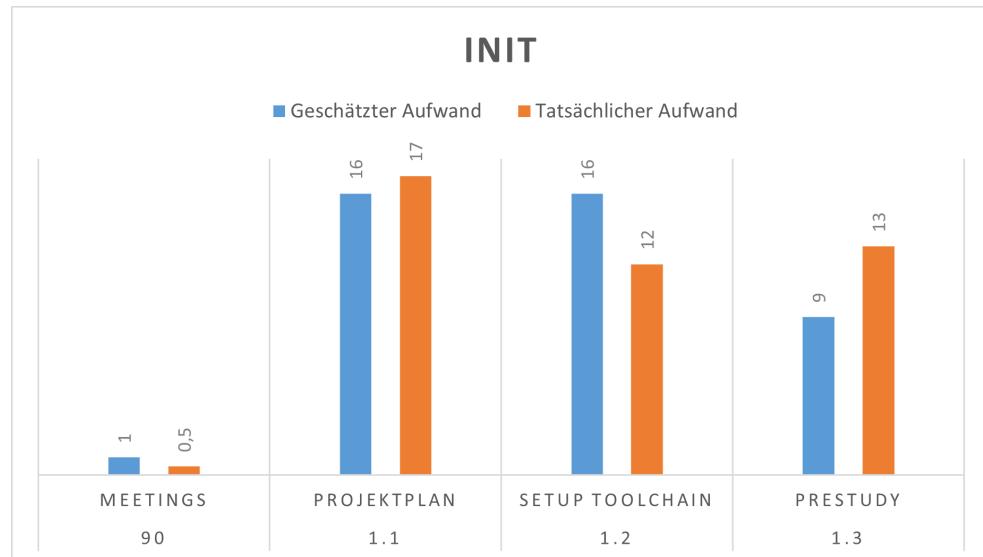


Abbildung 43: Auswertung der Arbeitspakete in der Initialisierungsphase

Geplanter Aufwand: 42 Stunden.

Tatsächlicher Aufwand: 42,5 Stunden.

### Phase 2: Recherche

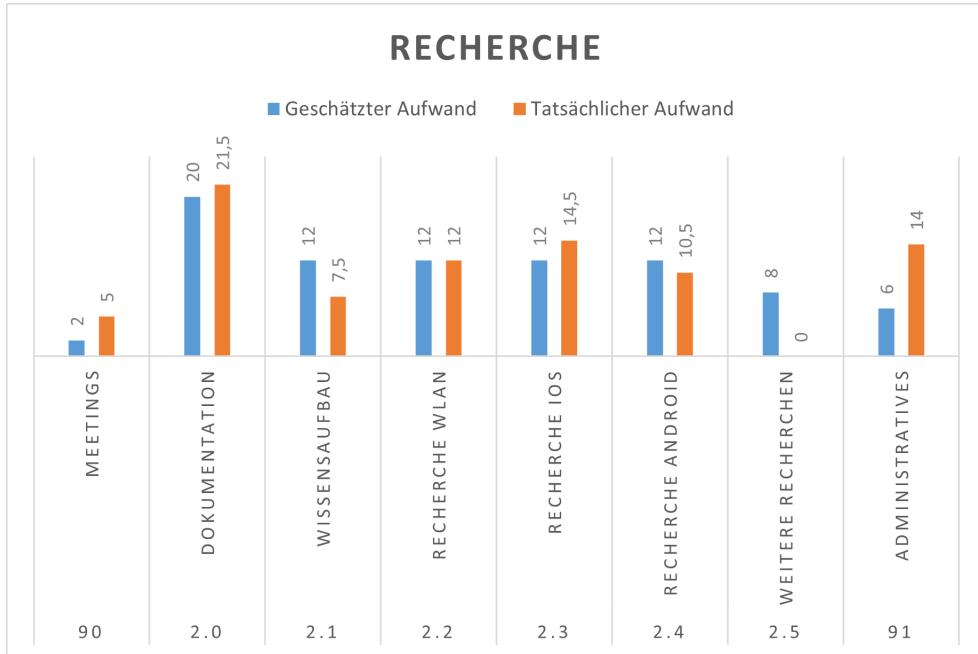


Abbildung 44: Auswertung der Arbeitspakete in der Recherchephase

Geplanter Aufwand: 84 Stunden.

Tatsächlicher Aufwand: 85 Stunden.

Die weiteren Recherchen wurden als Reserve eingeplant, falls sich die Projektteilnehmer für die dritte Phase noch zusätzlich in eine Technologie einarbeiten müssten.

Vor allem der administrative Aufwand wurde unterschätzt. Die Beschaffung der benötigten Mobilgeräte hat mehr Zeit beansprucht, als ursprünglich angenommen.

### Phase 3: Experimente, Evaluierung und Prototyp

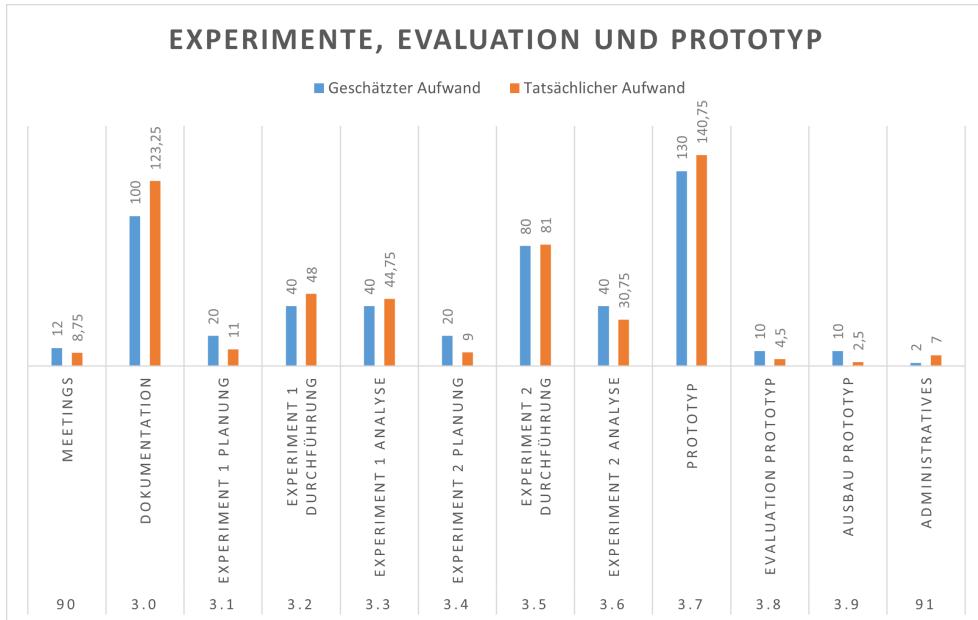


Abbildung 45: Auswertung der Arbeitspakete in der dritten Phase

Geplanter Aufwand: 504 Stunden.

Tatsächlicher Aufwand: 511,25 Stunden.

In der dritten Phase hatte es im Verlauf der Arbeit eine Änderung in der Planung gegeben. Die Android Messungen benötigten mehr Zeit als die iOS-Messungen. Im Meeting der Woche 7 wurde besprochen, dass die Messungen der Android-Geräte um eine Woche verlängert werden. Dadurch wurde die geschätzte Zeit für das Arbeitspaket von 40 auf 80 Stunden erhöht. Die Zeit wurde von den Arbeitspaketen Experiment 1 & 2 Planung (zuvor je 40 Stunden, danach 20 Stunden) entfernt.

Vor allem die Arbeitspakete Dokumentation und Prototyp haben mehr Zeit beansprucht, als ursprünglich geplant.

**Phase 4: Abschluss**

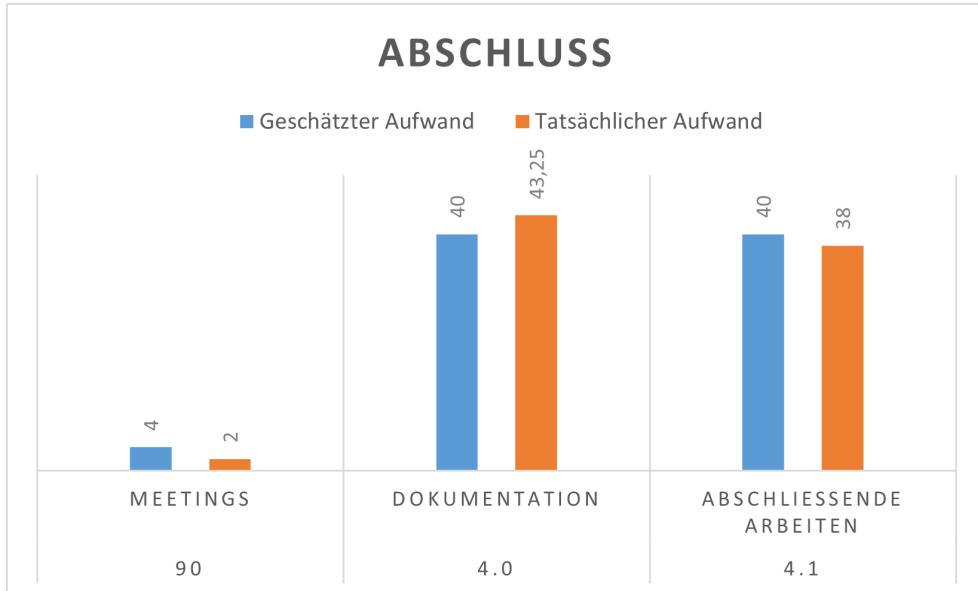


Abbildung 46: Auswertung der Arbeitspakete in der Abschlussphase

Geplanter Aufwand: 84 Stunden.

Tatsächlicher Aufwand: 83.25 Stunden.

## 23 Infrastruktur

Beide Projektmitarbeiter verwenden ihre eigenen Notebooks mit Windows 10 für die Durchführung der Bachelorarbeit. Weitere Hardware, die im Verlauf der Arbeit verwendet wird, wird fortlaufend dokumentiert.

