Fakultät Elektrotechnik, Feinwerktechnik und Informationstechnik

RaspberryPi CyberSec Lab: Development of a Penetration Testing Platform

Bachelorarbeit im Studiengang Elektro- und Informationstechnik

vorgelegt von

Matrikelnummer

Erstgutachter:

Zweitgutachter:

Abstract

This thesis documents the development and testing of the Raspberry Pi Cybersecurity Laboratory (RCSL). The RCSL is intended as a testing platform for demonstration and practice purposes of cybersecurity in various IT applications.

An expandable software architecture is developed, which focuses on usability and includes features for the creation of test environments in the Wi-Fi and web application domain. Development and functionality are documented, highlighting the challenges encountered along the process.

The RCSL's Wi-Fi functionality is tested by performing attacks on the WEP and WPA/WPA2 network standards. Several faults in the system were discovered, which prohibited the execution of the tests. After elimination of the errors, cracking of both standards was successfully performed multiple times. The results show the RCSL to be capable of establishing testing environments for Wi-Fi security.

Proposed usecases for the RCSL include demonstration and hands-on practice in cybersecurity and software development education. Future improvement and expansion of the project are recommended.

Kurzdarstellung

Diese Arbeit dokumentiert die Entwicklung und Erprobung des Raspberry Pi Cyber Security Laboratory (RCSL). Das RCSL ist als Testplattform für Demonstrations- und Übungszwecke von Cybersecurity in verschiedenen IT-Anwendungen gedacht.

Es wird eine erweiterbare Software-Architektur entwickelt, die sich auf die Benutzerfreundlichkeit konzentriert und Funktionen für die Erstellung von Testumgebungen im Wi-Fi- und Web-Anwendungsbereich enthält. Entwicklung und Funktionalität werden dokumentiert und die dabei aufgetretenen Herausforderungen aufgezeigt.

Die Wi-Fi-Funktionalität des RCSL wird durch Angriffe auf die Netzwerkstandards WEP und WPA/WPA2 getestet. Es wurden mehrere Fehler im System entdeckt, die die Durchführung der Tests verhinderten. Nach Beseitigung der Fehler wurden beide Standards mehrfach erfolgreich geknackt. Die Ergebnisse zeigen, dass das RCSL in der Lage ist, Testumgebungen für die Wi-Fi-Sicherheit einzurichten.

Vorgeschlagene Anwendungsfälle für den RCSL sind Demonstrationen und praktische Übungen in der Cybersecurity- und Softwareentwicklungslehre. Zukünftige Verbesserungen und Erweiterungen des Projekts werden empfohlen.

Contents

Chapter 1.

Introduction

The modern world is moving towards digitalization at a high pace, which aims to make it smart and interconnected. This transformation comes with the ever-increasing use of connected devices and services, which oftentimes carry sensitive data and are not always protected sufficiently. As a result, there is a rising number of cybercrime incidents and subsequent damages estimated to more than $1 trillion worldwide.

Compounding on top of the situation is a mismatch between demand and supply of cybersecurity professionals and a public, which is in large parts not aware of the risks posed by the use of connected IT technology. Cybercriminals can often easily exploit these unaware people to gain access to restricted systems and data. To combat this problem, there's a need for training cybersecurity professionals, as well as to raise awareness at every level of education and the public. As a method for raising awareness, teaching principles of cybersecurity and the methods to apply them, live demonstration and hands-on learning with realistic risk scenarios can be an effective tool. [Mari 24][Cybe 23]

1.1. Goal

The goal of this project is to develop a platform with a flexible and expandable architecture, which can be used for demonstration and practice of Pentesting[1] in cybersecurity education. A Raspberry Pi will be used as the main device to create a platform that is portable and easy to use. It should be able to reliably set up testing environments for various fields of application, to allow a lecturer to demonstrate cyberattacks and students to practice penetration testing in a realistic environment. In recent years, trends like the Internet of Things (IoT) have produced many headless[2] devices, which are often linked to smartphones for user interaction, using Wi-Fi or Bluetooth for communication. For this reason, wireless communication security in the form of Wi-Fi technology will be implemented as the first application specific testing environment. To teach secure design of web applications, the

[1] penetration testing, see section 2.2.2
[2] without any means for direct interaction

device should be able to host a server with the OWASP Juice Shop[3], which again ties into the security of IoT devices, because they are oftentimes accessed via web interfaces, which are often the source of vulnerabilities [Hell 23, page 174].

