
# Was ist im WLAN-Funkverkehr trotz SICHERHEIT für alle zugänglich und was können uns diese ausgelesenen Daten sagen und nützen?

Kantonsschule Solothurn

Simon Studer
Klasse: N11w
Betreuer: Vincent Tscherter
Schuljahr 2014/2015

# Inhaltsverzeichnis

# Vorwort

Ich habe seit unsere Familie einen ersten Laptop besass ein Interesse an Computern gewonnen. Auch zuvor wollte ich möglichst viele technische Vorgänge verstehen und mit dieser elektronischen Maschine kam ein neues Gebiet dazu. Mit den Interaktionsmöglichkeiten Maus und Tastatur in der realen Welt konnte man digitale Prozesse steuern, die Aufgaben für den Benutzer erleichtern oder abnehmen. Durch das vermehrte in Kontakt kommen mit elektronischen Geräten und Themen stieg mein Interesse bis jetzt im Gebiet der Informatik und Elektrotechnik besonders an und ich wollte dementsprechend auch diese Arbeit über einen Teilbereich davon schreiben.

Deswegen kam ich nach anfänglicher Themensuche für meine Maturaarbeit mit ein paar Ideen zu meinem Robotik-Freikurs Lehrer Herr Vincent Tscherter. Nach einigen Gesprächen mit ihm hatte ich auch einige Vorschläge von ihm, die mir teils gefielen. Einer davon war WLAN-Analyse. Ich kannte, wie mittlerweile die meisten Menschen, WLAN aus meinem täglichen Leben, aber habe mich noch nie ausgiebig darüber informiert, wie drahtlose Verbindungen damit hergestellt werden können und danach Datenaustausch möglich ist. Nach ersten Recherchen stellte sich heraus, dass es insgesamt ein sehr komplexes Thema ist mit all den Themen die dazugehören, jedoch das ganze in kleineren Teilen betrachtet durchaus geeignet für eine Arbeit sein könnte. Besonders den Teil, der mir Herr Tscherter vorschlug ist ein aktuelles Thema. Es werden mit kleinen WLAN-Abhörgeräten (Sniffer) oder durch Protokolldateien von WLAN-Betreiber nachvollzogen, wann sich jemand mit seinem Smartphone wo aufhält. Es hörte sich nach modernsten Verfahren an, die mich wegen ihren Techniken faszinieren.

Ein Verständnis dafür zu haben war sehr verlockend für mich und würde mir auch einen Einstieg in das gesamte Thema WLAN bieten. Es bot sich ausserdem an, weil ich kurz davor einen Mikrocontroller namens Arduino Yun aus dem Internet bestellt habe, der auch für WLAN- und Internetverbindungen ausgelegt ist und somit auch für dieses Projekt geeignet ist.

Für die Hilfe bei der Themensuche und begleitenden Unterstützung will ich mich herzlich bei Herrn Tscherter bedanken. Bei der Umsetzung eines später erläuterten Versuchs musste ich die Erlaubnis dazu bei dem IT-Leiter der Kantonsschule Solothurn Herrn Thomas Vogt einholen. Ich danke auch ihm für das baldige Interesse in diese Arbeit und das Ermöglichen meines praktischen Versuchs.

# 1 Einleitung

Die meisten Menschen begegnen heutzutage dem Begriff „WLAN“ oder auch „Wi-Fi“ alltäglich, meist ohne die genauere Bedeutung und Technik dahinter zu kennen. Viele besitzen heutzutage auch ein Smartphone, das WLAN-Verbindungen nutzen kann, was sehr geschätzt wird, da an vielen Orten bereits kostenloses WLAN angeboten wird, doch die meisten wissen nicht, was dahinter steckt und welche Gefahren sie vielleicht bei der Benutzung eingehen. Deshalb habe ich einen praktischen Versuch durchgeführt, der mir mehr sagen soll, was jemand wie ich mit leicht erhältlichen und simplen Mitteln über die übertragenen Daten herausfinden kann und für was dieser Dienst sonst noch gebraucht werden kann.

## 1. 1 Zur WLAN Technik

WLAN steht für „Wireless Local Area Network“ und bezeichnet ein drahtloses Netzwerk des Netzwerkstandards IEEE802.11. Es dient der drahtlosen Datenübertragung von einem Sender, meist einem Access Point oder auch AP genannt, zu einem Empfänger, dem Endgerät oder Client. Diese Standardisierungen wurden ab Februar 1980 vom internationalen „Institute of Electrical and Electronics Engineers“ (IEEE) eingeführt, um Datenübertragung, -sicherheit und -verwaltung zu vereinheitlichen. Dieses Institut hat für den Netzwerkbereich zahlreiche Standards festgelegt, wie zum Beispiel auch der Ethernet-Standard IEEE802.3 (Kabelanschluss) und der Bluetooth-Standard IEEE802.15.1.

Weil diese Standardliste eingeführt wurde und alle unter gleichen Voraussetzungen arbeiten, ist es auch leicht möglich, den Verkehr von einem Standard in einen anderen zu konvertieren. Dies erlaubt zum Beispiel auch den Anschluss von Wireless-Endgeräten mit WLAN, wie Smartphones, über ein entsprechend ausgerüstetes Modem zum Internetzugang über Kabelanschluss. [1]

Einrichtungen, die WLAN-Verbindungen verwenden, können unterschiedlich ausgelegt sein. Entsprechend den gewünschten Anforderungen gibt es einerseits Ad-hoc Netze, bei denen Verbindungen direkt zwischen Endgeräten aufgebaut werden, um direkt miteinander kommunizieren zu können, oder auch sogenannte Basisstationen oder Access Points, die im Infrastrukturmodus viele Endgeräte aufnehmen und auch verwalten können, um jedem dieser Geräte die gewünschten Daten zukommen zu lassen und als Schnittstelle zwischen mehreren weiteren Anschlüssen (kabelgebunden, wireless etc.)

dienen können. Auch wenn die Ad-hoc Vernetzungen genauso interessant sein können bezüglich der Fragestellung wie Anwendungen mit Access Points, ist ihre Erscheinung jeweils eher von kurzer Dauer und nicht immer lokal festgebunden, was nicht dem behandelten Thema entspricht. Es gibt Projekte, die sich tiefer mit dieser Technik für „freies WLAN“ befassen [2], jedoch gibt es sie nur selten und noch nicht in grossflächigen Gebieten. Beispiele dazu sind AirDrop auf iOS-Geräten und S Beam (änlich wie Android Beam) auf Android-Geräten. Beide Dienste dienen dem einfachen Datenaustausch zwischen zwei Smartphones.

In dieser Arbeit werden also die fest eingerichteten Access Points im Infrastrukturmodus behandelt, die meist als Brücke zwischen Kabel- und Funkübertragung arbeiten und den Client-Geräten den Zugang zum Internet ermöglichen.

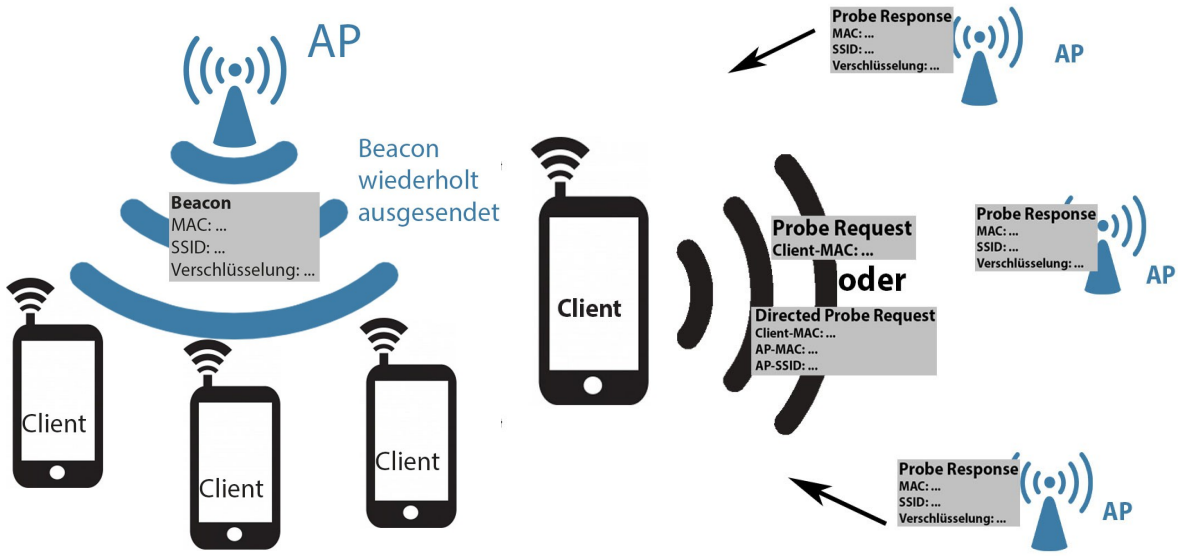
Unabhängig vom Modus, in dem die Geräte arbeiten, nutzen alle Techniken elektromagnetischen Funk, wobei dafür unterschiedliche Frequenzbändern um 2.4 Ghz und 5 Ghz verwendet werden. Das 2.4Ghz-Band ist aufgeteilt in 13 verschiedene Kanäle von 20 Mhz Bandbreite. All diese Kanäle dienen zur Verteilung der Access Points auf die Frequenzbänder, um Interferenzen zwischen den Funksignalen zu vermeiden oder zumindest zu verringern, da dies die Leistung und somit die Reichweite und Sendegeschwindigkeit beeinträchtig. Jede WLAN-Karte in Endgeräten und Access Points kann jeder dieser Kanäle benutzen, mit Ausnahme des 5GHz Bands, wenn dies nicht unterstützt wird. Mithilfe dieser Kanäle kann man also sich bezüglich der Ausstrahlungsflächen überschneidende Access Points trotzdem mit höchster Leistung anwenden, wenn man sie auf unterschiedliche Kanäle einstellt. [1]

## 1. 2 Datenaustausch im WLAN-Funknetz

Der folgende Abschnitt soll einen Einblick in die Kommunikationsweise zwischen Client-Geräten und Access Point geben.

