WLAN-AP mit regelmäßigem PSK-Tausch und QR-Code Anmeldung

Luca Asmus, 29994
Marius Würstle, 29853
Rolf Wiersch, 29837

Zusammenfassung

Das Ziel dieses Projekts ist es, die Sicherheit im eigenen Gast-WLAN zu gewährleisten, unter Berücksichtigung der Faulheit und Bequemlichkeit vieler Endnutzer. Der pre-shared Key eines WLANs wird in den meisten Netzwerken einmal oder nie geändert. Dadurch können Gäste dauerhaften Zugang zum Netzwerk behalten, obwohl das nicht erwünscht ist. Ein weiteres Problem ist die Umständlichkeit einen sicheren pre-shared Key zu verwenden. Es führt zu unangenehmen Mehraufwand eine kryptische und lange Zeichenkette auf Endgeräten einzugeben. Gelöst wird dies durch einen eigenen Access Point für Gäste. Über diesen wird der Zugriff ins Internet geleitet. Der pre-shared Key wird einmal die Woche oder manuell neu erzeugt und auf einem Display ausgegeben. Die Ausgabe erfolgt in Form eines QR-Codes und in Klartext.

December 12, 2020

Inhaltsverzeichnis

1 Motivation

Der Hauptgrund für dieses Projekt ist es die Sicherheit im eigenen Gast-WLAN zu gewährleisten, unter Berücksichtigung der Faulheit und Bequemlichkeit vieler Endnutzer.

Wenn Gäste in der heimischen Wohnung auftauchen, ist der Wunsch nach freiem WLAN meist sehr groß. Bedeutet, der pre-shared key muss abgelesen und den Gästen bekannt gemacht werden. Folgend muss dieser umständlich von Hand eingegeben werden. Um hierbei Sicherheit zu gewährleisten, ist dieser meist länger und kryptisch gewählt. Dies führt oft zur falschen Eingabe bzw. Mehrversuchen und darauf folgenden Ärger darüber. Weiterhin ist es vom Gastgeber nicht immer erwünscht, dass die Gäste nach der Verabschiedung den Zugriff zum WLAN behalten.

Da die geschilderten Umstände den Verfassern dieses Dokumentes nicht fremd sind, soll mit diesem Projekt eine eigenständige Lösung erstellt werden. Der Fokus liegt auf einfacher Bedienung und komfortabler Sicherheit. Weiterhin wird für die Sicherheit auf fertige Endprodukte von Drittanbietern verzichtet.

2 Grundlagen Hard- und Software

In den folgenden Abschnitten werden behandelte Hardware und Software genauer erläutert.

2.1 Hardware

2.1.1 Raspberry Pi

Der Raspberry Pi wurde für junge Menschen entwickelt, um ihnen eine preisgünstige Möglichkeit zu bieten, sich mit der Informatik zu beschäftigen. Der Einplatinencomputer ist etwa kreditkartengroß und kam Anfang 2012 auf den Markt. Er ermöglicht einen schnellen und praktischen Weg um Wissen in den Bereichen Programmieren und Hardware zu erlangen. Zudem ist er vielseitig einsetzbar. Abbildung 1 zeigt grafisch den Aufbau des Raspberry Pi.

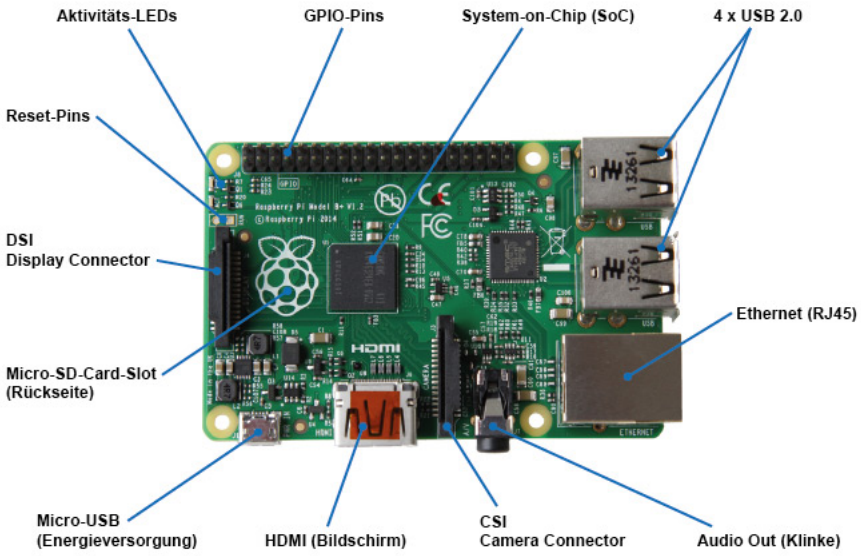

Abbildung 1: Raspberry Pi 3b - Quelle: [7]

Technische Spezifikationen des Raspberry Pi 3b:

- Quad Core 1.2GHz Broadcom BCM2837 64bit CPU
- 1GB RAM
- BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board
- 100 Base Ethernet
- 40-pin extended GPIO
- 4 USB 2 ports
- 4 Pole stereo output and composite video port
- Full size HDMI
- CSI camera port for connecting a Raspberry Pi camera
- DSI display port for connecting a Raspberry Pi touchscreen display
- Micro SD port for loading your operating system and storing data
- Upgraded switched Micro USB power source up to 2.5A [8]

2.1.2 Raspberry Pi Shield - Display LCD-Touch, 3,2in

Der Touchscreen wertet den Raspberry Pi zu einem vollwertigen Touch-PC auf. Für zusätzliche Funktionen besitzt der Display 3 Taster an der Seite, welche einfach über die GPIO Pins eingelesen werden können. Zu sehen ist der Display in Abbildung 2.