### Übersicht der Tools

Bezeichnung	Verwendungsgrund
GitHub	Versionsverwaltung und Continuous Integration/Delivery
GitKraken / SourceTree	Graphische Benutzeroberfläche für Git-Verwaltung
Latex & Visual Studio Code	Verfassung der Dokumentation
Microsoft Excel	Zeiterfassung, Arbeitspakete und Spreadsheet-Auswertungen
Whatsapp / Discord	Kommunikation innerhalb des Teams
Microsoft Teams	Kommunikation mit dem Betreuer

Tabelle 31: Übersicht der Tools

### Im Verlauf der Arbeit hinzugezogene Tools

Bezeichnung	Verwendungsgrund
Wireshark	Messungen der Mobilgeräte
Pycharm	Python Entwicklungsumgebung

Tabelle 32: Hinzugezogene Tools

### Verwendete Hardware

Die Tabelle 33 zeigt die in den Experimenten und Prototypenentwicklung verwendete Hardware, den Gerätetypen und den Hersteller. Mobilgeräte für die Messungen sind in den jeweiligen Abschnitten dokumentiert (iOS: Abschnitt 8 Tabelle 11. Android: Abschnitt 9 Tabelle 17)

Gerätebezeichnung	Gerätetyp	Hersteller
CAT S60	Smartphone	Bullit Group
Macbook Pro	Laptop	Apple
WaveXpert	WLAN-Messgerät	Softing IT Networks GmbH

Tabelle 33: Verwendete Hardware

## 24 Qualitätsmassnahmen

Um für die Bachelorarbeit die Qualität zu gewährleisten, werden folgende Massnahmen getroffen:

### Dokumentation

Damit die Dokumentation über die Versionsverwaltung gemeinsam von allen Projektteilnehmern vorgenommen werden kann, wird ein Latex-Dokument aufgesetzt und über GitHub verwaltet.

### Projektmanagement

#### GitHub

Für die Versionsverwaltung wird ein GitHub-Repository aufgesetzt. Im Bedarfsfall wird für die Umsetzung eines Prototyps eine Continuous Integration / Continuous Delivery Pipeline und allenfalls weitere Hilfsfunktionen konfiguriert.

Der main- und development-Branch sind beide so abgesichert, dass kein Projektteilnehmer direkt seine Änderung darauf Pushen kann, sondern die Anpassungen über einen Pull-Request eingibt. Dieser Request wird vom jeweils anderen Projektpartner kontrolliert und akzeptiert oder abgelehnt. Die Projektmitglieder verwenden für die Implementation jeweils spezifische Feature-Branches.

Diese Arbeitsweise erlaubt es, einen sicheren Arbeitsfluss zu gewährleisten, bei dem wenig Merge-Konflikte auftreten sollten und bei dem jede Änderung zuerst vom Partner durch ein Review abgesegnet werden muss, bevor sie integriert wird.

### Prozessmodell

Die Bachelorarbeit als Ganzes wird als Wasserfall-Modell aufgezogen. Der Beginn und das Ende der Arbeit sowie drei der Vier Projektphasen sind klar spezifiziert. Lediglich die dritte Phase wird als Iterative Phase durchgeführt, da im Vorherigen nicht gesagt werden kann, welche Resultate aus den Experimenten tatsächlich gewonnen werden können und ob diese Resultate für die Entwicklung eines Prototyps verwendet werden können.

### **Experimenteller Aufbau**

Damit die Experimente mit möglichst signifikanten Ergebnissen ausgeführt werden können, muss vor jeder Durchführung ein Plan mit den Parametern erstellt werden, wie genau das Experiment unter welchen Bedingungen durchgeführt wird, wie die zu erwartenden Ergebnisse aussehen und welche Daten für die weitere Verwendung in welchem Format gespeichert werden sollen. Nach den Experimenten müssen die Ergebnisse ausgewertet und dokumentiert werden.

# Anhang

---

## **Anhang A: Abkürzungsverzeichnis**

---

In der Tabelle 34 sind die in der Arbeit verwendeten Abkürzungen und deren Beschreibung dokumentiert.

---

<b>Abkürzung</b>	<b>Bezeichnung</b>
avg.	Average. Durchschnitt (-wert)
AVT	Arbeitsverwaltungstool der OST-Schweizer Fachhochschule.
BSSID & SSID	(Basic) Service Set ID. Für die Zuordnung von Geräten zu einem Netzwerk.
CSV	Comma Separated Values. Datenformat mit kommaseparierten Einträgen
DS Parameter	Direct Sequence Parameter. Beschreibt vom Netzwerk verwendeten WLAN-Kanal.
ECTS	European Credit Transfer and Accumulation System. Standard für die Vergabe von akademischen Credits.
GHz & MHz	Giga- & Megahertz. Einheit für Frequenz. Ein GHz ist 1000 MHz und 1 MHz ist 1000 Hz.
HT	Higher Throughput. 802.11n Erweiterung für höhere Datendurchsätze.
A-MPDU	Aggregate MPDU. Verfahren im 802.11n Standard für die Aggregation von MPDUs für erhöhten Datendurchsatz.
ICOM	Institut für Kommunikationssysteme der OST-Schweizer Fachhochschule.
IE-Felder	Information-Element-Felder. Zusätzliche Geräteparameter, die in Probe-Requests mitgesendet werden.
IEEE	Institute of Electrical and Electronics Engineers. Berufsverband von Elektroingenieuren welcher verschiedene Standards herausgebracht hat.
IBAT	Inter-Burst-Arrival-Time. Zwischenankunftszeit von Bursts.
IFAT	Inter-Frame-Arrival-Time. Zwischenankunftszeit von Frames.
IFS	Institut für Software der OST-Schweizer Fachhochschule.
INS	Institute for Networked Solutions der OST-Schweizer Fachhochschule.
iOS	Apple Betriebssystem.
IP	Internet-Protocol. Netzwerkprotokoll und Grundlage des Internets.
JSON	Javascript Object Notation. Datenformat für den Datenaustausch zwischen Anwendungen.
Tabelle wird auf der nächsten Seite fortgeführt	

---

Fortführung der Tabelle	
Abkürzung	Bezeichnung
MAC	Media Access Control. Adresse für den Medienzugriff in der Netzwerktechnik.
MPDU	MAC Protocol Data Unit. Nachricht, die zwischen MAC-Entitäten in Form von Frames ausgetauscht wird.
NIC	Network Interface Controller. Netzwerkkontroller-spezifische Kennung für die eindeutige Identifikation von Netzwerkkarten.
Nof	Number of. Anzahl von Entitäten.
OSI	Open Systems Interconnection. Konzeptionelles Modell für die Standardisierung eines Computer Systems.
OUI	Organisational Unique Identifier. Herstellerspezifische Kennung in der MAC-Adresse.
PNO & ePNO	(Enhanced) Preferred Network Offload. Apple Spezifikation für die Erkennung bekannter Netzwerke.
TCP	Transmission Control Protocol. Netzwerkprotokoll für die Kommunikation über das Internet.
U/L Bit	Universal/Local Bit. Synonyme Bezeichnung für das lokale Bit.
WEP	Wired Equivalent Privacy. Wi-Fi Standard für die Netzwerksicherheit.
Wi-Fi	Wireless Fidelity. Protokoll für kabellose Netzwerke. Oft als Synonym für WLAN verwendet.
WLAN	Wireless Local Area Network. Bezeichnung für ein Kabelloses Computernetzwerk.
WPS	Wireless Protected Setup. Netzwerkstandard für schnelle Verbindungen mit WLAN.
UUID	Universally Unique Identifier. 128-Bit ID für die eindeutige Identifikation von Computersystemen oder -anwendungen.

Tabelle 34: Abkürzungen

---

## **Anhang B: Quellenverzeichnis und Bibliografie**

---

Nachfolgend sind die in der Arbeit zitierten Quellen und die für die Recherche verwendete Bibliografie verzeichnet.

---

## **Quellenverzeichnis**

### **Paper: A Study of MAC Address Randomization.**

- Name: Study of MAC Address Randomization in Mobile Devices and When it Fails
- Autoren: Jeremy Martin, Travis Mayberry, Collin Donahue et al.
- Publikation: 2017, Proceedings on Privacy Enhancing Technologies

### **Paper: Defeating MAC Address Randomization Through Timing Attacks**

- Name: Defeating MAC Address Randomization Through Timing Attacks
- Autoren: Célestin Matte, Mathieu Cunche, Franck Rousseau et al.
- Publikation: 2016, WiSec Ausgabe Juli 2016

### **Paper: How Talkative is Your Mobile Device?**

- Name: How Talkative is your Mobile Device? An Experimental Study of Wi-Fi Probe-Requests
- Autoren: Julien Freudinger
- Publikation: 2015, WiSec Ausgabe Juni 2015

### **Paper: Noncooperative 802.11 MAC Layer Fingerprinting**

- Name: Noncooperative 802.11 MAC Layer Fingerprinting and Tracking of Mobile Devices
- Autoren: Pieter Robyns, Bram Bonné, Peter Quax and Wim Lamotte
- Publikation: 2017, Hindawi Security and Communication Networks Volume 2017

### **Paper: Why MAC Address Randomization is not Enough**

- Name: Why MAC Address Randomization is not Enough: An Analysis of Wi-Fi Network Discovery Mechanisms
- Autoren: Mathy Vanhoef, Célestin Matte, Mathieu Cunche et al.
- Publikation: 2016, ASIA CSS Ausgabe 2016

---

## **Pyshark - Packet parser using wireshark's tshark**

- Name: Pyshark Dokumentation
- Maintainer: KimiNewt
- URL <https://kiminewt.github.io/pyshark/>

## **Scapy**

- Name: Scapy Dokumentation
- Autoren: Biondi Philippe
- URL: <https://scapy.readthedocs.io/en/latest/introduction.html>

---

## Bibliografie

Apple Inc. Use private Wi-Fi addresses in iOS 14, iPadOS 14, and watchOS 7

<https://support.apple.com/en-us/HT211227>

Abgerufen am 23.09.2020

Apple Inc. iOS 14 Per-Network MAC Addresses

<https://developer.apple.com/forums/thread/651151>

Abgerufen am 23.09.2020

Buberenko Volodymyr. Diving into Android Oreo security changes.