Bevor die eigentliche Übermittlung von Daten stattfindet, müssen beide Seiten, Access Point und Client, eine stabile Verbindung zwischen ihnen aufbauen. Um das zu ermöglichen müssen sie erst gegenseitig voneinander wissen, dass sie in Reichweite sind. Dies kann auf zwei Arten geschehen: Einerseits kann jeder Access Point in wiederholenden Intervallen sogenannte Beacons aussenden. Diese Beacons enthalten die wichtigsten Informationen über den Access Point wie SSID (Netzwerkname), Verschlüsselungsart und bei manchen sogar ein Initialpasswort fürs erstmalige Verbinden. Diese Beacons werden als Broadcast auf dem vom Access Point benutzten Kanal an alle verfügbaren WLAN-Clients gesendet, falls der Access Point öffentlich bekannt sein soll, was üblicherweise der Fall ist. Wenn nun ein Client diese Beacons empfängt, kann er

dem Anwender die Verbindung mit diesem Access Point mit der übermittelten SSID anbieten. Der Client hat hier also nur passiv mitgehört, welche Access Points ihr Anwesen melden. Andererseits kann auch er Anfragen, sogenannte Probe Requests, auf allen Kanälen verschicken, sodass alle Access Points diese Anfrage empfangen können. Dabei kann diese Probe Request entweder als Broadcast ausgesendet werden, oder gerichtet an eine bestimmte SSID. Wenn nun der gewünschte oder die anwesenden Access Points diese Anfrage erhalten, senden sie eine dem Beacon ähnliche Probe Response zurück, die ebenfalls alle wichtigsten Daten zum Access Point enthält. Bisher erfolgte alles noch ohne Verschlüsselung der Daten und ist für alle WLAN-Geräte im Klartext sichtbar, was sich der folgende Versuch zu Nutze machen wird. [3]




*[4], [5]*

Wenn nun das Dasein beider Seiten sichergestellt ist, folgt nun die Authentifizierung des Clients und, falls erfolgreich, der Verbindungsaufbau. Es gibt verschiedene Arten wie eind Access Point feststellt, ob ein Client eine Verbindung herstellen darf. Es gibt immer noch viele ungesicherte, offene WLAN-Anbieter, bei denen kein Passwort notwendig ist. Der Client sendet hier eine Anfrage zur Verbindung und erhält ohne weiteres eine positive Antwort vom Access Point. Daneben gibt es noch mehrere Arten von Passwortgeschützen und verschlüsselten Methoden zur Verbindung. Bei allen sendet der Client sein vom Benutzer eingestelltes Passwort verschlüsselt mit und wenn dieses und das vom Access Point übereinstimmen, wird die Verbindung aufgebaut. Es gibt auch die Möglichkeit, dass jeder einzelne Client eine unterschiedliche Benutzername-Passwort Kombination

eingeben muss. Welche Art von Authentifizierung und Verschlüsselungsart vom jeweiligen Access Point angewandt wird, ist in den Beacons oder Probe Requests bereits als Information enthalten und der Client ist informiert. Zuletzt wird der Client auf Anfrage Assoziiert und eindeutig identifiziert, das heisst ihm wird vom Access Point eine freie und eindeutige IP-Adresse zugeteilt, damit der Access Point vom Client an einen Server angeforderte Daten wieder zum richtigen Client senden kann. Je nachdem findet ab jetzt der Datenaustausch gesichert statt und der Client kann die Verbindung, meist zum Internet, benutzen. [6],[7]

## 1. 3 WLAN-Tracking

Mit WLAN-Tracking bezeichnet man das Verfolgen von Client-Geräten (und somit auch den Trägern davon) mithilfe der auslesbaren Daten, die zwischen einem Access Point und Clients ausgetauscht werden. Man kann dies aktiv tun, um in der Nähe liegende Access Points mit einer Probe Request auf eine Antwort zu bitten und zusätzlich dessen Dasein zu erfahren, oder passiv, wobei man so keine Signale aussendet und nicht erkannt werden kann, jedoch versteckte Access Points, die ebenfalls nichts aussenden, unerkannt bleiben. Für passives Abhören benötigt man eine WLAN-Karte, die den Monitormodus erlaubt, um nicht ungewollt eigene Signale auszusenden und tatsächlich nur dem Funkverkehr zuzuhören. Da das Ziel in erster Linie war, Clients in Reichweite des Access Points zu erkennen, musste nicht aktiv gesucht werden. Auch so erhält man sofort alle Informationen darüber, mit welchem Access Point die Client-Geräte sich verbinden, denn das wird ebenfalls übermittelt. Das Verfolgen von einzelnen Client-Geräten ist möglich, da jedes WLAN-Gerät eine feste und weltweit einzigartige MAC-Adresse zur eindeutigen Identifizierung hat, mit der es sich beim Access Point meldet, wobei es nicht einmal verbunden sein muss. Solange der Standortdienst eines Geräts WLAN-Netze in der Umgebung zur Positionsbestimmung suchen darf, geschieht dies auch bei ausgeschalteter WLAN-Option auf Client-Smartphones im Hintergrund,. Dieses Verhalten kann unterdrückt werden und wird in Googles Android Betriebssystem „Erkennungsfunktion immer verfügbar“ unter erweiterten WLAN-Einstellungen aufgeführt. Auch in iOS und anderen Betriebssystemen ist eine solche Option verfügbar und ist bei vielen standardmässig eingeschaltet oder wurde durch andere Standortfunktionen automatisch eingeschaltet. [3]

WLAN-Tracking kann man nutzen, um zu erfahren, wie viele Personen sich ungefähr um einen Access Point befinden. Natürlich entspricht die Auswertung nicht der absoluten Anzahl, aber da viele Personen ein Smartphone mit eingeschalteter WLAN-Funktion bei sich tragen kann man sagen, wie viele Personen sich relativ zu anderen Zeitpunkten an

einem gewissen Ort etwa aufhalten. So können WLAN-Betreiber gewisse Prozesse optimieren, wenn sie anhand einer solchen Auswertung sehen, wann Ballungen von Personen entstehen und diese dann umleiten können. In Geschäften kann dies zur Kundenzählung benutzt werden, um beispielsweise die beliebtesten Einkaufszeiten erkennen zu können.[8]

Wenn man mehrere Access Points installiert hat, kann man ausserdem auswerten, wann sich ein Client ungefähr bei welchem Access Point befindet, da ein WLAN-Gerät automatisch das stärkste verfügbare Signal nutzen will und dazu zwischen Access Points mit gleicher SSID wechselt. Zusätzlich lässt sich auch noch die ungefähre Distanz zum Empfänger ermitteln, wenn man die Signalstärke der Client-Geräte misst. Somit sieht man nicht nur, wann viele WLAN-Geräte im Umfeld sind, sondern auch wo sich zu einem Zeitpunkt mehr befinden und wo weniger. Ausserdem kann man für die einzelnen Clients Bewegungsprofile anlegen, wodurch man zum Beispiel beliebte Routen beim Einkauf in einem Kaufhaus erkennen kann und dementsprechend gewisse Produkte oder Werbung zur Kundenfreundlichkeit und Gewinnoptimierung umplatziert.

Eine andere Möglichkeit, diese Daten zu nutzen, ist die genauere Standortbestimmung. Dies benötigt eine Anwendung auf den Client-Geräten, die ständig die Signalstärken zu nahen Access Points misst. Wenn sie nun die exakten Standorte der Access Points kennt, kann sie mittels Trilateration und der Signalstärken der erreichbaren Access Points den Standort des Geräts ermitteln. Je mehr Signale von Access Points aus der Nähe empfangen werden, desto genauer kann der Standort berechnet werden. Da dazu nur die WLAN-Technik benötigt wird, kann man solche Systeme an Orten installieren, an denen GPS nicht verfügbar ist. Zusätzlich kann man die beiden kombinieren, um noch genauere Standortbestimmung zu ermöglichen, um auch in Fällen, in denen eines der beiden Systeme nicht vorhanden ist, eine ungefähre Position zu erhalten. So wird ebenfalls eine grössere Unabhängigkeit von einem einzelnen System und grössere Zuverlässigkeit der kombinierten Systems geboten. Die erwähnten Standortdienste auf Smartphones und gegebenenfalls anderen Geräten nutzen genau diese Kombination. Um die Position aller möglichen Access Points zu erlangen senden Firmen wie Google sogenannte WLAN-Sniffer aus, die aktiv nach Access Points suchen und mit der trilaterierten GPS-Position abspeichern. Das gleiche geht auch über ihre Android Kunden, deren Geräte bei eingeschaltetem GPS und WLAN die Position und MAC-Adresse der Access Points paarweise übertragen können. So erlangen sie globale Informationen über Positionen der Access Points um diese zweigeteilte Positionsbestimmung zu ermöglichen. [9], [10], [11]

## 1. 4 Leitfrage und Vorgehen beim Versuch

Die hauptsächliche Fragestellung dieser Arbeit lautet: „Was ist im WLAN-Funkverkehr trotz Sicherheit für alle zugänglich und was können uns diese auslesbaren Daten sagen und nützen?“ Dabei ist wie gesagt der Funkverkehr bei Access Points im Infrastrukturmodus gemeint.

Um herauszufinden, wie viel und was tatsächlich durch diese Kommunikation zwischen Access Point und Clients öffentlich bekanntgegeben wird, wurde ein Versuch durchgeführt. Er sollte zeigen, wann welche Geräte erfasst werden können und was ein solches Gerät von sich bekannt gibt. Durch statistische Auswertung sollen Verhaltens- und Bewegungsabläufe dargestellt werden und damit eventuell Anwendungen zur Weiterverwendung der Daten in der Praxis erkannt werden.

Der Versuch sollte über mehrere Tage stattfinden, um mindestens das tägliche Verhalten der Nutzer aufzeichnen zu können. Dabei sollte nicht nur ermitteln werden, welche Geräte während des ganzen Versuchs aufgetaucht sind, sondern ein weiteres Ziel war einen zeitlichen Ablauf erstellen zu können, wann welches Gerät etwas von sich bekannt gibt in Form von Probe Requests oder Authentifizierungen. So würden sich die Daten auch einfach verständlich in einem Diagramm darstellen lassen, das eine Zeit-Achse hat und auf der zweiten Achse beliebige Daten aufgeführt werden können.