Abbildung 2: Touchscreen Display für den Raspberry Pi - Quelle: [16]

Technische Spezifikationen unseres Raspberry Pi Shield - Display LCD-Touch, 3.2in:

- Display 8,13cm (3,2")
- Auflösung 320 x 240 Pixel
- LED-Hintergrundbeleuchtung
- 3 frei belegbare Taster (angebunden an GPIO12, 16, 18)
- SPI-Schnittstelle
- Touchscreen Technologie resistiv

2.2 Software

2.2.1 balenaEtcher

Es handelt sich um ein Programm mit dem ein OS auf eine SD Karte geflasht werden kann.

2.2.2 Hostapd

Damit ist es möglich Geräte, die ein WLAN-Modul besitzen, als Access Point zu betreiben. Jedoch können keine Einstellungen im Bereich IP und Routing vorgenommen werden. Die Software ist nur für das Erstellen eines "wireless Ethernet switches" zuständig. [9] [3]

2.2.3 dnsmasq

Geräte in einem Netzwerk benötigen zur Kommunikation eine IP-Konfiguration und einen DNS Server für die Namensauflösung. Das kann durch dnsmasq erreicht werden. Mit dnsmasq kann ein DHCP Server und DNS forwarder erstellt werden [4]. So können IP-Konfigurationen im Netzwerk verteilt und DNS Anfragen angenommen werden.

2.2.4 netfilter-persistent und iptables-persistent

Für die Durchführung des Projektes ist es nötig iptables-Regeln anzulegen. Diese sollten nach einem Neustart nicht neu angelegt werden müssen. Deshalb werden die Pakete netfilter-persistent und iptables-persistent installiert. Damit können die Regeln in eine Datei abgespeichert und beim Neustart automatisch geladen werden.[5] [6]

2.2.5 Bash

Im Projekt wird Bash benutzt um die Konfiguration zu automatisieren. Bash ist als Standardshell bei Raspberry Pi OS Lite vorinstalliert.

2.2.6 Cron

Cron ermöglicht das zeitbasierte Ausführen des Bash Skripts. So kann beispielsweise jeden Montag um 03:00 Uhr nachts das Passwort automatisch geändert werden.

2.2.7 fim

fim ermöglicht es Bilder oder andere Media Dateien im Terminal anzuzeigen. In diesem Fall wird der QR-Code als .png erzeugt und mithilfe von fim angezeigt. [14]

2.2.8 Python 3.7

Für die Skripte zur Passwortgenerierung, QR-Code Generierung und zum Einlesen der Buttons wird die Sprache Python verwendet. Python 3.7 ist bei Raspberry Pi OS Lite vorinstalliert und erleichtert durch verschiedene Bibliotheken die Umsetzung des Projektes.

2.2.9 pyqrcode

Das Modul pyqrcode wird dafür benutzt, möglichst einfach und frei QR-Codes zu erzeugen. Zum Erzeugen des Codes benötigt es nur die Parameter die enthalten sein sollen. [13]

2.2.10 GPIO zero

Das Modul GPIO zero stellt ein Interface für GPIO Geräte dar [1]. Es wird in diesem Projekt für das Einlesen der einzelnen Taster genutzt, die bereits in dem Display integriert sind. Das Mapping der Taster wird auch in der Anleitung des Displays erklärt. (Taste1 = PIN12/GPIO18) [1]

3 Problemstellung

Vielen Endnutzern ist die eigene Bequemlichkeit sehr wichtig. Daraus resultiert, dass der pre-shared Key eines WLANs in den meisten Netzwerken einmal oder nie geändert wird. Dadurch können Gäste dauerhaften Zugang zum Netzwerk behalten, obwohl das nicht erwünscht ist. Dies hat zur Folge, dass sobald einmal das Passwort bewusst/unbewusst weitergegeben wurde, jeder sich mit dem Access Point verbinden kann. Negative Auswirkungen könnten sich äußern in Form von Missbrauch des Internetzuganges oder durch Angriffe auf das interne Netzwerk.
Ein weiteres Problem ist die Umständlichkeit einen sicheren pre-shared Key zu verwenden. Es führt zu unangenehmen Mehraufwand eine kryptische und lange Zeichenkette auf Endgeräten einzugeben bzw. zu merken, den viele Endnutzer nicht eingehen wollen. Deshalb werden oft einfache pre-shared Keys in Form von z.B Wortkombinationen verwendet. Dies hat zur Folge, dass der Zugang einfacher geknackt werden kann.

4 Anforderungsanalyse

Im Folgenden werden die verschiedenen Anforderungen genauer beschrieben.

4.1 Funktionale Anforderungen

Funktionale Anforderungen sind Anforderungen, die notwendig für die Umsetzung des Projektes sind.

4.1.1 Funktionaler Accesss Point

Um einen Zugriffspunkt für die Endnutzer ins Netzwerk zu schaffen, muss passende Hardware gefunden und konfiguriert werden. Hierauf wird ein Access Point mit gängiger Funktionalität erstellt.
Die Hardware sollte für ihre Aufgaben passend dimensioniert sein. Ebenso sollte sie ein geeignetes WLAN-Modul besitzen um ein eigenes WLAN aufspannen zu können. Am Besten ist es, wenn dieses die heutzutage gängigen Funkstandards unterstützt.