<https://uptech.team/blog/android-oreo-security-changes>

Abgerufen am 25.09.2020

Buberenko Volodymyr. Android Oreo: all you need to know.

<https://uptech.team/blog/android-oreo-overview>

Abgerufen am 25.09.2020

Cisco. Fundamentals of 802.11 Wireless Sniffing.

<https://www.cisco.com/c/en/us/support/docs/wireless-mobility/80211/>

200527-Fundamentals-of-802-11-Wireless-Sniffing.html

Abgerufen am 03.10.2020

David (Nachname Unbekannt). How to handle randomized MAC addresses on Android 9+ and iOS 14+.

<https://support.adamnet.works/t/how-to-handle-randomized-mac-addresses-on-android-9-and-ios-14/316>

Abgerufen am 29.09.2020

Dorsey Brannon. The Perils of Probe Requests.

<https://medium.com/@brannondorsey/wi-fi-is-broken-3f6054210fa5>

Abgerufen am 30.09.2020

Edwards Jim. Apple's New Anti-Tracking System For iPhones Doesn't Work, Researcher Claims. <https://www.businessinsider.com/ios-8-mac-randomization-wifi-iphone-doesnt-work-2014-10>

Abgerufen am 22.09.2020

---

Gast, Mathew S. 802.11 Wireless Networks: Chapter 4, 802.11 Framing in Detail. O'reilly: <https://www.oreilly.com/library/view/80211-wireless-networks/0596100523/ch04.html>

Gast, Mathew S. 802.11 Wireless Networks: The Definitive Guide, 2nd Edition. O'reilly: <https://www.oreilly.com/library/view/80211-wireless-networks/0596100523/>

Goodin Dan. Shielding MAC addresses from stalkers is hard and Android fails miserably at it. <https://arstechnica.com/information-technology/2017/03/shielding-mac-addresses-from-stalkers-is-hard-android-is-failing-miserably/>

Abgerufen am 22.09.2020

Google LLC. Privacy: MAC Randomization.

<https://source.android.com/devices/tech/connect/wifi-mac-randomization>  
Abgerufen am 23.09.2020

Google LLC. Android Enterprise Security White Paper.

[https://static.googleusercontent.com/media/www.android.com/de//static/2016/pdfs/enterprise/Android\\_Enterprise\\_Security\\_White\\_Paper\\_2019.pdf](https://static.googleusercontent.com/media/www.android.com/de//static/2016/pdfs/enterprise/Android_Enterprise_Security_White_Paper_2019.pdf)  
Abgerufen am 23.09.2020

Harris Nik . Tracking People & Devices with WiFi.

<https://nikharris.com/tracking-people/>  
Abgerufen am 20.11.2020

Hetting Claus. New 'Private Address' iPhone feature could severely impact the Wi-Fi industry, expert says. <https://wifinowglobal.com/news-and-blog/new-private-wi-fi-address-iphone-feature-could-severely-impact-the-wi-fi-industry-expert-says/>

Abgerufen am 22.09.2020

Jankie Michael. We analyse WWDC19. iOS 13 cracks down on location tracking. <https://whatthe.fi/we-analyse-wwdc19-ios-13-cracks-down-on-location-tracking-4a4167b0d7d5>

Abgerufen am 23.09.2020

Johannes (Nachname Unbekannt). MAC Address Randomization on iOS.

<https://www.turais.de/mac-address-randomization-on-ios-12/>  
Abgerufen am 23.09.2020

---

Mamiit Aaron. Apple implements random MAC address on iOS 8. Goodbye, marketers. <https://www.techtimes.com/articles/8233/20140612/apple-implements-random-mac-address-on-ios-8-goodbye-marketers.htm>  
Abgerufen am 23.09.2020

Nayanajith Rasika. CWAP 802.11- Probe Request/Response.  
<https://mrncciew.com/2014/10/27/cwap-802-11-probe-requestresponse/>  
Abgerufen am 10.10.2020

Peterson Mike. iOS 14 MAC randomization privacy feature may cause Cisco enterprise network issues.  
<https://appleinsider.com/articles/20/09/17/ios-14-mac-randomization-privacy-feature-may-cause-cisco-enterprise-network-issues>  
Abgerufen am 01.10.2020

Wikipedia. IEEE 802.11 - Channels and frequencies.  
[https://en.wikipedia.org/wiki/IEEE\\_802.11#Channels\\_and\\_frequencies](https://en.wikipedia.org/wiki/IEEE_802.11#Channels_and_frequencies)  
Abgerufen am 03.10.2020

---

## Anhang C: Abbildungsverzeichnis

---

1	MAC-Adresse Aufbau . . . . .	4
2	Detaillierte MAC-Adresse . . . . .	5
3	Kanalzuteilung im 2.4-GHz-Frequenzband . . . . .	6
4	OSI- und TCP/IP-Referenzmodell . . . . .	8
5	Generisches Management-Frame . . . . .	9
6	Zustände und Klassen für Frames . . . . .	11
7	Sequenzdiagramm Association und Authentifizierung . . . . .	12
8	Frame eines Probe-Requests. . . . .	13
9	Wireshark-Aufzeichnung von Probe-Requests. Die Bursts sind farbig gekennzeichnet. . . . .	14
10	Anzahl Smartphone User in der Schweiz . . . . .	30
11	Verteilung der Betriebssysteme . . . . .	31
12	Verteilung der Hersteller . . . . .	32
13	Offizielle Randomisierungs-Ankündigung von Apple . . . . .	36
14	Messaufbau in der Antennenmesskammer. . . . .	43
15	Gesamtergebnis der iOS-Messungen . . . . .	52
16	Messergebnisse iPhone 8 - iOS 14.0.1 . . . . .	53
17	Messergebnisse iPhone X - Raphael Jud - iOS 14.0.1 . . . . .	53
18	Messergebnisse iPhone X - IFS - iOS 12.3.1 . . . . .	54
19	Messergebnisse iPhone X - IFS - iOS 14.0.1 . . . . .	54
20	Gesamtergebnis der Android-Messungen . . . . .	60
21	Messergebnisse der vier Samsung Galaxy S8-Geräte mit Android 9 . . . . .	61
22	Messergebnisse Samsung Galaxy S9 - Android 10 . . . . .	62
23	Messergebnisse Samsung Galaxy S20 - Android 10 . . . . .	62
24	Messergebnisse Samsung A51 - Android 10 . . . . .	63
25	Messergebnisse Fairphone 3+ - Android 10 . . . . .	64
26	Messergebnisse Google Pixel 3 - Android 11 . . . . .	65
27	Fingerprinting vor iOS 8 / Android 9 . . . . .	69
28	Filterbasiertes, hierarchische Geräteunterscheidung . . . . .	70
29	Zeitbasierte Filterung von ausgesendeten Probe-Requests . . . . .	72

## Anhang C: Abbildungsverzeichnis

---

30	Auswertung der Bursts von vier Samsung Galaxy S8 Geräten.	80
31	Klassendiagramm des Prototyps . . . . .	82
32	Ausgabe des Prototyps für die Messung . . . . .	86
33	Ausgabe des Prototyp für die Messung . . . . .	87
34	Ausgabe des Prototyp für die Messung . . . . .	87
35	Ausgabe des Prototyp für die Messung . . . . .	87
36	Ausgabe des Prototyp für die Messung . . . . .	88
37	Ausgabe des Prototyp für die Messung . . . . .	88
38	Ausgabe des Prototyp für die Messung . . . . .	89
39	Architekturvorschlag für eine Zähleinrichtung . . . . .	91
40	Projektphasen . . . . .	102
41	Risikomatrix, die Grösse der Blasen entspricht dem gewichteten Schaden . . . . .	105
42	Eingetretene Risiken . . . . .	108
43	Auswertung der Arbeitspakete in der Initialisierungsphase . . . . .	109
44	Auswertung der Arbeitspakete in der Recherchephase . . . . .	110
45	Auswertung der Arbeitspakete in der dritten Phase . . . . .	111
46	Auswertung der Arbeitspakete in der Abschlussphase . . . . .	112

---

## Anhang D: Tabellenverzeichnis

---

1	Lokale MAC-Adressen . . . . .	5
2	Kanalzuteilung 5-GHz-Frequenzband. Jeder Kanal ist 20 MHz breit. . . . .	7
3	Felder in einem Management-Frame . . . . .	10
4	Felder in einem Probe-Request . . . . .	13
5	Attribute im Frame-Control-Feld . . . . .	16
6	Versionsgeschichte des iOS für iPhones, alle grau eingefärbten Zeilen benutzen MAC Randomisierung . . . . .	35
7	Abdeckung der verschiedenen Scans in iOS 8 & 9 . . . . .	37
8	Versionsgeschichte des Android, alle grau eingefärbten Zeilen benutzen MAC-Randomisierung. Supported Devices erhalten noch aktuelle Sicherheitsuptades . . . . .	39
9	Messplan: SSID-unabhängige Messungen . . . . .	46
10	Messplan: SSID-abhängige Messungen . . . . .	47
11	Gemessene iOS-Geräte . . . . .	49
12	Ergebnisse der iOS-Messungen . . . . .	51
13	IE-Felder, die in allen Messungen vorkommen . . . . .	55
14	IE-Felder der einzelnen iPhones . . . . .	55
15	Herstellerspezifische Felder (Vendor Specific - 221) . . . . .	56
16	Häufigste OUIs in der Vorarbeit . . . . .	56
17	Gemessene Android-Geräte . . . . .	57
18	Ergebnisse der Android-Messungen . . . . .	59
19	IE-Felder, die in allen Messungen vorkommen . . . . .	66
20	Herstellerspezifische Felder (Vendor Specific - 221) . . . . .	67
21	Klassenvariablen der Burst-Klasse . . . . .	83
22	Klassenmethoden der Burst-Klasse . . . . .	83
23	Klassenvariablen der Frame-Klasse . . . . .	84
24	Klassenvariablen der Tag-Klasse . . . . .	84
25	Klassenmethoden der Filter-Klasse . . . . .	85
26	Changelog Projektmanagement . . . . .	96