Erwünschte Daten, die analysiert werden sollen, sind bei den Access Points die MAC-Adresse, deren SSID, der benutzte Kanal, die Verschlüsselung und die ausgesendeten Beacons pro Intervall. Bei den Clients sollten die MAC-Adresse, die mit direkten Probe Requests aufgerufenen SSIDs, die ausgetauschten Datenpakete und die MAC-Adresse, mit dessen Access Point sie momentan verbunden sind erfasst werden, ebenfalls pro Intervall.

Da kein Zugriff auf ein Netzwerk aus Access Points zur Verfügung steht, muss ein passives Gerät, hier ein Arduino, als Zuhörer den Funkverkehr analysieren. Für Kundentracking mit der Verfolgung derer ungefähren Position wäre eine grosse Anzahl an Arduinos nötig, um das Versuchsareal flächendeckend auf WLAN-Signale abzuhören. Weil nur einer eingesetzt wurde, konnte nur erfahren werden, wann welche Clients in Reichweite des Arduinos sind. Kundentracking wäre auch recht komplex darzustellen und könnte wohl nicht in der vorhandenen Zeit umgesetzt werden. Weiter sollte nicht aktiv nach Access Points gesucht werden, denn bei dieser Untersuchung sollte das Anwesen eines Mithörers nicht jedem Client bekannt sein. Auch könnte man bei aktiver Suche das Ergebnis verfälschen, da nicht die Informationen aus den Client-Geräten verlangt werden sollen, sondern aufzeigen, was sie von selbst wiederholt aussenden.

# 2 Hauptteil

## 2. 1 Materialien für den Versuch

Bereits als erstes musste nach bereits bestehenden Hilfsmittel zur Erfassung von WLAN-Siganlen umgesehen werden, um die Machbarkeit sicherstellen zu können. Es gibt eine grosse Zahl an Softwarelösungen, die diese Aufgabe umsetzen können, doch eine Voraussetzung war, dass nur ein kleines, WLAN fähiges Gerät verwendet werden soll und nicht ein Laptop eine Woche lang auf dem Schulareal liegen und laufen gelassen werden muss.

### 2. 1. 1 Arduino Mikrocontroller

Bei der Suche nach Mikrocontroller im Internet stösst man schnell auf den Arduino mit Modellnamen Yun. Es gibt auch noch viele andere Mikrocontroller, doch die Arduino-Produktserie ist auch für Anfänger im Programmieren gedacht und lässt sich für vieles einsetzen.


[12] *Abbildung 1: Arduino Yun*

Der Yun ist ein Arduino-Gerät aus einer ganzen Serie, doch nur er kann mit WLAN und Internetanwendungen umgehen. Die Vorteile dieses Mikrocontrollers sind viele: Er ist sehr klein und kann mit einem USB-Anschluss und entsprechendem 5V-Adapter an jede 230V-Steckdose angeschlossen werden. Zum programmieren entfernt man den Adapter wieder und man kann ihn mit einem USB Typ A Stecker an einen Computer anhängen. Von Arduino wird eine Programmierplattform Arduino IDE angeboten, die das Programmieren mit der Processing-Programmiersprache erleichtert und das laden und starten von Programmen auf den Arduino ermöglicht. Dabei aufzupassen ist, dass Mikrocontroller allgemein eher wenig Speicher und Arbeitsspeicher besitzen und so nicht sehr grosse Programme mit vielen Variablen ausführen können, doch für den bevorstehenden Versuch reichten diese.

Um den internen Speicher zu erweitern kann man eine microSD Karte einsetzen, auf die man dann auch bequem von einen Laptop oder Computer aus zugreifen kann. Der Yun besitzt nebst dem Arduino-seitigen Prozessor (Atmega32u4), den auch andere Arduino-Platformen benutzen, einen zweiten Prozessor (Atheros AR9331), der das Betriebssystem openWrt-Yun besitzt. Es ist ein Linux Betriebssystem, das auch ähnlich auf Access Points angewandt wird und somit gut geeignet ist. Zwischen den beiden Prozessoren kann über

eine Arduino eigene Bridge-Library kommuniziert werden. Da nur der Linux-Prozessor direkten Zugriff auf die WLAN-Funktion hat, muss sicherlich dieser benutzt werden, um den WLAN-Funkverkehr auslesen zu können. Da jedoch der Linux-Prozessor kein Programm automatisch ausführt (was aber auch machbar wäre), wurde der Arduino-Prozessor benutzt, der das geladene Programm automatisch startet, um Skripte und Programme auf dem Linux-Prozessor zu starten und zu kontrollieren. [13]
Der Arduino Yun ist erhältlich auf store.arduino.cc für 62.51 Franken.
Er wurde per Yun sysupgrade[14] auf Version openWrt-Yun 1.5.3 geupdated.
Die benutzte Version der Programmierplattform ist Arduino IDE 1.5.6-r2.

### 2. 1. 2 Aircrack-ng

Das Überwachen des WLAN muss der Linux-Prozessor übernehmen, weshalb ein Programm für Linux gesucht werden muss, das nicht zu gross ist und gerade ein paar wenige Anforderungen erfüllt. Programme zum Abhören der WLAN-Frequenzen gibt es viele, die alle unterschiedliche Funktionen ermöglichen. Für diese Aufgabe reicht es, wenn das Programm die Ausgabe der gesammelten Daten in eine Datei speichert oder sie direkt über die Bridge auf dem Arduino zum Arduino-Prozessor übermitteln kann. Weiter soll der Monitormodus der WLAN-Karte verwendet werden, um passiv mithören zu können. Auch soll das Programm zum Abhören durch die Linux-Shell aufrufbar sein, die auf der Linux-Seite Befehle über die Bridge entgegennimmt und ausführt. Die Linux-Shell ist auf allen Linux-Versionen mit dabei und muss nicht auch noch bezogen werden.
Bei der Suche im Internet stösst man unter anderen auf Aircrack-ng. Es ist eine Programmsammlung, die Angriffe auf Access Points und Clients, errechnen von schlechten WEP-Passwörtern und das Abspeichern von Verläufen des Funkverkehrs erlaubt. Die letzte Funktion ist die hier gesuchte, wobei auch der Monitormodus dazu bereitgestellt wird. Airmon-ng ist der Teil der Programmsammlung, der den Monitormodus kontrolliert, und Airodump-ng der, der WLAN-Packete analysiert und MAC-Adressen und weitere Informationen in einer Log-Datei festhält.
Die benutzte Aircrack-ng Version ist 1.1, die über den in openWrt enthaltenen OPKG Programm Manager heruntergeladen werden kann. [15]

### 2. 1. 3 Lua

Ursprünglich wurde der Arduino so programmiert, dass auch die Auswertung gleich während der Durchführung gemacht wird, wozu die Programmiersprache Lua benutzt wurde. Nach einem später erläuterten Problem übernahm Lua auf dem Arduino nur noch das Ordner erstellen und durchnummerieren wenn ein neuer Versuch erneut durchgeführt

wird.
Da die Auswertung im finalen Programm nicht mehr während des Versuchs gemacht wird, muss sie auf einem Computer nach dem Versuch erfolgen. Dazu wurde Lua for Windows verwendet, das die geschriebenen Skripte ausführen kann.
Ich habe Lua als bevorzugte Programmiersprache auf der Linux-Seite des Yun gewählt, da sie sehr leicht erlernbar und simpel ist, ich bereits viele Programme in dieser Sprache geschrieben hat und sie ausserdem bereits auf dem Arduino Yun installiert war. Eine Alternative wäre Python gewesen, was ebenfalls von openWrt zur Verfügung gestellt wird und auch sehr simpel ist, jedoch hatte ich sie noch nie benutzt und wollte sie nicht erst noch erlernen müssen.
Die auf dem Arduino Yun benutzte Version ist Lua 5.1.4 und war bereits installiert.
Auf dem Computer wurde zur Analyse Lua for Windows 5.1.4 benutzt, die auf Google Code heruntergeladen werden kann. [16]

## 2. 2 Das Programm

Von Anfang an musste mit der Funktionsweise von den Befehlen der Aircrack-ng Programmsammlung auseinandergesetzt werden, um sie in ein Programm einzubauen. Die Befehle, die auf dem Linux-Prozessor aufgerufen werden müssen, sind folgende:

```
airmon-ng start wlan0
```

Dieser Schaltet die WLAN-Karte in den Monitormodus um. wlan0 sagt dabei, welches Interface umgestellt werden soll, was in diesem Fall das erste WLAN-Interface des Arduino Yun ist. Durch diesen Befehl wird ein neues Interface mit dem Namen mon0 erstellt, das dem Benutzer erlaubt, den WLAN-Funkverkehr abzuhören. `Airmon-ng stop wlan0` oder `airmon-ng stop mon0` beendet diesen Modus wieder auf diesem Interface.

```
airodump-ng -w outputFile.csv wlan0 &
```

Airodump-ng speichert alle aus dem WLAN gesammelten Daten in der Datei, die nach -w angegeben wird. wlan0 bezeichnet wiederum das zu benutzende Interface und & stellt sicher, dass Linux den Befehl im Hintergrund ausführt, um folgende Befehle direkt ausführen zu können und nicht auf die Beendigung des aktuellen warten zu müssen.
Die von Airodump-ng erstellte Ausgabedatei kann in verschiedenen Dateiformaten ausgegeben werden. Das .csv-Dateiformat wurde gewählt, da es sich mit Lua später leicht auslesen und bearbeiten lässt. Diese Datei kann auch übersichtlich in Excel oder ähnlichen Tabellenprogrammen dargestellt werden. Folgende Abbildung ist ein Beispiel einer solchen Ausgabedatei in Open Office dargestellt:

| BSSID | First time seen | Last time seen | channel | Privacy | Authenticatior | # beacons | LAN IP | ESSID |
|---|---|---|---|---|---|---|---|---|
| 1C:6A:77:03:84:03 | 2014-09-30 17:23:58 | 2014-09-30 17:24:42 | 1 | OPN | | 13 | 172. 23.144.2 | public-ksso |
| 84:1B:C5:E3:FF:A7 | 2014-09-30 17:24:41 | 2014-09-30 17:24:41 | 6 | WPA2 | PSK | 2 | 0. 0. 0. 0 | Martin Wlan |

| Station MAC | First time seen | Last time seen | # packets | BSSID | Probed ESSIDs |
|---|---|---|---|---|---|
| D5:A2:A0:43:01:DA | 2014-09-30 17:23:54 | 2014-09-30 17:24:37 | 406 | 90:BB:51:82:A3:54 | Arduino Yun-A4B2FF812C24 |
| A3:42:22:F1:32:56 | 2014-09-30 17:23:55 | 2014-09-30 17:23:55 | 1 | (not associated) | public-ksso QL-27395 |

*Abbildung 2: Beispielausgabe des Befehls airodump-ng (verfälschte MAC-Adressen)*

Die für diese Arbeit weniger interessanten Informationen wurden hier ausgelassen. Man sieht jedoch, dass die Access Points oben und die Clients oder Stationen unten dargestellt werden und ausserdem genügend Informationen da sind, um meine Aufgabe zu ermöglichen. Was noch fehlt ist die Möglichkeit die Informationen zeitabhängig darzustellen. Dazu musste man im eigentlichen Programm selbst sorgen, wobei hauptsächlich zwei Lösungen in Frage kamen:

Zum einen kann nach jeweils einem bestimmten sich wiederholenden Intervall airodump-ng gestoppt werden, indem der Prozess abrupt mit dem Linux-Befehl pgrep airodump-ng | kill beendet wird. Airodump-ng stellt keine Option zur Verfügung, die es von selbst stoppen lässt und so muss es mit diesem Befehl zum Beenden gezwungen werden, weil es sonst immer weiter laufen würde.