Weiterhin müssen sich die Endnutzer im WLAN anmelden und kommunizieren können. Unter anderem wird ein DNS und DHCP Server benötigt. Ohne diesen kann keine Kommunikation im Funknetzwerk entstehen. Grund ist die fehlende IP-Konfiguration der Endgeräte.

4.1.2 Zugriff ins Netzwerk

Ohne Internet Zugriff ist zwar eine Anmeldung beim Access Point möglich, aber nicht sinnvoll. Deshalb muss gewährleistet werden, dass der Datenverkehr vom Access Point weitergeleitet wird und nicht geblockt wird. Natürlich soll dieser auch wieder zurückgeleitet werden. Am Besten besitzt die Hardware außerdem einen LAN Anschluss. Dieser kann als Verbindung zur schon bestehenden Infrastruktur dienen.

4.1.3 Sicheres Passwort

Beim Erstellen des Passworts muss beachtet werden, dass dieses auch sicher ist. Es soll unwahrscheinlich sein dieses in seinem gültigen Zeitraum knacken zu können. Deshalb muss ein Konzept zur Erstellung eines sicheren Passworts erstellt und umgesetzt werden.

4.1.4 Automatisches Wechseln des Passworts

Es soll möglich sein das sich das Passwort vollautomatisch in einem gewissen Zeitraum ändert. Folglich muss ein Skript erstellt werden, welches das Passwort austauschen kann.

4.1.5 Ausgabe des Passworts

Die Endnutzer müssen über das neu erstellt Passwort benachrichtigt werden. Zu beachtet ist, dass die Darstellung so komfortabel wie möglich umgesetzt wird. Es soll kein zu großer Mehraufwand entstehen. Weitere Hardware in Form eines Bildschirm zur Darstellung muss angeschlossen werden.

4.2 Optionale Anforderungen

In diesen Abschnitt werden Anforderungen beschrieben, die nicht für die grundlegende Funktionalität benötigt werden.

4.2.1 Manueller Wechsels des Passworts

Wenn es gewünscht ist, soll es möglich sein das Passwort manuell neu erstellen und wechseln zu lassen. Das soll in Form einer physischen Benutzereingabe erfolgen. So wird sichergestellt, dass nur Befugte diesen Vorgang anstoßen können.

4.2.2 Energiespar-Konfiguration

Nicht jeder möchte, dass ein Bildschirm ununterbrochen beleuchtet wird. Weiterhin könnte es nicht erwünscht sein, dass das Gast-WLAN 24 Stunden sendet. Deshalb könnte ein Energiesparplan erstellt und implementiert werden. Dieser kann die Anschaltzeit des Displays und des Access Points regeln.

4.2.3 Automatisierungs-Skript

In Zukunft kann es wünschenswert sein, die erstellten Konfigurationen erneut auf anderen Geräten auszuführen. An dieser Stelle ist ein Skript zur Automatisierung sinnvoll. Das spart Zeitaufwand und kann bei der Veröffentlichung Anderen helfen.

4.3 Priorisierung

Das Wichtigste zu Beginn ist, dass die Hardware aufeinander abgestimmt wird. Bedeutet, alle Teile passen zusammen und können angeschlossen werden.

Danach muss die Konfiguration des Raspberry Pi zum Access Point erfolgen. Sobald dies funktioniert kann das Generieren des Passworts und dessen Austausch stattfinden.

Folglich wird die Ausgabe des Passworts am Bildschirm realisiert. Danach kann es um das Generieren des QR-Codes und dessen Ausgabe erweitert werden.

Wenn diese Punkte voll funktional umgesetzt werden konnten, kann sich um einen automatischen oder manuellen Job (bei Tastendruck) zum Generieren und Austauschen des Passworts gekümmert werden.

5 Lösungsidee

Ein Lösungsansatz stellt ein sich automatisch oder auf Tastendruck änderbares Passwort dar. Das Passwort wird jeden Montagmorgen um 03:00 Uhr automatisch durch einen "Cron-Job" gewechselt. Ein Taster kann zusätzlich betätigt werden, falls das Passwort sofort geändert werden soll.

Das Passwort wird so gewählt, dass er aus mathematischer Sicht nicht in dem Zeitraum geknackt werden kann, bis automatisch ein Neuer erzeugt wird. Durch diese Maßnahme wird die Sicherheit des WLANs verbessert und sicher gestellt.

Mit diesem neu generierten Passwort können sich Endgeräte anmelden und kommunizieren. Hiermit wird das Problem der ungewollten Nutzern gelöst. Diese können sich sobald das Passwort gewechselt hat, nicht mehr im Netz anmelden.

Der Passwort wird an einem Display in zwei verschiedenen Varianten angezeigt:

- Klartext für Endgeräte ohne Kamera z.B. Laptops
- QR-Code zum Scannen für z.B Smartphones

Ürsprünglich war ein RFID/NFC Transponder für die Anmeldung geplant. Es wurde sich für einen QR-Code entschieden, da hier die Möglichkeit des Abgreifens des Passworts nicht besteht. In Abbildung 3 ist der Aufbau des Lösungsansatzes graphisch dargestellt.