## Anhang D: Tabellenverzeichnis

---

27	Projektiterationen . . . . .	103
28	Meilesteine . . . . .	104
29	Risiken der Risikomatrix . . . . .	106
30	Änderungen der Risikoanalyse . . . . .	107
31	Übersicht der Tools . . . . .	113
32	Hinzugezogene Tools . . . . .	113
33	Verwendete Hardware . . . . .	114
34	Abkürzungen . . . . .	iii
35	Ergebnisse iPhone 8 Messungen, iOS-Version 14 . . . . .	xv
36	Ergebnisse iPhone X Messungen, iOS-Version 14 . . . . .	xvi
37	Ergebnisse iPhone X Messungen, iOS-Version 12 . . . . .	xvii
38	Ergebnisse iPhone X Messungen, iOS-Version 14 . . . . .	xviii
39	Ergebnisse Samsung A51 Messungen, Android-Version 10 . . . . .	xix
40	Ergebnisse Samsung Galaxy S9 Messungen, Android-Version 10 . . . . .	xx
41	Ergebnisse Samsung Galaxy S20 Messungen, Android-Version 10 . . . . .	xxi
42	Ergebnisse Google Pixel 3 Messungen, Android-Version 11 . . . . .	xxii
43	Ergebnisse Samsung Galaxy S8 (Erstes) Messungen, Android-Version 9 . . . . .	xxiii
44	Ergebnisse Samsung Galaxy S8 (Zweites) Messungen, Android-Version 9 . . . . .	xxiii
45	Ergebnisse Samsung Galaxy S8 (Drittes) Messungen, Android-Version 9 . . . . .	xxiv
46	Ergebnisse Samsung Galaxy S8 (Viertes) Messungen, Android-Version 9 . . . . .	xxiv
47	Ergebnisse Fairphone 3+ Messungen, Android-Version 10 . . . . .	xxv
48	Risikotabelle Projektbeginn . . . . .	xxxv
49	Risikotabelle Projektabschluss . . . . .	xxxvi
50	Beschreibung zu Risikotabellen 48 und 49 . . . . .	xxxvii

Mobile Fingerprinting	xiii
-----------------------	------

---

## Anhang E: Experimentelle Daten

---

Die folgenden Tabellen beschreiben die Auswertung der Messergebnisse aus den Experimenten. Die Ergebnisse sind nach den einzelnen Mobilgeräten und -Versionen sortiert.

## iOS Messungen

iPhone 8, iOS-Version 14.0.1

Messung	Anzahl Probe Requests	Anzahl Bursts	min. Burstgrösse	avg. Burstgrösse	max. Burstgrösse	Verpasste Frames	Zwischen- ankunftszeit
Aktiv Lang	384	86	1	4,465	18	241	42,16 s
Passiv Lang	71	29	1	2,448	6	129	126,83 s
Verbunden Lang	67	21	1	3,191	5	135	173,83 s
Aktiv On On	121	28	1	4,321	18	105	21,71 s
Aktiv On Off	72	23	1	3,130	8	100	21,52 s
Passiv On On	81	34	1	2,382	7	54	16,47 s
Passiv On Off	104	36	1	2,889	7	167	16,83 s
Verbunden On On	52	4	1	13,000	26	78	183,81 s
Verbunden On Off	23	5	1	4,600	12	35	136,39 s
Hotspot Verfügbar	66	15	1	4,400	18	34	38,58 s
Flugmodus	111	27	1	4,111	9	85	8,00 s
Startup	67	19	1	3,526	9	33	18,11 s
TOTAL	1219	327	1	4,372	26	1196	67,02 s

Tabelle 35: Ergebnisse iPhone 8 Messungen, iOS-Version 14

**iPhone X - Raphael Jud, iOS-Version 14.0.1**

Messung	Anzahl Probe Requests	Anzahl Bursts	min. Burstgrösse	avg. Burstgrösse	max. Burstgrösse	Verpasste Frames	Zwischen- ankunftszeit
Aktiv Lang	222	90	1	2,467	8	206	40,05 s
Passiv Lang	587	161	1	3,646	7	1578	22,20 s
Aktiv On On	70	18	1	3,889	12	138	34,25 s
Passiv On On	102	19	1	5,368	10	236	27,54 s
Verbunden On On	10	3	2	3,333	4	10	192,56 s
Hotspot Verfügbar	79	18	2	4,389	8	214	33,03 s
TOTAL	1070	309	1	3,849	12	2382	58,27 s

Tabelle 36: Ergebnisse iPhone X Messungen, iOS-Version 14

**iPhone X, iOS-Version 12.3.1**

Messung	Anzahl Probe Requests	Anzahl Bursts	min. Burstgrösse	avg. Burstgrösse	max. Burstgrösse	Verpasste Frames	Zwischen- ankunftszeit
Passiv Lang	52	14	1	3,714	7	105	266,81 s
Aktiv On On	73	24	1	3,042	6	30	25,32 s
Aktiv On Off	141	41	2	3,439	7	231	14,70 s
Passiv On On	122	22	1	5,545	12	64	28,51 s
Passiv On Off	47	10	2	4,700	6	28	60,91 s
Verbunden On On	147	37	1	3,973	8	578	8,77 s
Verbunden On Off	81	20	1	4,050	12	638	8,88 s
Flugmodus	57	15	1	3,800	6	113	8,88 s
Startup	42	10	2	4,200	6	22	13,90 s
TOTAL	762	193	1	4,051	12	1809	48,52 s

Tabelle 37: Ergebnisse iPhone X Messungen, iOS-Version 12

**iPhone X, iOS-Version 14.0.1**

Messung	Anzahl Probe Requests	Anzahl Bursts	min. Burstgrösse	avg. Burstgrösse	max. Burstgrösse	Verpasste Frames	Zwischen- ankunftszeit
Aktiv Lang	418	123	1	3,398	14	547	19,71 s
Passiv Lang	75	24	1	3,125	7	133	151,65 s
Verbunden Lang	58	57	2	4,526	18	489	47,06 s
Aktiv On On	107	21	1	5,095	9	145	29,64 s
Aktiv On Off	53	17	1	3,118	9	63	32,72 s
Passiv On On	75	16	1	4,688	7	158	36,60 s
Passiv On Off	89	20	1	4,450	8	64	29,80 s
Verbunden On On	229	47	1	4,872	9	302	12,15 s
Verbunden On Off	328	45	2	7,289	11	302	12,99 s
Flugmodus	76	15	1	5,067	10	44	9,91 s
Startup	76	15	2	5,067	9	73	14,08 s
TOTAL	1784	400	1	4,607	18	2320	36,03 s

Tabelle 38: Ergebnisse iPhone X Messungen, iOS-Version 14

## Android Messungen

### Samsung A51, Android Version 10

Messung	Anzahl Probe Requests	Anzahl Bursts	min. Burstgrösse	avg. Burstgrösse	max. Burstgrösse	Verpasste Frames	Zwischen- ankunftszeit
Aktiv Lang Mobile On	115	19	2	6,053	12	92	168,11
Aktiv Lang Mobile Off	14	3	2	4,667	8	13	1148,89
Passiv Lang Mobile On	34	6	2	5,667	12	16	551,28
Passiv Lang Mobile Off	25	4	3	6,250	8	26	369,40
Aktiv Mobile On	28	1	4	7,000	11	10	35,16
Aktiv Mobile Off	52	7	6	7,429	8	28	92,94
Passiv Mobile On	30	5	3	6,000	8	20	104,54
Passiv Mobile Off	21	3	6	7,000	8	11	272,61
Flugmodus	32	5	3	6,400	11	21	8,15
Startup	61	8	4	7,625	10	33	9,28
TOTAL	412	61	2	6,409	12	270	276,04

Tabelle 39: Ergebnisse Samsung A51 Messungen, Android-Version 10

## Samsung Galaxy S9, Android Version 10

Messung	Anzahl Probe Requests	Anzahl Bursts	min. Burstgrösse	avg. Burstgrösse	max. Burstgrösse	Verpasste Frames	Zwischen- ankunftszeit
Aktiv Lang On On	98	36	1	2,722	6	230	100,33
Aktiv Lang On Off	102	30	1	3,400	7	252	118,88
Passiv Lang On On	5	3	1	1,667	3	5	1001,52
Verbunden	147	46	2	3,196	6	89	12,92
Flugmodus	16	5	2	3,200	4	29	10,37
Startup	16	5	2	3,200	5	27	27,26
TOTAL	384	125	1	2,897	7	632	211,88

Tabelle 40: Ergebnisse Samsung Galaxy S9 Messungen, Android-Version 10

## Samsung Galaxy S20, Android Version 10

Messung	Anzahl Probe Requests	Anzahl Bursts	min. Burstgrösse	avg. Burstgrösse	max. Burstgrösse	Verpasste Frames	Zwischen- ankunftszeit
Aktiv Lang	60	32	1	1,875	2	28	112,74
Passiv Lang 1	142	130	1	1,092	2	12	27,92
Passiv Lang 2	32	16	2	2,000	2	16	221,63
Verbunden Lang	40	15	1	1,867	2	0	252,58
Aktiv 10 SSIDs On On	10	5	2	2,000	2	5	125,59
Aktiv 10 SSIDs On Off	26	13	2	2,000	2	13	37,95
Aktiv 5 SSIDs On Off	60	30	2	2,000	2	30	18,53
Aktiv 1 SSIDs On On	26	13	2	2,000	2	13	39,67
Aktiv 1 SSIDs On Off	10	6	1	1,667	2	4	102,90
Passiv 10 SSIDs On On	6	3	2	2,000	2	3	219,42
Passiv 10 SSIDs On Off	7	4	1	1,750	2	3	116,26
Passiv 5 SSIDs On On	8	4	2	2,000	2	4	141,44
Passiv 5 SSIDs On Off	4	2	2	2,000	2	2	263,26
Passiv 1 SSIDs On On	10	5	2	2,000	2	5	110,93
Passiv 1 SSIDs On Off	6	3	2	2,000	2	3	180,52
Flugmodus	26	14	1	1,857	2	12	16,45
Startup	16	8	2	2,000	2	8	18,21
TOTAL	489	303	1	1,889	2	161	118,00