Bei jedem Intervalldurchgang einer gewissen Zeit wurde somit eine neue Datei erstellt, die aufsteigend durchnummeriert werden kann. Dies war die erste Idee und wurde gleich versucht umzusetzen. Der Programmablauf war somit recht simpel: Zuerst wird mit airmon-ng den Monitormodus eingeschaltet und dann in einer sich ständig wiederholenden Schleife airodump-ng gestartet, die Intervalldauer abgewartet, dann airodump-ng zum Beenden gezwungen und die Schlaufe wieder von vorne begonnen. Dabei sollten auch gleich die gesammelten Daten in einer separaten Datei ausgewertet werden, um sie eventuell über ein zweites WLAN-Interface im Access Point Modus herunterladen zu können und so die Zwischenstände und das zuverlässige Funktionieren geprüft werden könnten. Nach einigen Ausprobieren wurde herausgefunden, dass ein WLAN-Interface zwar zwei Access Points gleichzeitig betreiben kann, um zum Beispiel ein Gast-Konto einrichten zu können, jedoch nicht nebst dem Monitormodus auch noch ein Access Point im Infrastrukturmodus aufbauen kann. Bevor dies jedoch festgestellt wurde, wurde bereits ein Lua Skript geschrieben (createStats.lua auf der microSD Karte), das diese Auswertung übernimmt und bei jedem Durchlauf der Schleife im Programm einmal ausgeführt wird. Das ganze lief bis auf wenige Details wunderbar. Nun wurde begonnen dieses Programm ausgiebig zu testen und gewisse Fehler zu verbessern. Erst kurz vor dem eigentlichen Versuch wurden Tests über längere Zeiträume durchgeführt. Dabei wurde festgestellt, dass das Programm nach einer gewissen Anzahl Schleifendurchläufe

nicht mehr funktioniert und abbricht. Nach langem Debugging (Fehlersuche) des Codes wurde erkannt, dass das Problem bei der Zuweisung der Linux-Prozess ID oder PID von ausgeführten Befehlen liegen muss. Wenn ein Prozess auf Linux gestartet wird, was beim Befehl airodump-ng beispielsweise der Fall ist, wird ihm in einer Prozessliste eine eindeutige PID in Form einer aufsteigenden Nummer zugeteilt, um die Prozesse auseinanderhalten zu können. Wenn ein Prozess wieder gestoppt wird, wird er aus der Prozessliste wieder entfernt und die PID freigegeben. Wenn die PID Nummer bei einer gewissen Grenze (abhängig vom Betriebssystem) ankommt, beginnt das Betriebssystem wieder von unten nach freien PID Nummern zu suchen.
Bei der abrupten Art den airodump-ng Prozess zu beenden geschah dies scheinbar nicht. Da bei jedem Schleifendurchgang wieder ein airodump-ng Prozess gestartet wurde, der noch weitere Hintergrundprozesse startete und PIDs besetzte, stieg die verwendete PID auf ungefähr 32'000, was bei diesem Linux auch zu erwarten war. Anschliessend jedoch fand es von unten wieder beginnend keine freien PID mehr. Durch den kill Befehl wurden wohl fälschlicherweise die PID nicht mehr freigegeben. Im Internet war kein ähnliches Problem und keine Antwort auffindbar, wie das Problem anzugehen wäre und eine zweite Variante wurde gesucht.
In der Ausgabedatei von airodump-ng gibt es die Spalte „last time seen“. Wenn der airodump-ng Prozess dauernd laufen gelassen wird, um das PID-Problem zu umgehen, kann nach jedem Intervall eine Kopie der Ausgabedatei erstellt werden, deren Namen auf den aktuellen Zeitpunkt gesetzt wird. Mithilfe dieser last-time-seen Spalte und der Benennung der Datei nach diesem Prinzip kann später nachvollzogen werden, ob der jeweilige Client oder Access Point während des letzten Intervalldurchgangs aktiv war oder nicht. Die restlichen Informationen können gleich wie bei der ersten Variante aus den kopierten Dateien ausgelesen werden. Doch wie die Auswertung stattgefunden hat wird im Kapitel 2.4 ausführlich beschrieben.
Ein weiteres Problem ist die spärliche Ausrüstung an Hardwarekomponenten auf dem Arduino. Er besitzt keine Real Time Clock (RTC), die die reale Zeit speichert und aktualisiert. Wenn der Arduino ausgeschaltet wird, das heisst wenn er von der Stromquelle getrennt wird, dann stoppt auch seine Uhr und beginnt beim nächsten Einschalten von dieser, nun falschen Zeit an weiterzulaufen. Der Arduino ist somit auf externe Hilfe angewiesen, entweder mit zusätzlicher Hardware oder Software. Da die Auswertung zeitabhängig werden sollte und die Zeit auch mit der tatsächlichen Zeit übereinstimmen sollte, musste eine Lösung gefunden werden. In einer Anleitung im Internet [17] wird ein Beispiel beschrieben, bei dem der Arduino im Access Point Modus

die Browserzeit eines verbundenen Clients nutzt, um die Zeit zu synchronisieren. Diese Idee wurden nun in die Umsetzung miteinbezogen.

Der grobe Programmablauf ist bei der finalen Programmversion wie folgt:

Nachdem der Arduino vollständig aufgestartet ist, was bei der langsameren Linux-Seite mehr als 30 Sekunden dauern kann, baut er ein Access Point auf, mit dem nun ein Client verbunden werden kann. Der Arduino durchläuft eine Schlaufe und überprüft in dieser ständig, ob ein Client da ist und Befehle übergibt. Wenn der Client in seinem Browser die Seite 192.168.240.1/sd/index.html aufruft, erhält er Zugriff auf eine Website, die auf der microSD Karte im www Ordner gespeichert ist und diese Befehle enthält. Dieser automatische Aufruf vom www Ordner ist von Arduino so eingerichtet worden, um solche Projekte mit Zugriff auf den Speicher des Arduinos zu ermöglichen. Diese Website „Arduino Control“ hat die Option zur Synchronisierung der Zeit des Arduinos mit der Browserzeit und die Option zum Fortfahren im Programmablauf. Beide Optionen übertragen über den REST Dienst von Arduino ihre Befehle zu diesem, indem sie mit Hilfe der Programmiersprache Javascript (und der Library jQuery) neue URLs aufrufen, die der Arduino auslesen kann. Die erste Option überträgt den Befehl `sync`, gefolgt von der Anzahl vergangener Sekunden seit dem 1. Januar 1970 um 00:00 Uhr. Diese Art von Datumsangabe wurde bei der Einführung des Betriebssystems Unix ebenfalls eingeführt und dementsprechend Unixtime benannt. Sie dient dazu, die Zeit als eine Zahl darzustellen, was dem menschlichen Auge nicht viel aussagt, jedoch für den maschinellen Gebrauch eine Vereinfachung darstellt. In vielen Programmiersprachen und Libraries dazu wird diese Option ebenfalls angeboten. Mit dem Linux-Shell Befehl


*Abbildung 3: Website index.html in Google Chrome auf einem Smartphone*

```
date +%s -s @ seconds
```

kann die Uhrzeit auf dem Arduino dann auf die übertragene eingestellt werden, wobei für `seconds` diese Zahl an Sekunden (Unixtime) eingesetzt wird. Die zweite Option der Website überträgt den Befehl `continue`, der den Arduino veranlasst die bisherige Schleife der Client-Abfrage zu unterbrechen und im Programmcode weiterzufahren. Ein dritter, in der Webseite versteckter Befehl wird mindestens jede Sekunde automatisch aufgerufen und verlangt die momentan eingestellte Zeit auf dem Arduino. Die Rückgabe wird dann auf der Website wieder ausgegeben und dient der Kontrolle der eingestellten

Zeit.
Wenn der continue Befehl erfolgte, schaltet der Arduino das WLAN vom Access Point zum Monitormodus. Anschliessend wird ein neuer Ordner für die Ausgabedatei und deren Kopien auf der microSD Karte erstellt und der Befehl

```
airodump-ng -w outputFile.csv wlan0 &
```

ausgeführt. Er startet nun das Auslesen und speichert die Information in der Datei outputFile.csv. Wenn eine Intervalldauer vorübergegangen ist, was in einer Schleife ständig überprüft wird, wird mit

```
cp outputFile.csv copiedOutputFile-time.csv
```

eine Kopie der Ausgabedatei von airodump-ng erstellt. Der Name dieser kopierten Datei enthält die Zeit, bei der sie erstellt wird und dies einfachheitshalber wiederum im besprochenen Unixtime-Format. Solange eine Variable `sniffing` nicht auf ungleich Null gesetzt wird, wird das Durchlaufen des Intervalls wieder gestartet und auf dessen Ende gewartet, um die Datei dann erneut zu kopieren. Während der Versuchsdurchläufe dieser Arbeit wurde die erwähnte Variable nicht verändert, damit das Aufzeichnen bis zur Trennung des Arduinos von der Steckdose dauert. Es wurde in vorangegangenen Programmversionen verwendet, bei denen das Stoppen theoretisch einfacher ist. Hier ist es nun aber bereits ein Ansatz, wie das Programm in Zukunft erweitert werden könnte, weswegen es im Code gelassen wurde.
Bei der Auswertung der Ausgabedateien wurde ein neues Problem des Programms erkannt. Wenn die Datei kopiert wird, kann es sein, dass Airodump-ng gerade in die Datei schreibt, da das Programm die Datei ständig für sich offen hält um live Updates aus dem Funkverkehr zu notieren. Die Datei wird also nicht nur während kurzen Zeitpunkten gespeichert, sondern ständig erneuert. Wenn die Datei nun kopiert wird, enthält sie fehlerhafte Informationen, was unbeendete, doppelte und veraltete Zeilen sein können, die nur mühsam manuell verbessert werden können. Im zweiten Versuch trat diese Problem sehr häufig auf und konnte nicht mehr rechtzeitig korrigiert werden.