1.2. Overview

- chapter 2 explains the difference between IT security and cybersecurity and outlines the principles and encountered threats.
- chapter 3 explores the history of Wi-Fi technology and explains its architecture, function and security mechanisms.
- chapter 4 documents the hardware and software of the RCSL.
- chapter 5 outlines the development process and the challenges faced along the process.
- chapter 6 explains the penetration tests conducted and documents the results of testing them on the RCSL.
- chapter 7 outlines the challenges of cybersecurity education and proposes use cases for the RCSL.

[3] is a web security training program, see section 4.2.6

Chapter 2.

Basics of Cybersecurity

This chapter briefly explains the terms of IT security and cybersecurity and gives an overview of their goals and methods. A summary of common threats in the cyberspace is given as well, giving context for chapter 6.

2.1. IT Security

IT Security describes the protection of digital data and computer- and communication systems from unauthorized access. Different principles are adhered to and methods and techniques applied in order to prevent the theft, interception, manipulation, and loss of data and systems. [Pogu 13]

2.2. Cybersecurity

IT- and cybersecurity are similar in that both have the same goals and apply the same principles and methods. While IT Security focuses on classical computer systems and networks, cybersecurity focuses on data and systems in the cyberspace. Included in the cyberspace are all systems and data connected to the internet.

2.2.1. Goals

The goals of cybersecurity include but are not limited to the well-known CIA triad, which is an acronym for Confidentiality, Integrity, and Availability [Oriy 17]. Confidentiality describes data only being accessible by the persons who are meant to do so. Integrity describes data being unchanged after storage or transmission, meaning the data was not corrupted or manipulated. Availability describes data and systems being accessible with the expected performance, whenever required.

2.2.2. Methods

Some methods for achieving the goals of cybersecurity are:

- Security Awareness is the knowledge of the threats present in the cyberspace and methods for protection. "Awareness of the risks and available safeguards is the first line of defence for the security of information systems and networks" [Pogu 13, page 26].
- Encryption is a cryptographic[1] method, which takes the readable data, called plaintext, and makes it unreadable for anyone besides the legitimate recipient. The encrypted text is called ciphertext and can be made readable again by decrypting it. To en- and decrypt a message, the sender and recipient have knowledge of a secret, often a password, which controls en-/decryption and is called key. [Watj 18, page 1]
- Authentication is the act of a user "performing some sort of action which proves that the claim a user is making about who they are is actually valid" [Oriy 17]. This is oftentimes done through the use of a password, in the case of Wi-Fi networks, the knowledge of the password proves that a user is allowed to access the network.
- Pentesting is a method to audit the security of a system or network. It involves a security professional performing various types of attacks to discover weaknesses in soft- and hardware.
- Monitoring with anomaly detection can be used to detect the unauthorized access of systems and networks. In the network domain, so-called Intrusion Detection Systems (IDS) employ predictive models to recognize malicious activity inside a network [Geekb].
- Incident Response is performed in case of a system breach in order to limit damage and secure the system.

2.3. Threats and Risks in the Cyberspace

There are various types of cybercriminals with different motives and intentions. Motives of cybercriminals can vary, but commonly are financial, political, social, or military. With these motives the intentions of an attack often are:

- blackmail
- espionage
- theft
- fraud
- sabotage
- corruption and insider deals

[1] cryptography = science of secretive writing

The success rate and risk of an attack are in large parts determined by a target's attack area, which is the combination of the present vulnerabilities and attack vectors. The term vulnerability describes a weak spot, which can be exploited by the attacker to break or circumvent security mechanisms. An attack vector is the approach an attacker takes for carrying out an attack. In regard to network security some common attack vectors are:

- Denial of Service (DoS) attacks availability by preventing or limiting the functional capability of a system.
- Cracking describes the act of breaking or circumventing security mechanisms, e.g. reverse engineering passwords.
- Brute-Force is a technique for reverse engineering passwords by trail-and-error.
- Dictionary attack is a variation of Brute-Force, using a collection of leaked passwords to guess the correct solution.
- Spoofing refers to faking identities, achieved by the obfuscation of the own identity or the replication of another identity.
- Sniffing is the action of collecting all openly available data, e.g. recording unencrypted Wi-Fi frames.
- Man-in-the-Middle (MitM) attacks involve injection of a malicious device into a line of communication, which subsequently can manipulate and eavesdrop on the traffic.
- Social Engineering is a form of manipulation that abuses psychological mechanisms like trust, curiosity, or shock to exploit victims.