Tabelle 41: Ergebnisse Samsung Galaxy S20 Messungen, Android-Version 10

## Google Pixel 3, Android Version 11

Messung	Anzahl Probe Requests	Anzahl Bursts	min. Burstgrösse	avg. Burstgrösse	max. Burstgrösse	Verpasste Frames	Zwischen- ankunftszeit
Aktiv Lang	1785	375	1	4,760	11	568	9,52
Passiv Lang	271	59	2	4,593	11	133	61,86
Verbunden	566	63	2	8,968	13	365	9,30
Aktiv 10 SSIDs On On	364	69	1	5,275	10	127	8,72
Aktiv 10 SSIDs On Off	93	22	1	4,227	10	45	28,77
Aktiv 5 SSIDs On On	127	29	1	4,379	15	50	21,35
Aktiv 5 SSIDs On Off	93	22	1	4,227	10	45	28,77
Aktiv 1 SSIDs On On	320	63	1	5,079	9	128	9,57
Aktiv 1 SSIDs On Off	261	66	1	3,955	9	113	9,21
Passiv 10 SSIDs On On	17	13	1	1,308	5	1	45,36
Passiv 10 SSIDs On Off	4	4	1	1,000	1	0	140,04
Passiv 5 SSIDs On On	32	8	1	4,000	14	1	76,87
Passiv 5 SSIDs On Off	11	11	1	1,000	1	0	56,03
Passiv 1 SSIDs On On	428	72	1	5,944	10	236	8,32
Passiv 1 SSIDs On Off	626	71	1	8,817	14	249	8,23
Flugmodus	33	8	2	4,125	7	12	14,64
Startup	17	5	1	3,400	5	3	24,25
TOTAL	5048	960	1	4,415	15	2076	32,99

Tabelle 42: Ergebnisse Google Pixel 3 Messungen, Android-Version 11

**Samsung Galaxy S8 One, Android Version 9**

Messung	Anzahl Probe Requests	Anzahl Bursts	min. Burstgrösse	avg. Burstgrösse	max. Burstgrösse	Verpasste Frames	Zwischen- ankunftszeit
Aktiv 20 min	44	12	2	3,667	6	32	94,27

Tabelle 43: Ergebnisse Samsung Galaxy S8 (Erstes) Messungen, Android-Version 9

**Samsung Galaxy S8 Two, Android Version 9**

Messung	Anzahl Probe Requests	Anzahl Bursts	min. Burstgrösse	avg. Burstgrösse	max. Burstgrösse	Verpasste Frames	Zwischen- ankunftszeit
Aktiv 20 min	60	11	2	5,455	10	64	101,21

Tabelle 44: Ergebnisse Samsung Galaxy S8 (Zweites) Messungen, Android-Version 9

### Samsung Galaxy S8 Three, Android Version 9

Messung	Anzahl Probe Requests	Anzahl Bursts	min. Burstgrösse	avg. Burstgrösse	max. Burstgrösse	Verpasste Frames	Zwischen- ankunftszeit
Aktiv 20 min	149	20	1	7,450	13	149	55,78

Tabelle 45: Ergebnisse Samsung Galaxy S8 (Drittes) Messungen, Android-Version 9

### Samsung Galaxy S8 Four, Android Version 9

Messung	Anzahl Probe Requests	Anzahl Bursts	min. Burstgrösse	avg. Burstgrösse	max. Burstgrösse	Verpasste Frames	Zwischen- ankunftszeit
Aktiv 20 min	101	14	4	7,214	11	200	80,29

Tabelle 46: Ergebnisse Samsung Galaxy S8 (Viertes) Messungen, Android-Version 9

## Fairphone 3+, Android Version 10

Messung	Anzahl Probe Requests	Anzahl Bursts	min. Burstgrösse	avg. Burstgrösse	max. Burstgrösse	Verpasste Frames	Zwischen- ankunftszeit
Aktiv Lang Macbook	308	52	1	5,923	17	152	69,53
Aktiv Lang WaveXpert	251	22	7	11,409	19	61	124,61
Passiv Lang Macbook	112	56	2	2,000	2	0	63,28
Passiv Lang WaveXpert	110	56	1	1,964	2	0	63,28
Verbungen Lang	407	28	1	12,893	20	134	130,54
Aktiv Ohne Sim 30 min	209	12	1	10,083	14	35	153,74
Aktiv 10 SSIDs	64	5	10	12,800	16	20	122,58
Aktiv 5 SSIDs	64	4	15	15,750	16	18	159,75
Aktiv 0 SSIDs	60	4	14	15,000	16	17	159,81
Aktiv On On	112	7	1	16,000	25	72	92,37
Aktiv On Off	55	3	14	18,000	21	29	225,49
Passiv On On	64	54	1	1,185	2	0	10,75
Passiv On Off	79	52	1	1,500	17	4	9,76
5GHz Messung	267	19	1	12,053	18	122	65,11
Flugmodus	72	5	11	14,400	18	45	10,87
Startup	95	7	9	13,571	19	35	37,58
TOTAL	2329	386	1	10,283	25	744	93,69

Tabelle 47: Ergebnisse Fairphone 3+ Messungen, Android-Version 10

---

## **Anhang F: Verhaltenskatalog der Mobilgeräte**

---

In den folgenden Abschnitten ist das Verhalten der einzelnen Mobilgeräte aus den Messungen in Form eines Gerätekatalogs beschrieben.

---

## Android-Geräte

### Samsung A51

Generelle Informationen zu den Messungen:

---

Android Version	10
Anzahl gemessene Bursts	61
Bursts pro Minute	0.47
Min Burst Grösse	2
Max Burst Grösse	12
Avg Burst Grösse	6.4
Avg Inter-Burst-Arrival-Time	276.04

---

Verhalten des Gerätes:

---

Sequenz Nummer	Nicht Randomisiert
Local Bit	Immer Gesetzt
IE-Felder	<ul style="list-style-type: none"><li>• 0 : SSID</li><li>• 1 : Supported Rates</li><li>• 50 : Extended Supported Rates</li><li>• 3 : DS Parameters</li><li>• 45 : HT Capabilities</li><li>• 127 : Extended Capabilities</li><li>• 221 : Microsoft Corp.</li></ul>
Spezielles Verhalten	Das Samsung A51 sendet die Bursts immer mit der gleichen MAC-Adresse

---

---

## Samsung Galaxy S9

Generelle Informationen zu den Messungen:

---

Android Version	10
Anzahl gemessene Bursts	125
Bursts pro Minute	0.65
Min Burst Grösse	1
Max Burst Grösse	7
Avg Burst Grösse	2.897
Avg Inter-Burst-Arrival-Time	211.88

---

Verhalten des Gerätes:

---

Sequenz Nummer	Randomisiert
Local Bit	Immer Gesetzt
	<ul style="list-style-type: none"><li>• 0 : SSID</li><li>• 1 : Supported Rates</li><li>• 50 : Extended Supported Rates</li><li>• 3 : DS Parameters</li><li>• 45 : HT Capabilities</li><li>• 127 : Extended Capabilities</li><li>• 221 : Epigram, Inc.</li><li>• 221 : Microsoft Corp.</li><li>• 221 : Broadcom</li><li>• 221 : Epigram, Inc.</li></ul>
IE-Felder	
Spezielles Verhalten	-

---

---

## Samsung Galaxy S20

Generelle Informationen zu den Messungen:

---

Android Version	10
Anzahl gemessene Bursts	303
Bursts pro Minute	0.97
Min Burst Grösse	1
Max Burst Grösse	2
Avg Burst Grösse	1.889
Avg Inter-Burst-Arrival-Time	118

---

Verhalten des Gerätes:

---

Sequenz Nummer	Randomisiert
Local Bit	Immer Gesetzt
IE-Felder	<ul style="list-style-type: none"><li>• 0 : SSID</li><li>• 1 : Supported Rates</li><li>• 50 : Extended Supported Rates</li><li>• 3 : DS Parameters</li><li>• 45 : HT Capabilities</li><li>• 127 : Extended Capabilities</li><li>• 255 : FILS Req. Params.</li><li>• 255 : HE Capabilities</li><li>• 221 : Epigram, Inc.</li><li>• 221 : Microsoft Corp.</li><li>• 221 : Broadcom</li><li>• 221 : Wi-Fi - Alliance</li></ul>
Spezielles Verhalten	Das Samsung Galaxy S20 sendet in der Regel Bursts mit der Grösse 2

---

---

## Google Pixel 3

Generelle Informationen zu den Messungen:

---

Android Version	11
Anzahl gemessene Bursts	960
Bursts pro Minute	3.29
Min Burst Grösse	1
Max Burst Grösse	15
Avg Burst Grösse	4.415
Avg Inter-Burst-Arrival-Time	32.99

---

Verhalten des Gerätes:

---

Sequenz Nummer	Randomisiert
Local Bit	Immer Gesetzt
IE-Felder	<ul style="list-style-type: none"><li>• 0 : SSID</li><li>• 1 : Supported Rates</li><li>• 50 : Extended Supported Rates</li><li>• 3 : DS Parameters</li><li>• 221 : Microsoft Corp.</li><li>• 221 : Wi-Fi - Alliance</li></ul>
Spezielles Verhalten	Das Pixel 3 sendet mehr Bursts als alle anderen gemessenen Geräte.

---

---

## Fairphone 3

Generelle Informationen zu den Messungen:

---

Android Version	10
Anzahl gemessene Bursts	386
Bursts pro Minute	1.26
Min Burst Grösse	1
Max Burst Grösse	25
Avg Burst Grösse	10.293
Avg Inter-Burst-Arrival-Time	93.69

---

Verhalten des Gerätes:

---

Sequenz Nummer	Nicht Randomisiert
Local Bit	Immer Gesetzt <ul style="list-style-type: none"><li>• 0 : SSID</li><li>• 1 : Supported Rates</li><li>• 50 : Extended Supported Rates</li><li>• 3 : DS Parameters [ca. 70%]</li><li>• 45 : HT Capabilities [ca. 70%]</li><li>• 221 : Microsoft Corp. [ca. 70%]</li></ul>
IE-Felder	
Spezielles Verhalten	Das Pixel 3 sendet mehr Bursts als alle anderen gemessenen Geräte.