Mehr Details zu den einzelnen Schritten im Programm sind als Kommentare im Code eingefügt, der auf GitHub auf https://github.com/Studi11/Maturaarbeit-WLAN.git in der Programmdatei analyseWlan_net_1.ino oder der Website index.html abrufbar ist.

## 2. 3 Versuchsdurchführung

Da zu statistischen Auswertungen eine grosse Grundmenge gebraucht wird, um möglichst aussagekräftige Feststellungen und Folgerungen zu erhalten, wurde ein WLAN-Netzwerk gesucht, das möglichst mehrere Access Points bereitstellt und ebenfalls möglichst viele Nutzer hat. Da erst kürzlich an der Kantonsschule Solothurn ein neues WLAN-Netzwerk eingerichtet wurde fiel es nahe, dass dies möglicherweise ein guter Standort ist. Dafür sprachen die grosse Anzahl an Schülern und Lehrer, die dieses WLAN täglich benutzen, die Anzahl und Platzierung der Access Points, die Erreichbarkeit, da der Autor selbst täglich dorthin zur Schule geht und das baldige Erlauben der IT-Zuständigen.

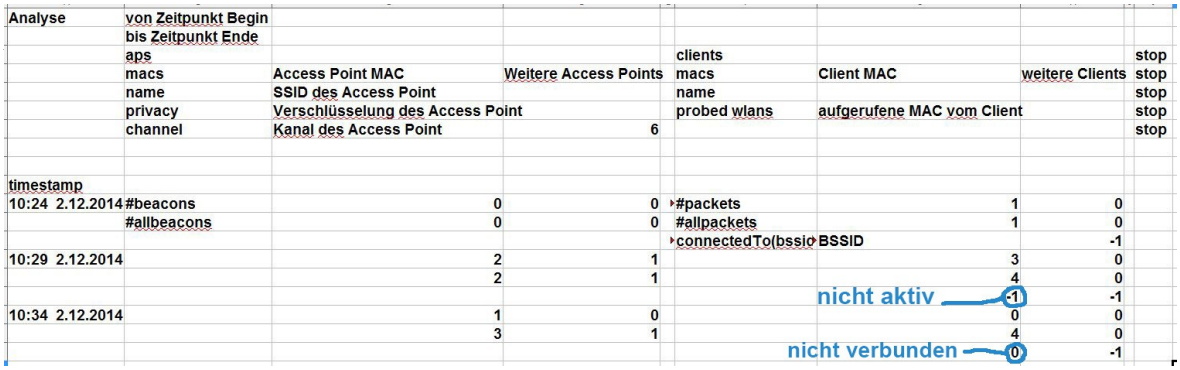
Das WLAN an der Kantonsschule besitzt viele Access Points, die alle in den Gebäuden verteilt sind. Da der Versuch nur mit einem Arduino an einem Standort durchgeführt werden soll, galt es gute Platzierungsmöglichkeiten zu suchen. Die Kriterien waren, dass an dem gewählten Ort eine grosse Menge an Schülern und Lehrer vorbeigeht und eventuell auch als Aufenthaltsort dient. Dies ist vor allem bei den Haupteingängen zu finden, wobei einer gleich neben der Kantine der Schule, der Mensa, ist und ein zweiter im Naturwissenschaftlichen Pavillon, der den Schülern Tische und Stühle bietet, an denen sie sich aufhalten können. Die weiteren Eingänge haben diese Aufenthaltsmöglichkeit nicht. Ausserdem kommen bei den zwei gewählten Standorten die meisten Schüler und Lehrer in die Schule, denn der im Pavillon ist über einem grossen Velokeller, in dem die meisten ihr Fahrrad versorgen, und beim anderen die Bushaltestelle „Kantonsschule“, welche viele zur An- und Abreise nutzen. Der Arduino benötigte ausserdem eine freie Steckdose zur Stromversorgung und sollte nicht frei im Raum stehen, damit er nicht sofort gesehen oder sogar gestohlen werden kann. In der Mensa befindet sich eine Pausenbar in der Mitte des Raumes und ist von Access Points umgeben. In dieser Pausenbar konnte der Arduino gut an eine freie Steckdose angeschlossen und verstaut werden, nachdem auch dort ein Erlaubnis der Mensabetreiber einzuholen war. Im Nawi-Pavillon gibt es anschliessend an den Aufenthaltsbereich Schulzimmer. In das anschliessende davon konnte der Arduino in einem Schrank mit Steckdosen platziert werden. Die Standorte waren damit gefunden und geplant war den Arduino je eine Woche da zu lassen.

Sobald der Arduino an Strom angeschlossen wird, kann man auf den neu erstellten Access Point warten, mit dem man sich dann mit einem Smartphone oder Laptop verbinden kann. Nach dem Aufruf der Arduino Control Website (192.168.240.1/sd/index.html) kann man vor dem Starten des Abhörens die Zeit synchronisieren. Wenn man gestartet hat, kann man das fehlerlose Funktionieren nur noch dadurch sehen, dass der Access Point wieder beendet wird.

Nachdem die gewünschte Zeit für den Versuch durch war, was mehreren Tagen entspricht, konnte der Arduino von der Steckdose entfernt werden, um an einem Computer die Daten auf der microSD Karte auszuwerten.

## 2. 4 Versuchsauswertung

Ein erwartetes Ergebnis ist das Erkennen von Verhaltensabläufen. Dazu müssen die zahlreichen Kopien der Ausgabedatei von aircrack-ng ausgewertet werden und die gesammelten Daten übersichtlich in einer weiteren Datei gespeichert werden. Diese soll wie in der Abbildung 7 organisiert sein.




| Analyse | von Zeitpunkt Begin | | | | | | |
|---|---|---|---|---|---|---|---|
| | bis Zeitpunkt Ende | | | | | | |
| | aps | | | clients | | | stop |
| | macs | Access Point MAC | Weitere Access Points | macs | Client MAC | weitere Clients | stop |
| | name | SSID des Access Point | | name | | | stop |
| | privacy | Verschlüsselung des Access Point | | probed wlans | aufgerufene MAC vom Client | | stop |
| | channel | Kanal des Access Point | 6 | | | | stop |
| | | | | | | | |
| timestamp | | | | | | | |
| 10:24 2.12.2014 | #beacons | 0 | 0 | #packets | 1 | 0 | |
| | #allbeacons | 0 | 0 | #allpackets | 1 | 0 | |
| | | | | connectedTo(bssid | BSSID | -1 | |
| 10:29 2.12.2014 | | 2 | 1 | | 3 | 0 | |
| | | 2 | 1 | | 4 | 0 | |
| | | | | | -1 | -1 | |
| 10:34 2.12.2014 | | 1 | 0 | | 0 | 0 | |
| | | 3 | 1 | | 4 | 0 | |
| | | | | | 0 | -1 | |

Auf der y-Achse in der ersten Spalte befinden sich die „timestamp“, das heisst der Zeitpunkt, an dem diese Daten in diesem Zustand waren. Jeweils drei Zeilen zusammen gehören zu einem Zeitstempel, damit bis zu drei Informationen für diesen gespeichert werden können pro Access Point oder Client. Auf der x-Achse Befinden sich die Access Points und in einer zweiten Tabelle die Clients. Die Informationen oberhalb der zeitabhängigen Analysen wurden aus der letzten Kopie der Ausgabedatei ausgelesen, da angenommen wird, dass sie in normalem Betrieb nicht geändert werden. So konnte auch Platz in der zeitlichen Analyse eingespart werden, sodass jeder Client oder Access Point nur die genannten drei Zeilen pro Zeitstempel benötigt. Diese Zeilen wurden bei den Access Points mit „#beacons“, der Anzahl ausgesendeter Beacons im letzten Intervall, und „#allbeacons“, der gesamten Anzahl gesendeter Beacons seit Versuchsbegin. Für die Clients wurde dasselbe mit den sogenannten Packets gemacht, die Probe Requests, Authentifizierungspackete oder weiteres sein können. Zusätzlich wird auch jeweils die MAC-Adresse eingetragen, mit dem ein Client gerade verbunden ist, oder -1 falls er inaktiv ist und 0 wenn er mit keinem Access Point verbunden ist.