[Pogu 13, page 16-20]

Chapter 3.

Basics of Wi-Fi

This chapter explains the basics of Wi-Fi technology, needed to understand chapter 6. Insight into the history and development of Wi-Fi is given and its functionality and security mechanisms are briefly explained.

3.1. History

In 1985 the American Federal Communications Commission ruled several bands in the 2.4 GHz radio band to be unlicensed, meaning everyone can broadcast radio signals in these frequencies. At the beginning of the 90s, this led to the development of different wireless communication protocols by multiple companies. To ensure interoperability between these protocols, the IEEE 802.11 standard was created and released, and WLAN was born. In 1999 the Wi-Fi Alliance (WFA) was founded to further improve compatibility, nowadays consisting of over 750 international companies. [Sank 21, page 1-3], [Wiki]

Since its inception over 25 years ago, Wi-Fi has gone through constant development, leading to six major generations of the technology. Improvements in speed, efficiency, and security have led to the technology becoming a "fundamental utility [...], that everyone [...] expects to be available everywhere" [Sank 21, page 1]. Wi-Fi connects billions of devices, like computers, smartphones, game consoles, or IoT devices to the internet, generating a value of around 4.9 trillion USD worldwide [Sank 21, page 1].

Wi-Fi operates at the physical (PHY) and the medium access control (MAC) layer of the OSI model[1], sending Ethernet frames[2] over radio waves in the 2.4 and 5 GHz frequency band. The MAC layer is responsible for setting rules on transmission and reception of data, connection management and power management. Data sent on the MAC layer then gets converted to radio waveforms by the PHY layer, which is also responsible for converting received data back to the MAC protocol. [Sank 21, page 4-7]

[1] describes the components of the internet communication protocol
[2] see section 3.2.2

3.2. MAC layer

Devices in the MAC layer are addressed with a MAC address, which consists of 12 hexadecimal characters and is unique for every network interface controller (NIC).

A MAC address could look like this: **`61:A3:E2:73:9A:F3`**.

3.2.1. Architecture

Commonly, Wi-Fi networks have access points (AP), which are connected to the internet via a wired ethernet connection, and stations (STA), which are provided with internet access over Wi-Fi by the AP. Wi-Fi has a MAC layer, which is designed to be flexible and can be configured in the following topologies:

- **BSS**: Basic Service Set
- **IBSS**: Independent BSS / peer-to-peer
- **WDS**: Wireless Distribution System
- **MBSS**: Mesh BSS

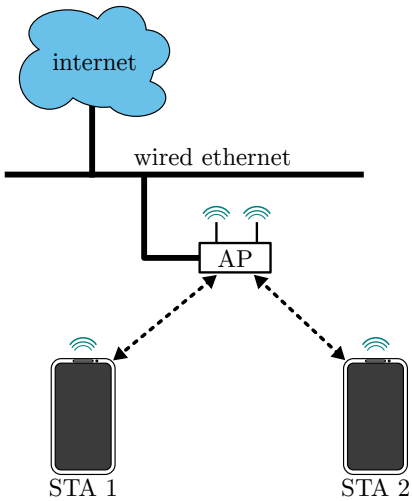


Figure 3.1.: BSS Wi-Fi network configuration

BSS describes the most common and basic configuration, where multiple STAs are connected to one AP, which has wired connection to the internet, as shown in fig. 3.1. An AP's address is called the Basic Service Set Identifier (BSSID) and the networks name, visible from the outside, is called the Service Set Identifier (SSID). This configuration is used for the RCSL and therefore all other configurations are less relevant for the project, but additional information can be found in appendix A.2.1.