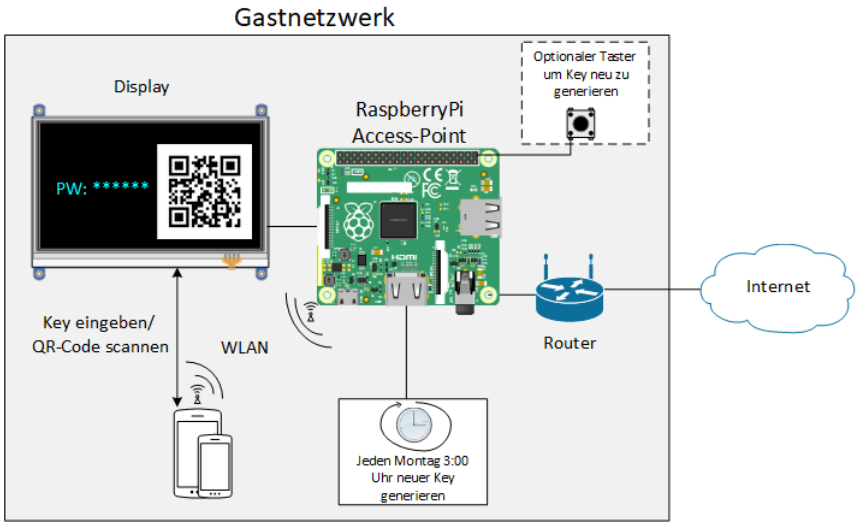

Abbildung 3: Aufbau des Gastnetzes

Es ist zu erwähnen, dass sich diese Lösung auf die Umsetzung eines Gastnetzes bezieht. Es melden sich dort hauptsächlich mobile Geräte an. Dies führt zu einer hohen Fluktuation an Endgeräten im Netzwerk und ist nicht für stationäre Geräte wie z.B Drucker geeignet.

5.1 Hardware

Für dieses Projekt wird ein Raspberry Pi 3 B benötigt. Grund hierfür ist das integrierte WLAN-Modul, ausreichend Leistung und alle benötigten Anschlüsse sind vorhanden.
Zusätzlich wird eine Mikro-SD Karte zum Laden des Betriebsystems und zur

Speicherung der Daten benötigt.

Das Display, welches verwendet wird, muss eine ausreichende Auflösung für die Darstellung des QR-Codes besitzen. Deshalb können keine kleineren und billigeren LCD Anzeigen verwendet werden. Außerdem muss es die Möglichkeit haben entweder externe Taster anzubinden oder schon eigene Taster zu besitzen. Für diese Aufgabe bietet sich das Raspberry Pi Shield - 3,2in LCD-Touch Display an, da es diesen Anforderungen entspricht.

5.2 Software

Um den Raspberry Pi als Access Point verwenden zu können, müssen zusätzliche Pakete installiert (Hostapd) und eine Netzwerkkonfiguration vorgenommen werden. Darunter fällt das Einstellen von DHCP und DNS für die Clients durch "dnsmasq" und Anpassen der IP-Konfiguration. Das Routing erfolgt durch eine neue iptables Regel.

Als Skriptsprache empfiehlt sich Python, da es dort sehr viele Libraries gibt, die einiges an Arbeit abnehmen. Zudem kann mit Python auf einfache Kommandos des Betriebssystems zugegriffen werden. Für die QR-Code Generierung eignen sich die Libraries "qrcode" und "PyQRCode".

5.3 Passwortkonzept

Als Passwortkonzept wurde sich auf einen 12 Zeichen langen Key geeinigt. Dieser benützt 62 Zeichen in Form von Groß-, Kleinbuchstaben und Zahlen. Begründet wurde diese Entscheidung mit folgender Annahme:

Angenommen ein leistungsstarker Rechner schafft durch Brute-Force 2 Billionen Keys pro Sekunde, so würde er 18670 Tage benötigen, um alle Keys zu testen. Wenn schon zur Hälfte der Zeit der richtigen Key gefunden wurde, wäre dies immernoch mehr als ausreichend für eine Woche. Zu sehen ist dies in der folgenden Rechnung:

$$(62^{12})\ keys\ \div\ 2000000000000\ \frac{keys}{s} = 1613133381.198\ s \tag{1}$$

$$1613133381.198949911\ s\ \div\ 60\ \div\ 60\ \div\ 24\ \approx 18670\ Tage \tag{2}$$

$$18670\ Tage\ \div\ 2\ =\ 9335\ Tage \tag{3}$$

So wäre die Sicherheit des Key gewährleistet.

5.4 Tests

Zu Testzwecken werden unterschiedliche Smartphones (Android/iOS) und Notebooks (Windows/Linux/macOS) benutzt. So soll sichergestellt werden, dass mögliche Probleme aufgrund von Diskrepanzen zwischen den Betriebssystemen bzw. Hardware erkannt werden.

6 Bewertung der Lösung anhand der Anforderungen

Der Raspberry Pi 3b ist für die Anforderungen geeignet, da ein WLAN-Modul, genügend Rechenleistung und alle nötigen Anschlüsse vorhanden sind.

Durch Hostapd kann die Access Point Funktionalität optimal realisiert werden. Zusätzlich durch dnsmasq wird der DHCP und der DNS Server umgesetzt. Beide genannten Lösungen bieten eine einfache und schnelle Konfiguration.

Das Weiterleiten des Datenverkehrs übernimmt die iptables Konfiguration. Iptables empfiehlt sich, weil es vorinstalliert und das Team bereits damit vertraut ist.

Mit der genannten Beispielrechnung wurde die Sicherheit des Passworts gezeigt. Das Passwort ist damit für den Zeitraum einer Woche mehr als sicher genug.

Cron ist spezialisiert für zeitbasierte Aufgaben, womit es prädestiniert für das vollautomatische Tauschen des Passworts mithilfe des Passworttausch Skripts ist.