Die Auswertung der Kopien der Ausgabedatei übernimmt das Lua-Skript analyse.lua, das auf einem Computer in dasselbe Verzeichnis kopiert werden muss, in dem sich die Kopien befinden. Wenn es ausgeführt wird, wozu Lua for Windows nötig ist (je nach

Betriebssystem natürlich auch andere Versionen), erstellt es eine Datei nach der obigen Vorlage und benennt sie Overview.csv. Dies stellt die einfachste Zusammenfassung aller gesammelten Daten dar. Aus ihr können dann einzelne Daten ausgelesen werden, um Statistiken zu erstellen, die mehr über die gefragten Verhaltensabläufe aussagen.
Dazu wurde wiederum ein Skript geschrieben, das als Beispiel die Anzahl Clients oder Access Points zu jedem Zeitstempel ermittelt und wiederum in einer Datei usercount-ap.csv oder usercount-client.csv abspeichert. Das Programm muss mit 2 Parametern aufgerufen werden, wozu man erst eine Kommandozeile (Windows: Start->Suche->cmd.exe) aufrufen muss, um anschliessend den Befehl `Dateipfad/usercount.lua inputFile ap/client` auszuführen. Dateipfad entspricht dem Speicherort des Skripts usercount.lua und inputFile ist hier meist Overview.csv, da dies die Standartausgabe des analyse.lua-Skripts ist, solange die Datei nicht umbenannt wurde. Für ap/client muss die Option gewählt werden, die gewünscht ist. Die Ausgabedatei dieses Skripts besitzt dann zwei Spalten, in der ersten die Zeitstempel und in der zweiten die jeweilige Anzahl. In einem Programm wie Excel oder OpenOffice Calc kann man damit Diagramme erstellen, die das Resultat visuell und verständlicher darstellen können. Im Anhang befinden sich einige Diagramme aus den Versuchen dieser Arbeit. Sie stammen alle aus dem Versuch im Nawi-Trackt da von den Dateikopien aus der Mensa zu viele fehlerhaft sind und es zu zeitaufwendig war, diese alle zu korrigieren. Vom ersten Versuch konnten auch nur 2 Tage rekonstruiert werden und zeigen nicht den ganzen Tag, doch sieht man am 2.12.2014 bereits so recht viel am 5.12.2014 schienen jedoch nicht allzu viele das WLAN zu benutzen.
Was bei diesen Diagrammen herausgelesen werden kann, ist die ungefähre Nutzung von WLAN-Benutzer während eines Tages, wobei die interessantesten Stunden während der Unterrichtszeiten sind. Die Maxima an Benutzer treten deutlich an den zu erwartbaren Zeitpunkten auf, nämlich bei Schulbeginn und -ende und in den grösseren Pausen. Diese sind mit grau markierten Bereichen gekennzeichnet.
Andere interessante Erkenntnisse kann man aus den zum Teil zahlreichen ESSID-Angaben der Clients schliessen. Da einige WLAN-Netzwerke nach Familiennamen oder deren Standorte  benannt sind kann man daraus schliessen, welches Gerät wo war oder vielleicht sogar (mit zusätzlichen Informationen) wem es gehört. Mein persönliches Smartphone als Beispiel gab zahlreiche benutzte WLAN-Netze an. Einige davon waren die Heimnetze, der während des Versuchs verbundene Arduino, aber auch eindeutige öffentliche Netze wie Europa-Park_Mobile oder MIGROS Wifi. Insgesamt wurden ungefähr zwei Drittel aller auf diesem Smartphone gespeicherten WLAN-Netze

ausgerufen, wobei das letzte public-ksso ist und somit wahrscheinlich gleich anschliessend mit einem Access Point davon verbunden wurde, da dies das untersuchte Netz ist und somit in Reichweite war. Bei den meisten anderen Clients sind nur kurze Listen bis gar keine Vorhanden. Allerdings weiss man zum Beispiel wo ein Client war, wenn Port Of Jersey WiFi Hotspot, McDonald's Free WiFi oder Sturbucks in dieser Liste auftauchen. Mithilfe von online verfügbaren Tools wie die Website www.wigle.net kann die Position von vielen von deren Datenbanken erfasste WLAN-SSIDs sogar auf einer Karte angezeigt werden, das heisst auch weniger aussagende Netzwerknamen wie RIR-20385 können lokalisiert werden. Im Umfang dieser Arbeit wurde nicht weiter auf eine Auswertung der erschienenen SSIDs eingegangen, weil das Ziel nur war, sie aus dem WLAN-Funkverkehr auslesen zu können und die Auswertung wohl auch nicht allzu viel aussagen würde, wenn nicht versucht wird, den Client komplett zu identifizieren.

## 2. 5 Diskussion

Aus der Auswertung kann man sehen, dass man WLAN-Besucherverhalten erkennen kann, um daraus Schlüsse ziehen zu können. Nicht betrachtet wurde bisher, dass man mit mehreren Arduinos oder ähnlichen WLAN-Sniffern zur Erzeugung von Bewegungsprofilen auch beispielsweise Stundenpläne nachvollziehen könnte, indem man den Tagesverlauf eines Clients erstellt und die möglichen Klassenzimmer eruiert. Wenn man die Stundenpläne aller Schüler kennt (abrufbar auf www.ksso.ch) kann man so die Client-Geräte den Klassen zuordnen. Dadurch wurde bereits mehr Information gewonnen, die für grosse Firmen interessant sein können. Das hier gezeigte Beispiel findet in der Realität sicherlich nicht so statt, aber Firmen wie Google nutzen solche Techniken, um immer mehr Informationen über ihre Kunden zu erhalten und ihnen dafür genauere Dienste zu ermöglichen. Diese Informationen ermöglichen Google jedoch auch Kunden- und Standortbezogene Werbung anzuzeigen und dafür viel Geld zu verdienen, wobei das nicht nur Google macht.[18] Anstatt Werbung können WLAN-Betreiber ihre Netzwerk-Infrastrukur auch nutzen, um selbst ihren Kunden eine Anwendung zur Verfügung zu stellen, die mithilfe der Standortinformation auf den Kunden bezogene Hilfe geben kann. So ist dies beispielsweise im Europa Park Rust der Fall, der automatisch auf einer Karte anzeigt, wo man sich befindet. Weitere Dienste sind dann einfach umzusetzen, wie die Anzeige der nächstgelegenen Achterbahnen, Restaurants oder ebenfalls durch WLAN-Analyse gewonnene Wartezeiten. Zukunftsvisionen zeigen Extreme dieser Anwendung, die zum Teil aber bereits umgesetzt wurden. In London gab es Mülleimer, die mit solchen Sniffer-Geräten ausgestattet wurden und somit Bewegungen von WLAN-Nutzer in der ganzen Stadt aufgezeichnet werden konnten. In Kaufhäusern können ausserdem auch

Überwachungskameras mithelfen Bewegungen von Menschen oder deren Smartphones zu analysieren und darzustellen. [19] Eine sehr gute Illustration, wie solche Bewegungsprofile fertig ausgewertet aussehen können sieht man bei einem Versuch an der re:publica 2013 in Berlin, der mittels WLAN-Aanalyse Besucherströme von Vortragssaal zu Vortragssaal interaktiv zeigt. [20]

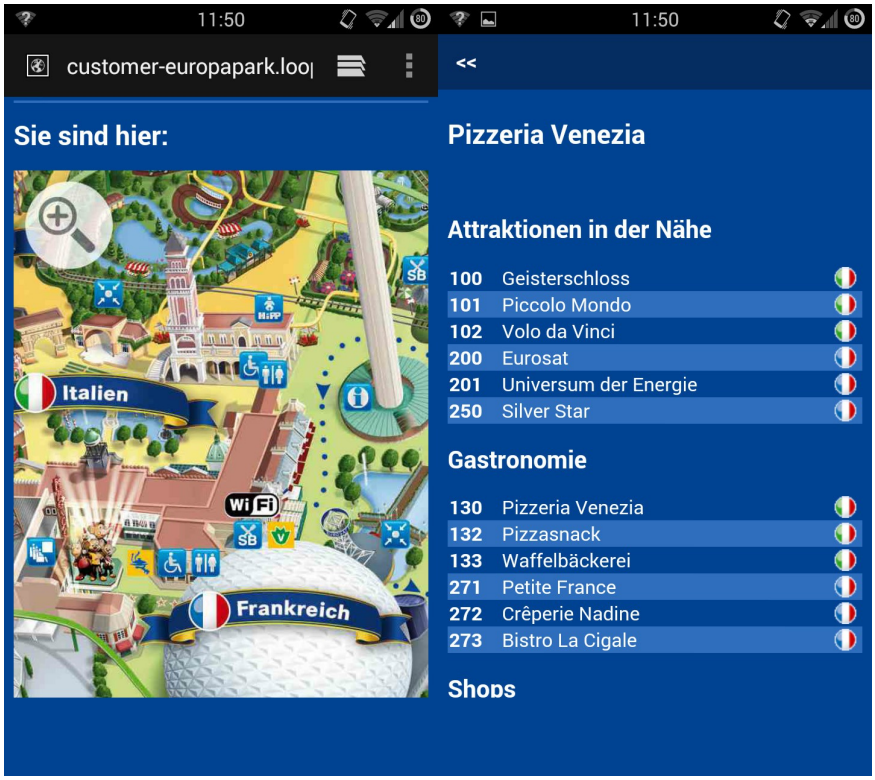



*Abbildung 5: App des Europaparks Rust mit Lokalisierungstechnik*

# 3 Fazit

In dieser Arbeit konnte gut gezeigt werden, dass man mit einfachsten Mitteln ganze WLAN-Netzwerke analysieren kann. Die erhaltenen Daten könnten zu Verbesserungszwecken (falls nötig) verwendet werden oder einfach zur Veranschaulichung der WLAN-Nutzung hingezogen werden. Für diese Projekt gibt es für die Zukunft Möglichkeiten zur detaillierteren Auswertung und Verbesserung der Vorgehensweise. Die Programme sind noch nicht ganz fehlerlos und machen noch nicht alles automatisch. Was auch weiter nachgeforscht werden könnte ist der Ablauf der Abfrage mittels direkter Probe Requests, denn einige Clients haben sehr viele davon ausgesendet, auch wenn von den Access Points Beacons geschickt wurden. Nach Apple benutzen ihre neuen Geräte und iOS8-Betriebssystem eine neue Technik, um WLAN-Tracking nicht möglich zu machen, was im Versuch nicht wiederlegt werden konnte doch laut heise.de nicht stimmt. [21] Vorstellbar wäre auch eine Umsetzung des re:log-Versuch in einem eigenen Umfeld für interessante Darstellungen, [20] der jedoch die Daten direkt von den Access Points erhielt. All diese Vorschläge nutzen verschiedene Techniken, bauen jedoch auf den hier erhaltenen Erkenntnissen auf. Etwas ganz anderes ist das Behandeln der Erkenntnisse bezüglich Ad-hoc Netzwerken, die weitere Verwaltungsstrukturen anwenden und somit noch mehr Kommunikation zwischen den Teilnehmern benötigen.

Das Programmieren der verwendeten Programme und Skripte war noch nicht fehlerfrei und viel Zeit musste in die Fehlersuche investiert werden. Die Programmiersprachen waren allerdings bereits bekannt und die Hardware simpel aufgebaut, weswegen die Fehlerquellen in den langen und komplexen Skripten und dem mangelhaften Verständnis der Prozess-Eigenschaften in Linux liegen müssen. Trotzdem kann man sagen, dass eine gute Basis aufgebaut wurde, die leicht erweiterbar ist.