---

---

## Apple-Geräte

### iPhone 8

Generelle Informationen zu den Messungen:

---

iOS Version	14.0.1
Anzahl gemessene Bursts	327
Bursts pro Minute	2.16
Min Burst Grösse	1
Max Burst Grösse	26
Avg Burst Grösse	4.372
Avg Inter-Burst-Arrival-Time	62.02

---

Verhalten des Gerätes:

---

Sequenz Nummer	Randomisiert
Local Bit	Randomisiert
IE-Felder	<ul style="list-style-type: none"><li>• 0 : SSID</li><li>• 1 : Supported Rates</li><li>• 50 : Extended Supported Rates</li><li>• 3 : DS Parameters</li><li>• 45 : HT Capabilities</li><li>• 127 : Extended Capabilities</li><li>• 107 : Interworking [ca. 25%]</li><li>• 221 : Apple [ca. 45%]</li><li>• 221 : Microsoft Corp. [ca. 45%]</li><li>• 221 : Broadcom [ca. 45%]</li></ul>

---

Spezielles Verhalten -

---

---

## iPhone X

Generelle Informationen zu den Messungen:

---

iOS Version	14.0.1
Anzahl gemessene Bursts	309
Bursts pro Minute	1.66
Min Burst Grösse	1
Max Burst Grösse	12
Avg Burst Grösse	3.849
Avg Inter-Burst-Arrival-Time	58.27

---

Verhalten des Gerätes:

---

Sequenz Nummer	Randomisiert
Local Bit	Randomisiert
IE-Felder	<ul style="list-style-type: none"><li>• 0 : SSID</li><li>• 1 : Supported Rates</li><li>• 50 : Extended Supported Rates</li><li>• 3 : DS Parameters</li><li>• 45 : HT Capabilities</li><li>• 127 : Extended Capabilities</li><li>• 107 : Interworking [ca. 5%]</li></ul>

---

Spezielles Verhalten -

---

---

## **Anhang G: Risikotabelle**

---

In den nachfolgenden Unterabschnitten sind der Projektplan, jeweils zum Beginn der Bachelorarbeit und zum Ende der Bachelorarbeit abgelegt.

Nr	Titel	Maximaler Schaden [h]	Eintrittswahrscheinlichkeit	Gewichteter Schaden	Vorbeugung	Verhalten beim Eintreten	Risikoabdeckung
R1	Testgeräte	10	10%	1	Genaue Planung, welche Geräte benötigt werden und wie diese beschafft werden können	Anpassen der Gerätespezifikationen und beschaffung über weitere Quellen (Ausleihen von Instituten, Kollegen, Familie)	Genaue Spezifikation der benötigten Geräte und -betriebssysteme. Frühe Beschaffung durch Projektteilnehmer und -Betreuer
R2	Testvorbereitung	20	25%	5	Genaue Spezifizierung von Testparametern, bevor der Versuchsaufbau stattfindet	Testspezifikationen müssen überarbeitet werden.	Verifizieren, dass Versuchsaufbau durchführbar ist und zu den erwarteten Ergebnissen führt. Abklären des Aufbaus mit Experten.
R3	Testdurchführung	40	15%	6	Testgeräte genau auf den Versuchsaufbau vorbereiten, genaue Testspezifikationen. Sicherstellen durch periodische Überprüfung, dass laufende Tests die Betriebsparameter erfüllen	Der gesamte Test oder Teile davon müssen erneut durchgeführt werden.	Versuchsaufbau unter kontrollierten Bedingungen. Periodisches Überprüfen nach jedem Versuchsschritt, dass keine Fehler vorgekommen sind
R4	Testauswertung	42	10%	4,2	Erwartete Resultate in der Versuchsplanning definieren und bei der Durchführung kontrollieren	Versuchsaufbau anpassen und Tests wiederholen	Risiko lässt sich mitigen, indem die Gerätespezifikation im vornherein sehr genau studiert wird und die erwarteten Ergebnisse in der Versuchsplanning definiert werden
R5	Format Versuchsdaten	60	15%	9	Datenformat und Erwartete Resultate in der Testplanung spezifizieren. Vorbereiten der Testdokumentation	Versuche müssen wiederholt werden	Recherche, welche Formate und Speichermöglichkeiten sich am besten für die Versuche eignen.
R6	Gerätemerkmale	100	20%	20	Recherchen, um vorab zu wissen, wie sich die Mobilgeräte verhalten	Abklären mit Betreuer, ob die Aufgabenstellung der BA an die Erkenntnisse angepasst werden muss.	Höhere Anzahl Messungen und genau spezifizierte erwartete Messergebnisse
R7	Geräteverhalten	40	15%	6	Mehrere Messungen mit gleichem Gerätetyp und OS-Version, um Abweichungen zu erkennen.	Mehr Geräte organisieren und weitere Versuche anstellen	Messungen mit unterschiedlichen Gerätetypen und verschiedenen Betriebssystemversionen
R8	Hardwareverhalten	42	10%	4,2	Mehrere Messungen mit identischem OS auf unterschiedlicher HW durchführen	Mehr Geräte organisieren und weitere Versuche anstellen	Mehrere Geräte mit identischem OS verwenden

Tabelle 48: Risikotabelle Projektbeginn

Nr	Titel	Maximaler Schaden [h]	Eintrittswahrscheinlichkeit	Gewichteter Schaden	Vorbeugung	Verhalten beim Eintreten	Risikoabdeckung
R1	Testgeräte	5	5%	0,25	Genaue Planung, welche Geräte benötigt werden und wie diese beschafft werden können	Anpassen der Gerätespezifikationen und beschaffung über weitere Quellen (Ausleihen von Instituten, Kollegen, Familie)	Genaue Spezifikation der benötigten Geräte und -betriebssysteme. Frühe Beschaffung durch Projektteilnehmer und -Betreuer
R2	Testvorbereitung	10	10%	1	Genaue Spezifizierung von Testparametern, bevor der Versuchsaufbau stattfindet	Testspezifikationen müssen überarbeitet werden.	Verifizieren, dass Versuchsaufbau durchführbar ist und zu den erwarteten Ergebnissen führt. Abklären des Aufbaus mit Experten.
R3	Testdurchführung	30	10%	3	Testgeräte genau auf den Versuchsaufbau vorbereiten, genaue Testspezifikationen. Sicherstellen durch periodische Überprüfung, dass laufende Tests die Betriebsparameter erfüllen	Der gesamte Test oder Teile davon müssen erneut durchgeführt werden.	Versuchsaufbau unter kontrollierten Bedingungen. Periodisches Überprüfen nach jedem Versuchsschritt, dass keine Fehler vorgekommen sind
R4	Testauswertung	42	5%	2,1	Erwartete Resultate in der Versuchsplanning definieren und bei der Durchführung kontrollieren	Versuchsaufbau anpassen und Tests wiederholen	Risiko lässt sich mitigen, indem die Gerätespezifikation im vornherein sehr genau studiert wird und die erwarteten Ergebnisse in der Versuchsplanning definiert werden
R5	Format Versuchsdaten	60	5%	3	Datenformat und Erwartete Resultate in der Testplanung spezifizieren. Vorbereiten der Testdokumentation	Versuche müssen wiederholt werden	Recherche, welche Formate und Speichermöglichkeiten sich am besten für die Versuche eignen.
R6	Gerätemerkmale	80	20%	16	Recherchen, um vorab zu wissen, wie sich die Mobilgeräte verhalten	Abklären mit Betreuer, ob die Aufgabenstellung der BA an die Erkenntnisse angepasst werden muss.	Höhere Anzahl Messungen und genau spezifizierte erwartete Messergebnisse
R7	Geräteverhalten	30	15%	4,5	Mehrere Messungen mit gleichem Gerätetyp und OS-Version, um Abweichungen zu erkennen.	Mehr Geräte organisieren und weitere Versuche anstellen	Messungen mit unterschiedlichen Gerätetypen und verschiedenen Betriebssystemversionen
R8	Hardwareverhalten	42	5%	2,1	Mehrere Messungen mit identischem OS auf unterschiedlicher HW durchführen	Mehr Geräte organisieren und weitere Versuche anstellen	Mehrere Geräte mit identischem OS verwenden

Tabelle 49: Risikotabelle Projektabschluss

Titel	Beschreibung
Testgeräte	Die erforderlichen Testgeräte (Smartphones/Access-Points) können nicht organisiert werden oder sind im Rahmen der Experimente nicht brauchbar.
Testvorbereitung	Die Testspezifikation ist fehlerhaft/mangelhaft und Experimente können nicht durchgeführt werden.
Testdurchführung	Fehler, welche bei der Testdurchführung auftreten.
Testauswertung	Der durchgeführte Test liefert keine signifikanten Resultate und es können keine Schlüsse aus dem Resultat gezogen werden.
Format Versuchsdaten	Daten, die in den Versuchen gewonnen werden, lassen sich nicht weiter verwenden (falsches Format, ungenügende Resultatmenge)
Gerätemerkmale	Mobilgeräte lassen sich nicht oder ungenügend anhand der ausgesendeten Probe Requests unterscheiden
Geräteverhalten	Geräte des selben Typs/OS verhalten sich nicht immer gleich. Schwierig/Unmöglich, herauszufinden, welche Faktoren ein unterschiedliches Verhalten begünstigen
Hardwareverhalten	Hohe Komplexität, da sich OS je nach unterliegender Hardware verschieden verhalten können

Tabelle 50: Beschreibung zu Risikotabellen 48 und 49

---

## **Anhang H: Protokolle der Meetings**

---

Nachfolgend sind die Protokolle der einzelnen Meetings mit dem Betreuer, die besprochenen Pendenzen und die getroffenen Entscheidungen dokumentiert.