Das ausgewählte Display ist geeignet, aufgrund der ausreichenden Auflösung und der bereits angeschlossenen Taster. Die Ausgabe als QR-Code vereinfacht die Eingabe des Passworts für die Nutzer. Für andere Endgeräte ohne Kamera wird das Passwort zusätzlich dort im Klartext angezeigt.

Abschließend lässt die Lösungsidee genügend Spielraum zur Umsetzung der optionalen Anforderungen.

7 Implementierung

7.1 Vorbereitung des Raspberry Pi

7.1.1 Auswahl und Installation des Betriebssystem

Um mit dem Projekt beginnen zu können wird zuerst ein Betriebssystem bestimmt. Es wurde sich für das Raspberry Pi OS Lite entschieden. Begründet ist diese Entscheidung durch die weniger vorinstallierten Pakete und einer fehlender grafischen Bedienoberfläche. Hierdurch kann Speicherplatz und Sicherheitsrisiken eingespart werden. Je weniger unbenutzte Software vorhanden, desto weniger Angriffsfläche wird angeboten.

Nach der Auswahl des Betriebssystems kann es auf eine SD-Karte geschrieben werden. Hierzu wird die Software balenaEtcher verwendet.

7.1.2 SSH Zugriff einrichten

Da das Projektteam aus drei Personen besteht, wird ein SSH Zugriff in den Einstellungen des Raspberry Pi eingerichtet. Die Einstellungen kann mit folgendem Befehl geöffnet werden:

```
1 sudo raspi-config
```

In diesem Zuge wird der SSH Zugriff aktiviert und das Standardpasswort geändert. Durch den Zugriff kann das parallele Arbeiten am Projekt ermöglicht werden.

7.1.3 Aktualisierung und Paketinstallation

Nach der Neuinstallation eines Betriebssystems fehlen diesem oft die aktuellsten Versionen von Softwarepaketen und Updates. Deshalb werden diese zuerst aktualisiert und installiert. So können Konflikte aufgrund veralteter Software vermieden und die Sicherheit verbessert werden. Darauf folgt das Nachinstallieren der für das Projekt noch benötigten Pakete. Diese wurden im Abschnitt "Grundlagen Hard- und Software" genauer beschrieben.

7.2 Konfiguration des RaspberryPi als funktionalen Access Point

7.2.1 WLAN Interface

Der Raspberry Pi benötigt eine statische IP Konfiguration für sein WLAN Interface. Diese wird in der Datei /etc/dhcpcd.conf vorgenommen. Die Datei wird um folgendes ergänzt:

```
interface wlan0
    static ip_address=192.168.4.1/24
    nohook wpa_supplicant
```

Mit der 192.168.4.1 wird eine statische IP Adresse vergeben unter die der Raspberry Pi im WLAN erreichbar ist. Weiterhin wird der wpa_supplicant deaktiviert um keine Konflikte mit Hostapd zu verursachen. [9] [15]

7.2.2 Routing

Der Access Point muss den Datenverkehr der Endgeräte im WLAN zum Router weiterleiten können. Hierzu wird in /etc/sysctl.d/routed-ap.conf ein Eintrag hinzugefügt bzw. das Kommentarzeichen entfernt:

```
# Enable IPv4 routing
net.ipv4.ip_forward=1
```

Endgeräte können nun den Hauptrouter erreichen. Um jedoch eine Kommunikation zu ermöglichen muss NAT eingestellt werden. Die wird durch einen Eintrag in die iptables Firewall erreicht:

```
sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

Bei Datenverkehr zum Hauptrouter wird nun die Absender IP Adresse der Endgeräte mit der IP der LAN-Schnittstelle ersetzt. Bei Rückantworten an den Raspberry Pi werden diese an den jeweiligen Absender richtig weitergeleitet. [9] [15]

Um die Firewall Regel bei einem Neustart zu behalten, wurde diese abgespeichert:

```
sudo netfilter-persistent save
```

7.2.3 DNS und DHCP

Durch dnsmasq können nun die DHCP und DNS Einstellungen erfolgen. Diese werden in der /etc/dnsmasq.conf Datei vorgenommen. Diese dient als Vorlage und gibt Erklärungen zu den Einstellungen. Zur Übersichtlichkeit wurde diese in dnsmasq.conf.orig umbenannt und eine neue Datei mit dem ürsprünglichen Namen erzeugt. In der neuen Datei werden nur die getätigten Konfigurationen eingetragen:

```
interface=wlan0 # Listening interface
dhcp-range=192.168.4.2,192.168.4.20,255.255.255.0,24h
                # Pool of IP addresses served via DHCP
domain=wlan     # Local wireless DNS domain
address=/gw.wlan/192.168.4.1
                # Alias for this router
```

Zuerst wird das Interface angeben, bei den die DHCP/DNS Konfiguration gelten soll. Die ist das schon vorherig erstellte Interface "wlan0". Es wurde sich auf einen DHCP Bereich von 192.168.4.2/24 - 192.168.4.20/24 entschieden. Dieser umfasst 18 IP Adressen, welcher als ausreichend für eine Woche angesehen wird. Die Lease-Zeit wurde auf 24 Stunden eingestellt, da Gäste meist nicht länger als einen Tag anwesend sind. Zuletzt wurde eine lokale DNS Domäne und ein Alias für den Access Point eingestellt. Unter diesem Alias ist dieser nun erreichbar.

7.2.4 Access Point Einstellungen

Um den Raspberry Pi als Access Point zu nutzen muss Hostapd konfiguriert werden. Hierzu wird zuerst der Dienst aktiviert und so eingestellt das er beim booten gestartet wird:

```
sudo systemctl unmask hostapd
sudo systemctl enable hostapd
```

Nun muss die Konfigurationsdatei unter /etc/hostapd/hostapd.conf erstellt und gefüllt werden. In dieser werden verschiedene Parameter eingestellt. Darunter fällt unter anderem die SSID, das Passwort und die Art der Verschlüsselung. Es wird eingestellt, dass nur WPA2 verwendet wird, da WEP als unsicher gilt. Weiterhin wird der Funkstandard auf IEEE 802.11n und 2.4GHz eingestellt. Grund hiefür ist, dass der Raspberry Pi keinen höheren Standard in Form von z.B IEEE 802.11ac unterstützt. Weiterhin ist der eingestellte Standard ausreichend für das surfen im Gast-Internet.

Der Kanal wird fest auf Sechs gesetzt. Eine passende Kanalsuche mittels "Automatic Channel Selection" (ACS) [2] kann nicht erfolgen. Die Implementierung von ACS in Hostapd wird nur von bestimmten Atheros Treibern unterstützt [2]. Der Raspberry Pi besitzt onboard jedoch nur einen Broadcom-Chip [8] und ist somit nicht kompatibel.

Um Wireless-Networking auf dem Raspberry Pi zu ermöglichen muss ein "Country Code" gesetzt werden. In diesem Fall auf "DE", welches Deutschland entspricht [10]. Dieser ist notwendig, denn je nach Land sind die Frequenzbänder unterschiedlich vergeben bzw. reguliert. Im Folgenden der Inhalt der Konfigurationsdatei:

```
country_code=DE
interface=wlan0
ssid=HimbeerWLAN
hw_mode=g
ieee80211n=1
channel=6
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=GeneratePW
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
```

Nach den Einstellungen erfolgt ein Reboot und der Access Point ist nun einsatzbereit.

7.3 Passwortskript

In dem Python Skript changePassword.py befindet sich der gesamte Ablauf des Passworttauschs. Das vollständige Skript findet sich im dazugehörigen Github Repository.[12]

7.3.1 Passwortgenerierung

Die erste Funktion des Skripts ist für die Generierung eines sicheren Passworts zuständig. Das Skript verwendet die zwei Imports string und secrets.

Mithilfe der Bibliothek string wird definiert, aus welchem Alphabet das Passwort generiert wird.

Das secrets Modul wird für das Generieren von stark kryptographischen Passwörter verwendet. Die verwendete Funktion secrets.choice wählt aus der mitgelieferten Sequenz ein zufälliges Zeichen aus.

Das Zeichen wird anschließend an den schon vorhandenen String angehängt. Dies wird 12 mal wiederholt um die gewünschte Passwortlänge zu erreichen.

```
#!/usr/bin/env python3

import secrets
import string

def get_random_pw():
        alphabet = string.ascii_letters + string.
            ↪ digits
        password = ''.join(secrets.choice(alphabet
            ↪ ) for i in range(12))
```

```
9           return password
```

7.3.2 Austauschen des Passworts

Nachdem das Passwort generiert wurde, liest changePassword.py den Typ der Verschlüsselung sowie die SSID des Access Point aus der hostapd.conf Datei aus. Die drei Parameter werden einmal als Klartext auf der Konsole ausgegeben und dann werden sie der Funktion zum Erstellen des QR-Codes übergeben.

Im Anschluss führt das Skript mit dem Subprocess Modul das Unix Tool sed aus. Mit sed wird die Zeile der hostapd.conf angepasst, welche das Passwort enthält. Mit der Option -i nimmt sed die Änderung direkt an der gegebenen Datei vor statt nur zu stdout zu schreiben.

Damit der Tausch in Kraft tritt muss der Hostapd Service neu gestartet werden.

```
1  #!/usr/bin/env python3
2  import string
3  import re
4  import subprocess
5
6  def get_hostapd_text():
7      file = open('/etc/hostapd/hostapd.conf', 'r')
8      text = file.read()
9      file.close()
10     return text
11
12 def get_wpa():
13     text = get_hostapd_text()
14     matches = re.findall("wpa=.", text)
15     return matches[0].replace('wpa=', 'WPA')
16
17 def get_ssid():
18     text = get_hostapd_text()
19     matches = re.findall("ssid=.*", text)
20     return matches[0].replace('ssid=', '')
21
22 def change_password(password):
23     subprocess.run(
24         [
```

```
25              'sed',
26              '-i',
27              's/wpa_passphrase=.*/wpa_passphrase={
                    ↪ password}/g'.format(
28                  password=password
29              ),
30              '/etc/hostapd/hostapd.conf'
31          ]
32      )
33      subprocess.run(['systemctl', 'restart', '
            ↪ hostapd.service'])
34
35  if __name__ == "__main__":
36      ssid = get_ssid()
37      wpa = get_wpa()
38      pw = get_random_pw()
39      print('SSID:', ssid,  wpa, 'PW:', pw)
40      create_qr_code(ssid, wpa, pw)
41      change_password(pw)
```

7.3.3 Erstellen des QR-Code

Die Funktion zur QR-Code Generierung bekommt als Argumente die SSID des Netzwerks, die WPA Einstellung und das Passwort. Diese werden ausgelesen und in die pyqrcode.create als String mitgegeben. Das Format des Strings ist sehr wichtig, denn so wird definiert wie das Endgerät den QR-Code zu interpretieren hat.

Der Code wird als png-Datei im Homeverzeichnis gespeichert, von wo er später ausglesen und angezeigt werden kann.