# 4 Literaturverzeichnis

[1] Wikipedia, Wireless Local Area Network, http://de.wikipedia.org/wiki/Wireless_Local_Area_Network, letzter Besuch: 15.12.2014

[2] openwireless, openwireless.ch, http://solothurn.openwireless.ch/node, letzter Besuch: 15.12.2014

[3] David Wischnjak, Ronald Eikenberg,Auf Schritt und Tritt, in: c't 21 (2014), S. 158-161

[4] clker.com, Access Point, http://www.clker.com/cliparts/f/f/e/4/12065572121317625675no_hope_Wireless_access_point.svg.hi.png, letzter Besuch: 3.1.2015

[5] freepik.com, Smartphone, http://cdns2.freepik.com/fotos-kostenlos/smartphone-symbol_21377498.jpg, letzter Besuch: 3.1.2015

[6] Elektronik Kompendium, Wlan Authentifizierung, http://www.elektronik-kompendium.de/sites/net/1101181.htm, letzter Besuch: 15.12.2014

[7] Intel, Wireless Networking, http://www.intel.com/support/wireless/wlan/sb/CS-025325.htm, letzter Besuch: 15.12.2014

[8] Laura Stresing, Digitales Kundentracking: Gläsern in der Fußgängerzone, http://blog.zdf.de/hyperland/2014/01/digitales-kundentracking-glaesern-in-der-fussgaengerzone/, letzter Besuch: 3.1.2015

[9] PC Magazine, Google: Google Street View Cars Sniffed Wi-Fi Networks, http://www.pcmag.com/article2/0,2817,2363835,00.asp, letzter Besuch: 2.1.2015

[10] Wikipedia, WLAN-basierte Ortung, http://de.wikipedia.org/wiki/WLAN-basierte_Ortung, letzter Besuch: 3.1.2015

[11] heise.de, WLAN-MAC-Adressen: Googles langes Gedächtnis, http://www.heise.de/netze/meldung/WLAN-MAC-Adressen-Googles-langes-Gedaechtnis-1261893.html, letzter Besuch: 3.1.2015

[12] tavendo.com, Arduino Yun, http://tavendo.com/static/img/blog/arduino-yun-getting-started-part-1/arduino-yun.jpg, letzter Besuch: 3.1.2015

[13] Arduino, Arduino Yun, http://arduino.cc/en/Main/ArduinoBoardYun?from=Products.ArduinoYUN, letzter Besuch: 3.1.2015

[14] arduino.cc, Upgrading the OpenWrt-Yun image on the Yún, http://arduino.cc/en/Tutorial/YunSysupgrade, letzter Besuch: 3.1.2015

[15] Aircrack-ng, Aircrack-ng, http://aircrack-ng.org/, letzter Besuch: 2.1.2015

[16] Google Code, luaforwindows, https://code.google.com/p/luaforwindows/, letzter Besuch: 2.1.2015
[17] Physics Light, Arduino Yun Time Synchronization, http://physicslight.wordpress.com/2014/07/02/arduino-yun-time-synchronization/, letzter Besuch: 3.1.2015
[18] Stefan Kuhn von com-magazin.de, So bestimmt Google den Standort, http://www.com-magazin.de/praxis/internet/geolocation-google-weiss-wo-ihr-pc-steht-3418.html?page=1_so-bestimmt-google-den-standort, letzter Besuch: 3.1.2015
[19] Marin Majica von zeit.de, Beim Shoppen auf Schritt und Tritt verfolgt, http://www.zeit.de/digital/datenschutz/2013-08/kunden-tracking-offline-einzelhandel, letzter Besuch: 3.1.2015
[20] datenjournalist.de/relog/, re:log - Besucherstromanalyse per re:publica W-LAN, http://apps.opendatacity.de/relog/, letzter Besuch: 3.1.2015
[21] heise.de, WLAN-Tracking unter iOS 8 weiterhin möglich, http://www.heise.de/mac-and-i/meldung/WLAN-Tracking-unter-iOS-8-weiterhin-moeglich-2404719.html, letzter Besuch: 3.1.2015
[22] jquery.com, jQuery, http://jquery.com/download/, letzter Besuch: 3.1.2015
[23] arduino.cc, Anleitung zu Node.js, http://blog.arduino.cc/2014/05/06/time-to-expand-your-yun-disk-space-and-install-node-js/, letzter Besuch: 3.1.2015

# 5 Abbildungsverzeichnis

# 6 Anhang

Der ganze verwendete Code der Programme ist auf GitHub auf https://github.com/Studi11/Maturaarbeit-WLAN.git abrufbar.

Die Programmdateien sind folgende:

Auf dem Arduino: analyseWlan_net.ino

Auf der microSD: index.html

createFolder.lua

Skripte zur Auswertung auf einem Computer:

analyse.lua

usercount.lua

Zusätzlich zum eigenen Code wurden folgende Libraries und Vorgehensweisen benutzt:

- Arduino Yun Time Synchronisation [17]
- jquery.1.11.1.min.js [22] / [23]

Es folgen einige Diagramme zu den Auswertungen zu den Versuchen am 2. und 5.12.2014.