---

## **Meeting Vorbesprechung**

Ort 8.U25  
Datum 31.08.2020  
Zeit 09:00  
Betreuer Prof. Beat Stettler  
Teilnehmer Janik Schlatter, Mike Schmid

### **Agenda**

1. Erstbesprechung Bachelorarbeit

### **Entscheidungen**

1. Grobe Projektplanung

### **Nächste Termine**

Termin 22.09.2020

---

## Meeting Woche 2

Ort 2.103  
Datum 22.09.2020  
Zeit 11:00  
Betreuer Prof. Beat Stettler  
Teilnehmer Janik Schlatter, Mike Schmid

### Agenda

1. Projektplan
2. Abzugebene Dokumente
3. Experte & Gegenleser
4. Arbeitspaket
5. Aufgabenstellung
6. Beschaffung Mobilgeräte
7. Messkammer

### Entscheidungen

1. Die Rückmeldung für den Projektplan wird noch folgen. Es wird noch ein aktualisierter Projektplan zusammen mit den Technischen Risiken nachgereicht.
2. Die Studenten stellen eine Liste der abzugebenden Dokumente zusammen, welche vom Dozenten bestätigt wird.
3. Der Experte & Gegenleser sind momentan noch nicht definiert. Die Personen werden zu einem späteren Zeitpunkt bekannt gegeben.
4. Die Arbeitspaket werden, wo sinnvoll erstellt.
5. Es soll die Aufgabenstellung vom AVT übernommen werden.
6. Es wird noch abgeklärt welche Mobilgeräte zur Verfügung stehen / benötigt werden.
7. Das ICOM unterhält eine Antennenmesskammer. Die Studenten organisieren die Reservationen für die Messkammer selber.

---

### **Abzugebendes**

---

<b>Wer</b>	<b>Was</b>	<b>Bis</b>
Janik, Mike	Projektplan & Technische Risiken	22.09.2020
Janik, Mike	Liste abzugebende Dokumente	25.09.2020

### **Nächste Termine**

Termin 29.09.2020

---

## Meeting Woche 3

Ort	8.U25
Datum	29.09.2020
Zeit	10:00
Betreuer	Prof. Beat Stettler
Teilnehmer	Janik Schlatter, Mike Schmid

### Agenda

1. Abgabetermin der Bachelor Arbeit
2. Besprechung Risikoanalyse
3. Mobilgeräte-Analyse
4. Besprechung Dokument Struktur

### Entscheidungen

1. Die Abgabe der Bachelor Arbeit wird auf den 8 Januar 2021 datiert.
2. Es soll eine theoretische Mobilgeräteanalyse durchgeführt werden, bevor die Messungen beginnen.
3. In der Dokumentation soll eine Auflistung des Verhalten der Mobilgeräte als "Verhaltenskatalog" erstellt werden.
4. Beim Prototyp sollen alle Überlegungen und Architektur-Entscheidungen dokumentiert werden, auch wenn diese nicht umgesetzt wurden.
5. Die vorgeschlagene Liste der abzugebenden Dokumente wurde mit geringen Änderungen angenommen.

---

### **Abzugebendes**

---

<b>Wer</b>	<b>Was</b>	<b>Bis</b>
Janik, Mike	Grobe Messeplanung	02.10.2020
Janik, Mike	Analyse	02.10.2020
Janik, Mike	Dokument Struktur mit Anpassungen	02.10.2020

---

### **Nächste Termine**

Termin        05.10.2020  
Kommentar    Janik Schlatter wird voraussichtlich per Teams am Meeting teilnehmen.

---

## **Meeting Woche 4**

Ort            8.U25 (Teams-Meeting)  
Datum        05.10.2020  
Zeit           14:30  
Betreuer      Prof. Beat Stettler  
Teilnehmer    Mike Schmid

### **Agenda**

1. Besprechung Messaufbau
2. Terminabsprache Meeting mit Experte und Gegenleser
3. Verhalten während Abwesenheit B.Stettler

### **Entscheidungen**

1. Zwei weitere Tests sollen den Messungen hinzugefügt werden.
2. Der vorläufige Termin für das nächste Meeting ist der 22.10.2020.

### **Nächste Termine**

Termin        22.10.2020  
Kommentar    Potentiell Meeting mit Experte und Gegenleser.  
Termin        30.10.2020  
Kommentar    Definitiver Termin mit Experte und Gegenleser.

---

## **Meeting Woche 7**

Ort	Teams-Besprechung
Datum	30.10.2020
Zeit	09:30
Betreuer	Prof. Beat Stettler
Teilnehmer	Janik Schlatter, Mike Schmid

### **Agenda**

1. Besprechung Zwischenstand
2. Abmachen Termin für Zwischenpräsentation

### **Entscheidungen**

1. Die Zwischenpräsentation mit Experte und Gegenleser wird am 05.11.2020 per Teams durchgeführt.
2. Die Android Messungen werden um eine Woche verlängert.

### **Nächste Termine**

Termin 05.11.2020 - Zwischenpräsentation

---

## Meeting Woche 8

Ort	Teams-Besprechung
Datum	05.11.2020
Zeit	10:00
Betreuer	Prof. Beat Stettler
Experte	Martin Willi
Gegenleser	Claudio Fuchs
Teilnehmer	Janik Schlatter, Mike Schmid

### Agenda

1. Zwischenpräsentation mit Experte & Gegenleser
2. Präsentation Ergebnisse aus iOS-Messungen

### Entscheidungen

1. Es sollen Hypothesen für Fingerprinting-Verfahren formuliert werden.
2. In der Vorarbeit wurde eine Tabelle mit den häufigsten vorkommenden MAC-Adressen erstellt. In den Messungen aufgezeichnete MAC-Adressen sollen damit verglichen werden.

### Abzugebendes

---

Wer	Was	Bis
Janik, Mike	Ansätze für Fingerprinting	19.11.2020

---

### Nächste Termine

Termin 19.11.2020

---

## **Meeting Woche 10**

Ort	Teams-Besprechung
Datum	19.11.2020
Zeit	15:00
Betreuer	Prof. Beat Stettler
Teilnehmer	Janik Schlatter, Mike Schmid

### **Agenda**

1. Zwischenbesprechung mit Betreuer

### **Entscheidungen**

1. Für das nächste Meeting sollen Ansätze für den Prototypen formuliert werden.

### **Abzugebendes**

<b>Wer</b>	<b>Was</b>	<b>Bis</b>
Janik, Mike	Ansätze für Fingerprinting	19.11.2020

### **Nächste Termine**

Termin 26.11.2020

---

## **Meeting Woche 11**

Ort	Teams-Besprechung
Datum	26.11.2020
Zeit	10:00
Betreuer	Prof. Beat Stettler
Experte	Martin Willi
Teilnehmer	Janik Schlatter, Mike Schmid

### **Agenda**

1. Besprechung Zwischenstand nach Analyse.
2. Besprechung der Ansätze

### **Entscheidungen**

1. In der Dokumentation soll das "Big Picture" ersichtlich sein.

### **Nächste Termine**

Termin 03.12.2020

---

## **Meeting Woche 12**

Ort	Teams-Besprechung
Datum	03.12.2020
Zeit	09:00
Betreuer	Prof. Beat Stettler
Experte	Martin Willi
Teilnehmer	Janik Schlatter, Mike Schmid

### **Agenda**

1. Zwischenbesprechung mit Betreuer & Experte

### **Entscheidungen**

1. Die Checkliste für die abzugebenden Dokumente soll an den Gegenleser gesendet werden.
2. Es sollen Konzepte für weiterführende Arbeiten formuliert werden.

### **Nächste Termine**

Termin 17.12.2020

---

## **Meeting Woche 14**

Ort	Teams-Besprechung
Datum	17.12.2020
Zeit	09:00
Betreuer	Prof. Beat Stettler
Experte	Martin Willi
Teilnehmer	Janik Schlatter, Mike Schmid