```
1  #!/usr/bin/env python3
2
3  import pyqrcode as pqr
4
5  def create_qr_code(ssid, security, password):
6      qr = pqr.create('WIFI:S:{ssid};T:{security};P
           ↪ :{password};;'.format(
7          ssid=ssid,
8          security=security,
9          password=password
10     ))
11     qr.png('qrcode.png', scale=9)
```

7.3.4 Zeitbasiertes Ausführen

Cron erlaubt es, das Passwort regelmäßig zu einer definierten Zeit, hier beispielsweise Montags um 03:00 Uhr nachts, zu tauschen. Nachts bietet sich an, da zu dieser Zeit für gewöhnlich das Netz kaum bis nicht genutzt wird.

Damit das Passwort dennoch auf dem angeschlossenen Bildschirm angezeigt werden kann ist es wichtig beim Aufrufen des Skripts die Standardausgabe auf das korrekte Gerät umzulenken. In diesem Fall wird stdout auf /dev/tty1 umgelenkt.

Da manche Änderungen root-Rechte benötigen, wird der Aufruf mit dem Nutzer root definiert.

```
1 * 3 * * 1 root changePassword.py > /dev/tty1
```

7.4 Einrichten des Displays

Die Einrichtung des Displays wurde nach der mitgelieferten Anleitung durchgeführt. [11] Zusätzlich wurde auf einem anderen Raspberry Pi das Beispiel Image installiert und mit dem Vorliegenden verglichen. Zunächst musste die /boot/config.txt so verändert werden, dass die Ausgabe mit dem richtigen Treiber auf dem Display erscheint. Folgende Konfigurationen sind zu machen:

```
1 dtparam=audio=on
2 dtparam=spi=on
3 dtoverlay=joy-IT-Display-Driver-32b-overlay:rotate
    ↪ =270,swapxy=1
4
5 hdmi_ignore_edid=0xa5000080
6 hdmi_force_hotplug=1
```

Anschließend wird der Treiber auf die passende Konsole gemapped. Hierzu fügt man an das Ende der ersten Zeile in /boot/cmdline.txt die Konfiguration:

```
1 fbcon=map:10
```

Natürlich muss der Treiber herunterladen, auspackt und nach /boot/overlays verschoben werden. Über diesen Link kann das Paket heruntergeladen werden:

"http://joy-it.net/files/files/Produkte/RB-TFT3.2-V2/joy-IT-Display-Driver-32b-overlay.zip"

Jetzt muss man die Datei 99-calibration.conf in /usr/X11/xor.conf-d/ mit folgenden Einstellungen erstellen:

```
1 Section "InputClass"
2         Identifier "calibration"
3         MatchProduct "ADS7846␣Touchscreen"
4         Option "Calibration" "189␣3767␣3842␣249"
5         Option "SwapAxes" "0"
6 EndSection
```

In diesem Projekt wird das Raspberry OS Lite verwendet. Weil in diesem Betriebssystem sehr wenige Pakete vorinstalliert sind, muss für den Display noch zusätzlich xserver-xorg-video-fbturbo installiert werden.
Anschließend muss für die Touch-Funktion die Datei /usr/share/X11/xorg.conf.d/10-evdev.conf in das selbe Verzeichnis kopiert und in 45-evdev.conf umbennant werden. Zum Schluß muss auch für die Touch-Funktion das Packet server-xorg-input-evdev installiert und der Raspberry Pi neugestartet werden.

7.5 Ausgabe des Passworts

Bei der Erstellung des Passworts wird es bereits im Format: "SSID: <ssid> WPA<version> PW:<passwort>" als Plaintext auf die Konsole geschrieben.

7.5.1 Tastenbelegung

Damit der Tastendruck erkannt wird, gibt es in dem GitHub Repository [12] das buttonInput.py Skript. Mithilfe von GPIO zero kann den Tastern der gewünschte GPIO zugewiesen werden, auf den sie reagieren sollen.

Taster1 zeigt den QR-Code an. Die wird im Abschnitt "Anzeigen des QR-Codes" genauer erläutert.

Taster2 wird dafür verwendet ein neues Passwort zu generieren. Dort wird das Passworttausch Skript aufgerufen.

Taster3 führt ein "clear" aus. Somit kann der Display aufgeräumt werden, falls dort zu viele Passwörter stehen.

7.5.2 Anzeigen des QR-Codes

Damit der QR-Code angezeigt wird muss der erste Taster an dem Display gedrückt werden. Dies hat den Grund, dass der QR-Code eine .png Datei ist. Die Datei kann nicht einfach auf dem Terminal ausgegeben werden. Ein weiterer Grund ist, dass der QR-Code den Plaintext verdrängen würde und dann nicht mehr sichtbar wäre.

Nach dem Tastendruck wird ein Subprocess erstellt und damit, sobald der Childprozess getötet wird, nicht der Parent Process mitstirbt, das Kommando mit dem Parameter preexec_fn=os.setsid vorher ausgeführt. 15 Sekunden später wird die ganze Prozess Gruppe getötet. Mit dem gesendeten Signal SIGTERM wird die fim Ansicht beendet und man kehrt zur Shell zurück.