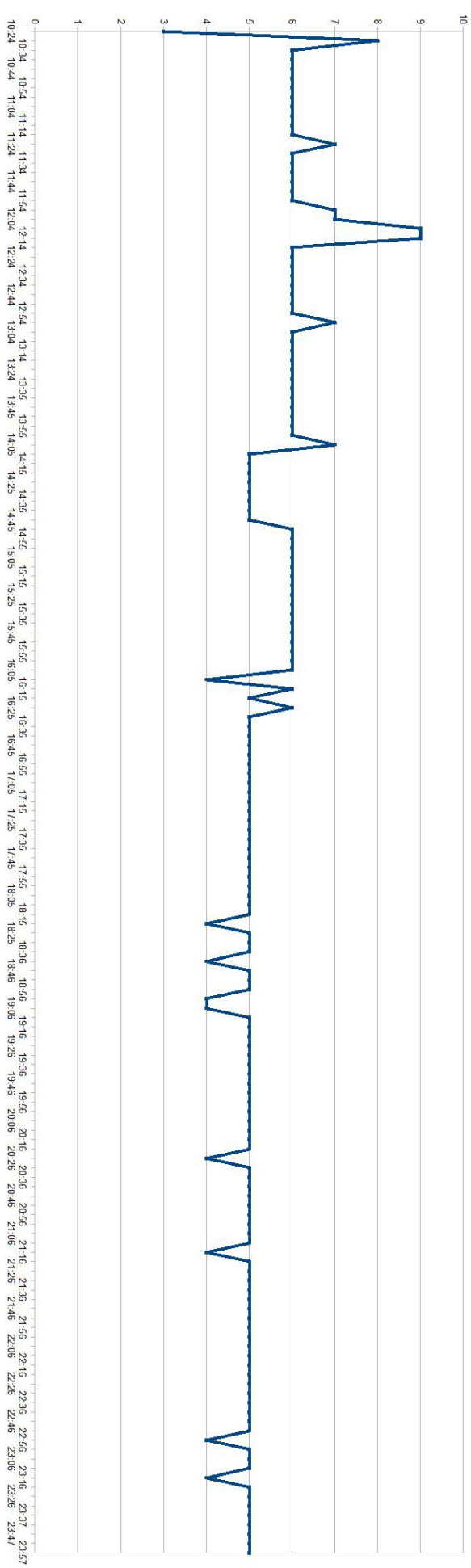


*Abbildung 7: Anzahl APs als Verlauf dargestellt(20.12.2014)*

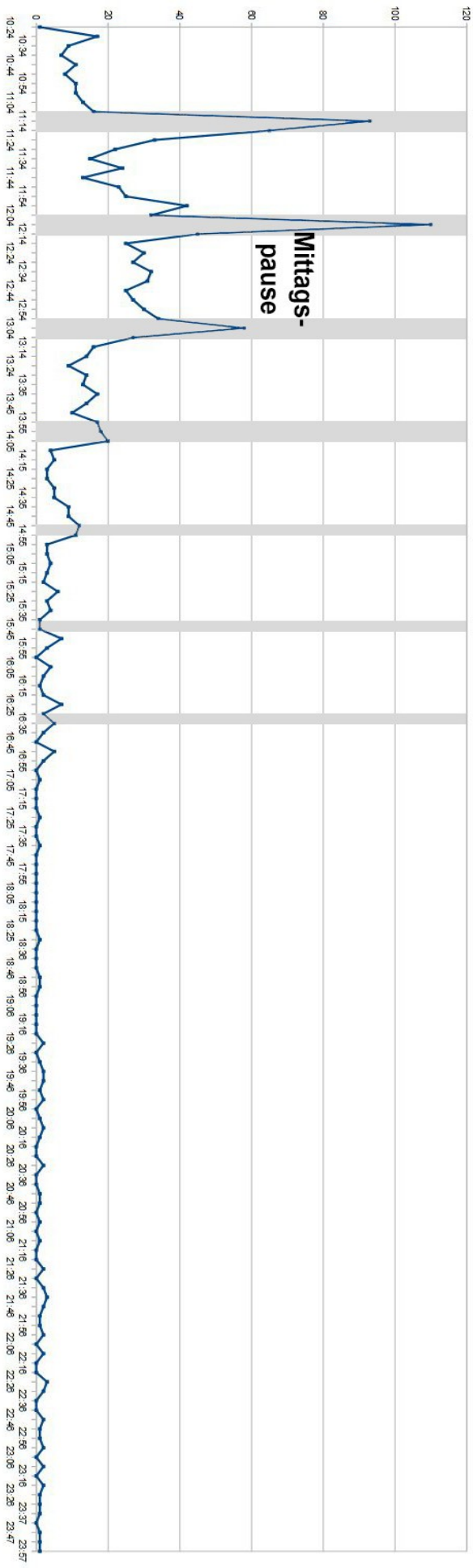



*Abbildung 6: Anzahl Clients als Verlauf dargestellt (2.12.2014)*

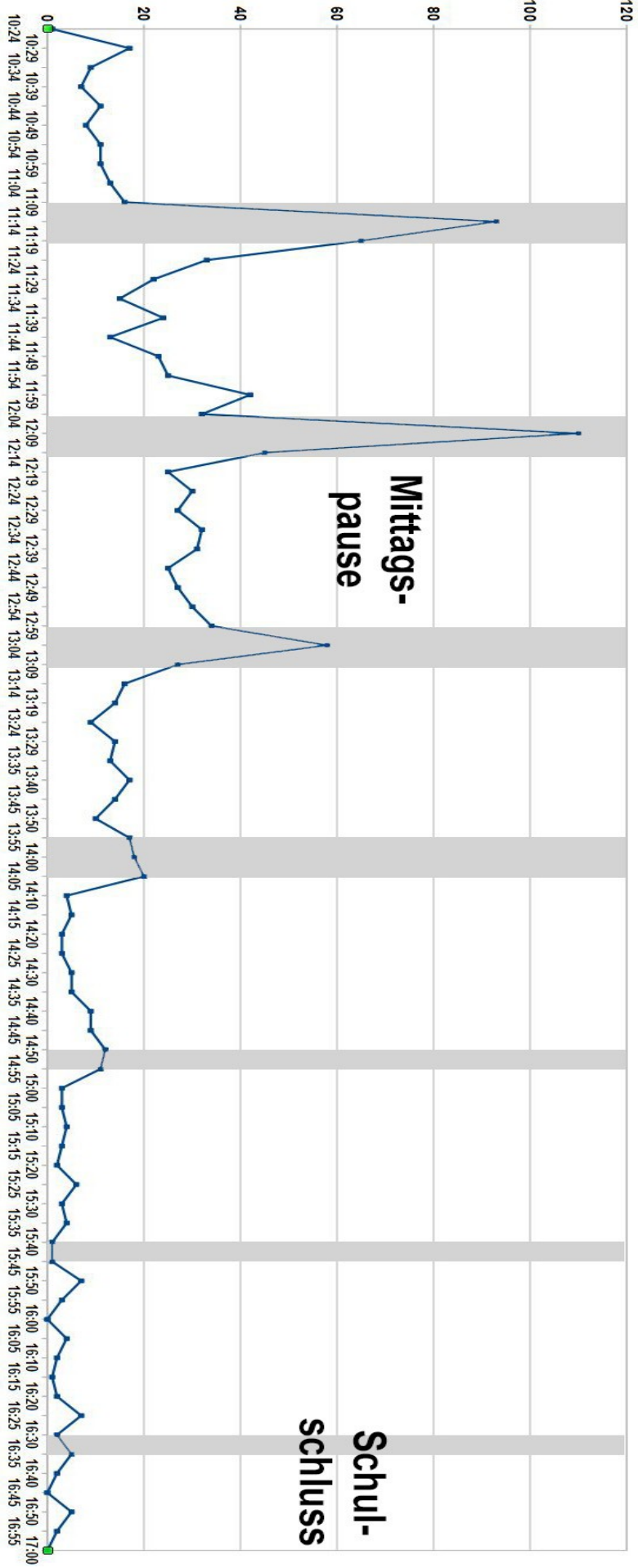



*Abbildung 8: Anzahl APs als Verlauf dargestellt (Unterrichtszeiten 2.12.2014)*

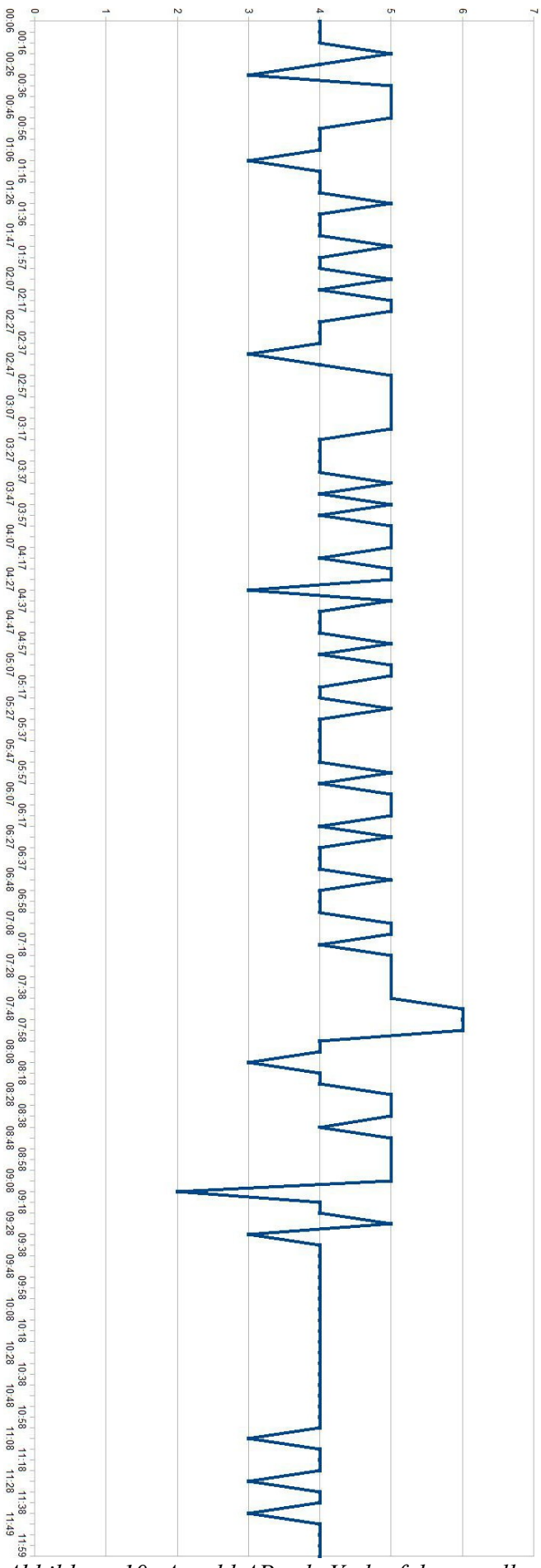


*Abbildung 10: Anzahl APs als Verlauf dargestellt (5.12.2014)*

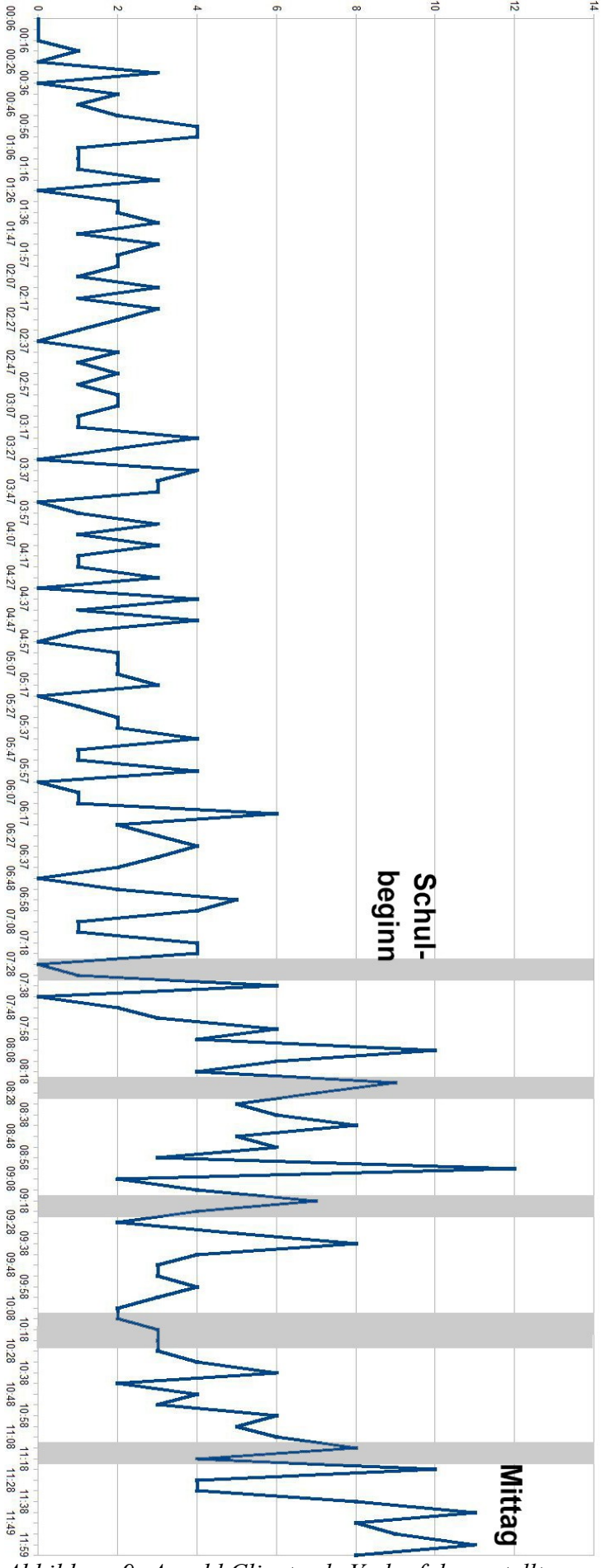



*Abbildung 9: Anzahl Clients als Verlauf dargestellt (5.12.2014)*

Kantonsschule Solothurn
Schulleitung

KANTON solothurn

# Erklärung zur Maturaarbeit

Name Studer Vorname Simon Klasse N11w

## A Selbstständigkeitserklärung

Ich bestätige hiermit, dass ich meine Maturaarbeit:

Titel Was ist im WLAN-Funkverkehr trotz Sicherheit für alle zugänglich und was können uns diese ausgelesenen Daten sagen und nützen

Betreuer/in Vincent Tscherter

selbstständig und ohne unerlaubte Mithilfe verfasst und keine anderen als die angegebenen Quellen benutzt habe. Alle Stellen, die wörtlich oder sinngemäss aus Quellen entnommen wurden, habe ich als solche gekennzeichnet.

Ich bestätige auch, dass ich den Betreuer / die Betreuerin über jegliche Art von Vereinbarungen mit Drittpersonen oder Institutionen informiert habe.

## B Weiterverwendung der Maturaarbeit

Die Arbeiten werden grundsätzlich während drei Jahren in der Mediothek der Kantonsschule archiviert. Sie können dann abgeholt werden oder werden vernichtet.

Ich bin damit einverstanden, dass meine Maturaarbeit Dritten zugänglich gemacht wird.

☒ JA ○ NEIN

Ort / Datum Oberdorf, 4.1.2015

Unterschrift S Studer