### **Agenda**

1. Demo Prototyp
2. Besprechung Projektabschluss

### **Entscheidungen**

1. Die Abgabe der Bachelorarbeit wird auf den 05.01.2021 datiert.
2. Die Präsentation soll am 12.01.2021 stattfinden.

### **Abzugebendes**

---

<b>Wer</b>	<b>Was</b>	<b>Bis</b>
Janik, Mike	Projektdokumentation	05.01.2021

---

### **Nächste Termine**

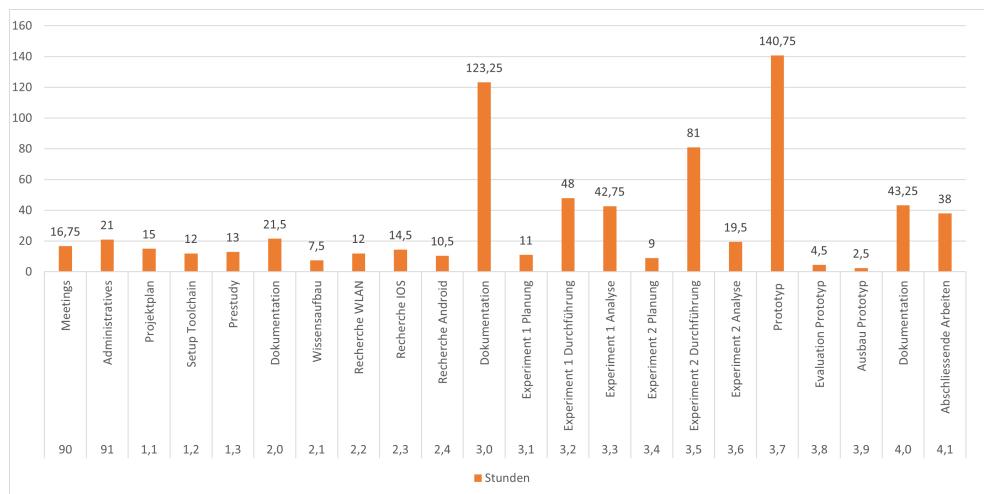
Termin 26.11.2020

---

## Anhang I: Zeitauswertung

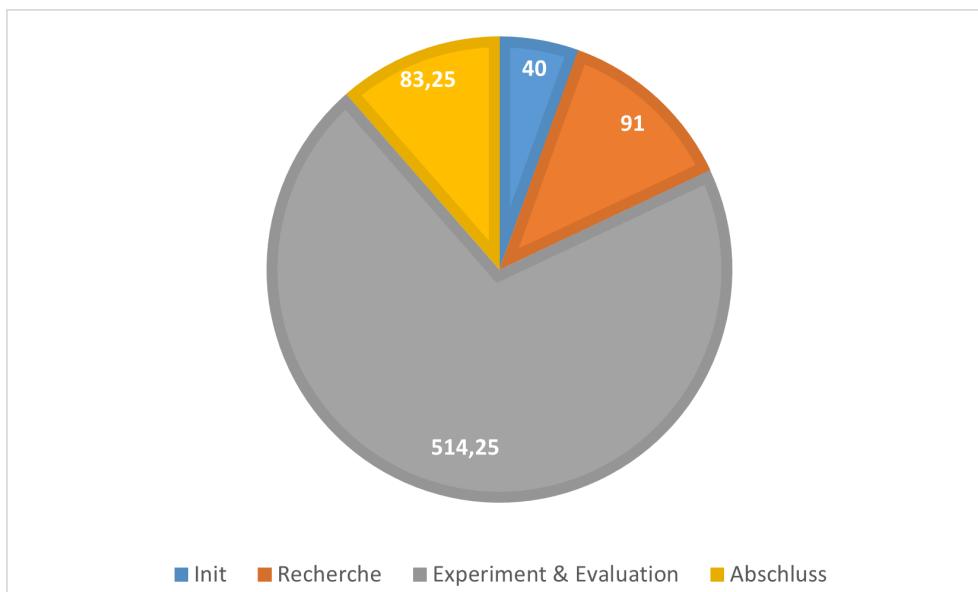
---

### Aufwand nach Arbeitspaket

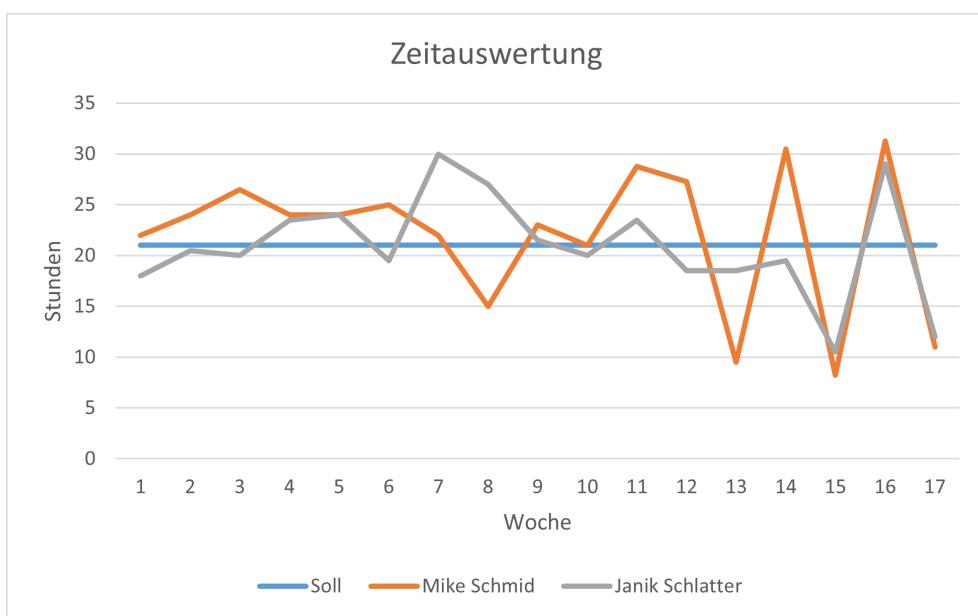


---

## Aufwand pro Phase



## Aufwand pro Woche



---

## **Anhang J: Eigensändigkeitserklärung**

---

Die Autoren erklären mit der Unterschrift,

- dass die vorliegende Arbeit selber und ohne fremde Hilfe durchgeführt wurde, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt ist oder mit dem Betreuer vereinbart wurde.
- dass sämtliche verwendeten Quellen erwähnt und gemäss den gängigen wissenschaftlichen Zitierregeln korrekt angegeben wurden.
- dass keine durch Urheberrecht geschützten Materialien (z.B. Bilder) in dieser Arbeit in unerlaubter Weise verwendet wurden.

Ort, Datum:

Ort, Datum:

Name, Unterschrift:

Name, Unterschrift:

---

## **Anhang K: Urheber- und Nutzungsrechte**

---

---

## **Vereinbarung Urheber-/ Nutzungsrechte**

### **1. Gegenstand der Vereinbarung**

Mit dieser Vereinbarung werden die Rechte über die Verwendung und die Weiterentwicklung der Ergebnisse der Bachelorarbeit "Mobile Fingerprinting" von Janik Schlatter und Mike Schmid unter der Betreuung von Beat Stettler geregelt.

### **2. Urheberrecht**

Die Urheberrechte stehen den Autoren zu.

### **3. Verwendung**

Die Ergebnisse der Arbeit dürfen sowohl von den Autoren, von der OST (ehemals HSR), sowie vom INS Institut for Networked Solutions nach Abschluss der Arbeit verwendet und weiter entwickelt werden.

Rapperswil, den.....  
.....  
Die Studentin/ der Student

Rapperswil, den.....  
.....  
Die Studentin/ der Student

Rapperswil, den.....  
.....  
Der Betreuer/ die Betreuerin der  
Studienarbeit

---

## **Persönliche Berichte**

---

---

## **Janik Schlatter**

Ich war vorerst nicht komplett von dem Thema überzeugt, da ich noch nie eine Forschungsarbeit gemacht habe und eher Praktisch veranlagt bin. Jedoch ist das Thema inhaltlich spannend und es hat mich thematisch interessiert.

Zu Beginn hatten wir einige Probleme mit dem Projektmanagement, da wir keine Ahnung hatten wie ein Projektplan für eine Forschungsarbeit aussehen muss. Danach lief es jedoch reibungslos und wir konnten bald mit dem Testen der Mobilgeräten beginnen.

Das Testen der Mobilgeräte in der Messkammer war sehr interessant, jedoch wurden wir durch Corona in der Benutzung der Kammer ein wenig eingeschränkt. Nach der ersten Testwoche waren einige unserer Hoffnungen erloschen, da das neue iOS Betriebssystem ihre Privacy im Griff hat. Nach den Android Messung wurden dann zum Glück doch noch ein Paar Ansätze ersichtlich wie wir die Sache angehen könnten.

Die Entwicklung des Prototyps war spannend, wir mussten bereits zu Beginn einige Entscheidungen treffen, beispielsweise wie wir die Daten aus dem Wireshark in das Python Programm bringen. Als wir dann zum Proof-of-Concept kamen mussten wir einige Filter-Methoden jedoch verwerfen, da sie keine anständigen Resultate lieferten.

Die Zusammenarbeit mit Mike Schmid verlief reibungslos, da wir bereits an einigen Projekten an der HSR zusammengearbeitet haben und wir uns in unseren Stärken sehr gut ergänzen. Zusätzlich stand uns Beat Stettler stets mit Hardware und professioneller Beratung zur Verfügung.

Abschliessend sind wir leider nicht zu einem Fingerprinting gekommen und ich denke nicht, dass es realistisch ist in Zukunft nur durch Probe-Requests ein Fingerprinting zu erreichen, da die neuen iOS und Android Versionen die Sicherheit generell immer verstärken. Trotzdem bin ich mit unserer Arbeit sehr zufrieden und wir konnten dennoch einiges erreichen und lernen.

---

## **Mike Schmid**

Da ich mich für Privacy- und Security-Themen interessiere, konnten wir mit der Zuteilung der Bachelorarbeit eine Arbeit durchführen, die in meinem Interessengebiet liegt.

Zum Thema "Mobile Fingerprinting" finden sich viele Paper und wir konnten uns in der Recherche phase einen guten Überblick verschaffen, welche Ansätze zielführend sind, und welche im Anbetracht der neuen Android/iOS-Betriebssystemversionen wahrscheinlich nicht angewandt werden können. Die breite Unterstützung durch den Betreuer, das INS, das IFS und das ICOM ist mir sehr positiv Aufgefallen und meiner Meinung nach wäre eine Durchführung der Messungen ohne diese Unterstützung nicht so angenehm verlaufen.

Die grössten Probleme traten mit der Projektplanung auf. Da weder Janik Schlatter noch ich jemals eine Forschungsarbeit betrieben haben, konnten wir uns auf kenerlei bestehende Erfahrungen stützen. Dadurch haben wir uns vor allem im Aufwand für die Arbeitspakete oft verschätzt und mussten im Verlauf der Arbeit den Meilenstein nach den Android-Messungen verschieben, da der zeitliche Aufwand für die Messdurchführung unsere schätzungen deutlich übertroffen hat.

Die Zusammenarbeit mit Janik Schlatter war sehr angenehm. Wir haben bereits gemeinsam einige Projekte an der Hochschule durchgeführt und zu Beginn der Bachelorarbeit waren wir bereits so gut aufeinander Abgestimmt, dass während der Arbeit keine Probleme aufgetreten sind. Weiterhin haben wir uns sehr gut in unseren Stärken/Schwächen und den jeweiligen Interessen für die einzelnen Tätigkeiten ergänzt. Janik, eher der praktisch veranlagte Mensch, konnte sich um die Durchführung der Messungen und Umsetzung des Prototypen kümmern, während ich die Organisation und den theoretischen Input in der Umsetzung liefern konnte. Auch in den gemeinsamen Programmiersessionen konnten wir effizient die anstehenden Probleme lösen und die in den Messungen gelernten Fakten praktisch umsetzen.

Mit den Ergebnissen bin ich weitestgehend zufrieden. Gerne hätte ich ein Fingerprinting und Tracking der Mobilgeräte praktisch umsetzen können, aber dadurch, dass die Verschleierung der MAC-Adressen in den neueren Betriebssystemversionen eine Unterscheidung von Geräten fast unmöglich macht, konnten wir nach bestem Wissen keinen funktionierenden Ansatz umsetzen. Allerdings bin ich aus Privacy-Sicht auch froh, dass Hersteller die Technologien weiterentwickeln und dadurch die Privatsphäre der Benutzer besser schützen können.