```
1  #!/usr/bin/env python3
2
3  from gpiozero import Button
4  from time import sleep
5  import subprocess
6  import os
7  import signal
8
9  key1 = Button(18)
10 key2 = Button(23)
11 key3 = Button(24)
12
13 while True:
14         if key1.is_pressed:
15                 p = subprocess.Popen('exec␣fim␣/
                       ↪ home/pi/qrcode.png', shell=
                       ↪ True, preexec_fn=os.setsid)
16                 sleep(15)
17                 os.killpg(os.getpgid(p.pid),
                       ↪ signal.SIGTERM)
18         if key2.is_pressed:
19                 subprocess.run(['changePassword.py
                       ↪ '])
20         if key3.is_pressed:
21                 os.system('clear')
```

8 Fazit und Ausblick

8.1 Fazit

Die hier vorgestellte Implementierung war sehr erfolgreich. Der Raspberry Pi kann als Access Point verwendet werden und wechselt seinen Schlüssel automatisch. Dieser wird dann auf dem Display angezeigt. Somit sind die angegebenen funktionalen Anforderungen erfolgreich umgesetzt.

Ein Großteil der optionalen Funktionalitäten sind ebenfalls umgesetzt. Das

Passwort kann zusätzlich mit einem Tastendruck zu einer beliebigen Zeit gewechselt werden. Weiterhin wurde das Setup Skript erstellt. Dieses vereinfacht das nochmalige Aufsetzen des Projekts sehr. Bei Bedarf müssen lediglich die im Repository [12] gezeigten Schritte durchgeführt werden, um den Access Point zu konfigurieren. Die Energiespar-Funktionalität wurde in diesem Projekt nicht umgesetzt.

Die Tests mit verschiedenen Endgeräten offenbarten keine Probleme mit den verschiedenen Beriebssystemen. Gäste können sich nun über das eingerichtete Gäste-WLAN sich erfolgreich anmelden und im Internet surfen.

8.2 Ausblick

Im Folgenden nennen wir einzelne Verbesserungen und mögliche Erweiterungen des Projektes.

Eine Verbesserung ist die Einschränkung des Zugriffs auf das Heimnetzwerk. Aktuell können Gäste auf das Internet, aber auch auf die Geräte im Heimnetz zugreifen. Dieses Verhalten ist nicht immer erwünscht. Eine mögliche Lösung wäre das Erstellen weiterer Firewall-Regeln. Diese schränken den Zugriff der Gäste auf nur das Internet ein. Weiter Anfragen werden verworfen.

Weiterhin ist das Absichern der hostapd.conf Datei durch ein Passwort zu empfehlen. Diese beinhaltet das Passwort für den WLAN-Zugang im Klartext. Sobald jemand Zugriff auf diese Datei besitzt, kann dieser die Konfiguration des gesamten Gastnetzes verändern. Nach Absicherung der Datei muss in diesem Zuge das Skript angepasst bzw. bearbeitet werden.

Eine weitere Verbesserung ist ein dediziertes WLAN-Modul, denn derzeit limitiert der Raspberry Pi die Frequenz auf 2,4GHz [8]. Das Upgrade auf 5 GHz ermöglicht eine höhere Datenübertragung, falls dies benötigt wird. Hierbei sollte auch darauf geachtet werden, dass das Modul die Implementierung, der automatischen Channel Auswahl von Hostapd, unterstützt.

9 Abbildungsverzeichnis

List of Figures

10 Quellenverzeichnis

References

[1] Ben Nuttall, 2020. https://gpiozero.readthedocs.io/en/master/index.html [aufgerufen am 11.12.2020].

[2] Chaitanya T K, 2020. https://wireless.wiki.kernel.org/en/users/documentation/acs [aufgerufen am 10.12.2020].

[3] Das Debian-Projekt, 2020. https://manpages.debian.org/testing/hostapd/hostapd.8.en.html [aufgerufen am 26.11.2020].

[4] Das Debian-Projekt, 2020. https://wiki.debian.org/dnsmasq [aufgerufen am 12.12.2020].

[5] Das Debian-Projekt, 2020. https://packages.debian.org/de/source/sid/iptables-persistent [aufgerufen am 12.12.2020].

[6] Das Debian-Projekt, 2020. https://packages.debian.org/de/sid/netfilter-persistent [aufgerufen am 12.12.2020].

[7] Elektronik-kompendium, 2020. http://www.elektronik-kompendium.de/sites/raspberry-pi/bilder/19052512.jpg [aufgerufen am 25.11.2020].

[8] Elektronik-kompendium, 2020. http://www.elektronik-kompendium.de/sites/raspberry-pi/2102291.htm [aufgerufen am 25.11.2020].

[9] Gentoo Foundation, Inc., 2020. https://wiki.gentoo.org/wiki/Hostapd#Capabilities_of_Hostapd [aufgerufen am 26.11.2020].

[10] International Organization for Standardization, 2020. https://www.iso.org/obp/ui/#iso:code:3166:DE [aufgerufen am 11.12.2020].

[11] joy-it, 2020. https://joy-it.net/files/files/Produkte/RB-TFT3.2-V2/RB-TFT-Manual_04082020.pdf [aufgerufen am 11.12.2020].

[12] Luca Asmus, Rolf Wiersch, Marius Würstle, 2020. https://github.com/1xca/QRRaspAP [aufgerufen am 11.12.2020].

[13] Michael Nooner, 2020. https://pythonhosted.org/PyQRCode/ [aufgerufen am 12.12.2020].

[14] Michele Martone, 2020. https://www.unix.com/man-page/debian/1/fim/ [aufgerufen am 12.12.2020].

[15] Raspberry Pi Foundation, 2020. https://www.raspberrypi.org/documentation/configuration/wireless/access-point-routed.md [aufgerufen am 26.11.2020].

[16] Reichelt, 2020. https://cdn-reichelt.de/bilder/web/artikel_ws/A300/TFTV2.jpg [aufgerufen am 26.11.2020].