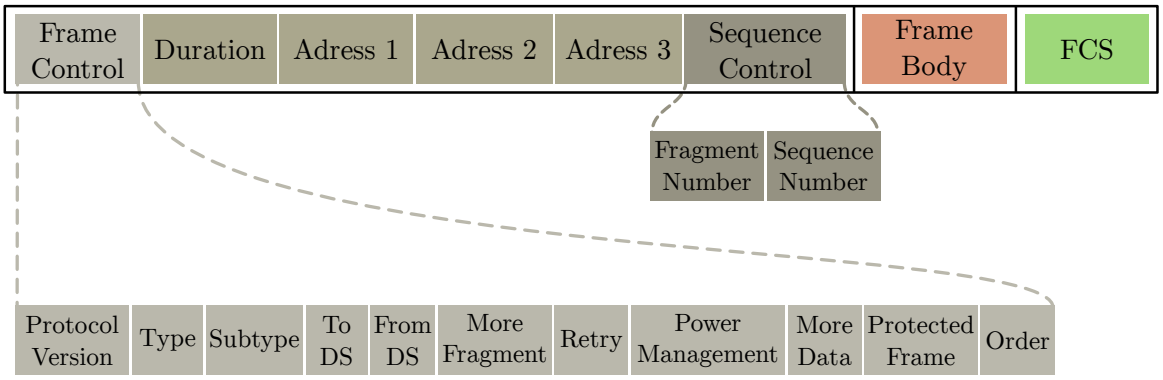


Figure 3.2.: MAC Frame of BSS Wi-Fi networks

3.2.2. MAC Frames

When data is sent over the MAC layer, it gets divided into chunks, which are called MAC Frames (see fig. 3.2). MAC Frames have three components: the header, the body, and the frame check sequence. The header contains information about the source and destination, the sequence and several control functions, which is described in detail in appendix A.2.2.

The frame body then contains the payload, meaning the actual information to be sent. Every type of MAC frame is configured a different way to suit their specific function. "While data frames are meant to carry data, management frames are used for connection management and control frames [...] for triggering specific actions. Control frames have a smaller MAC header and no payload data [...]. Management frames have a structured payload composed of some fixed length fields and one or more variable length fields called information elements (IEs)" [Sank 21, page 12]

After the frame body comes the frame control sequence (FCS), which ensures the integrity of the frame. The FCS is a number calculated from the frame header and body content. When the frame is compromised in traffic, the result of the FCS changes, therefore the faulty frame can be identified. [Geeka] [Sank 21, page 10-12]

3.2.3. Medium Access

In LAN and WLAN networks, all devices share the same medium, therefore only one device at a time can transmit information. Ethernet networks use full-duplex transceivers, meaning a device can transmit and receive data at the same time. This allows the transceiver to transmit and simultaneously listen for collisions with other frames. Collision detection is quite effective, but can not be used in Wi-Fi networks because Wi-Fi transceivers are only capable of half-duplex transmission. Wi-Fi instead uses a medium access control referred to as carrier sense multiple access collision avoidance (CSMA-CA).

When using this technique, the medium gets blocked during transmission and is freed by an acknowledgment frame, sent by the receiving device. All STAs start off in the random backoff state, where they listen for traffic. After the random backoff counter has expired and no traffic is present, the STA will start transmission. CSMA-CA can be described by the way "humans converse in a group, where every individual would first listen to check if someone else is talking, wait for the person talking to finish, and then start talking after a brief random pause" [Sank 21, page 12]. [Oriy 17] [Sank 21, page 12-17]

For addressing a STA, its MAC address needs to be known. For sharing the MAC address of a STA with all other STAs on the network, the Address Resolution Protocol (ARP) is employed, which broadcasts the STAs MAC addresses, allowing all other devices to save it.

3.2.4. Connection Establishment and Termination

Before a connection can be established, the STA first needs to discover the AP it wants to connect to. This can be done by either active or passive scanning of all available channels.

During active scanning, the STA transmits a management frame, called probe requests on all channels to all SSIDs in the vicinity. The APs answer with a probe response frame, which contains the AP's SSID, BSSID and its capabilities.

Passive scanning works by the STA listening for beacon frames. Beacon frames are periodically sent out by APs and also contain information on SSID, BSSID and capability. [Sank 21, page 18]

Once the desired network is discovered, the STA decides which AP to connect to, based on parameters, like the receive signal strength indicator (RSSI) or the APs capacity, performance and security. The STA then sends an authentication request, which the AP gives an authentication response to. If the credentials of the STA are valid and the authentication is successful, the STA proceeds to send an association request. The association response, sent by the AP, then indicates if the connection was established successfully.

To terminate a connection, either the STA or the AP can send a disassociation or deauthentication frame, "containing a reason code that specifies the reason for termination" [Sank 21, page 20]. [Sank 21, page 19]

3.3. PHY layer

The Wi-Fi PHY layer is based on radio frequency (RF), therefore Wi-Fi transceivers contain a transmitter and receiver circuit, as well as an antenna. When sending data, MAC frames

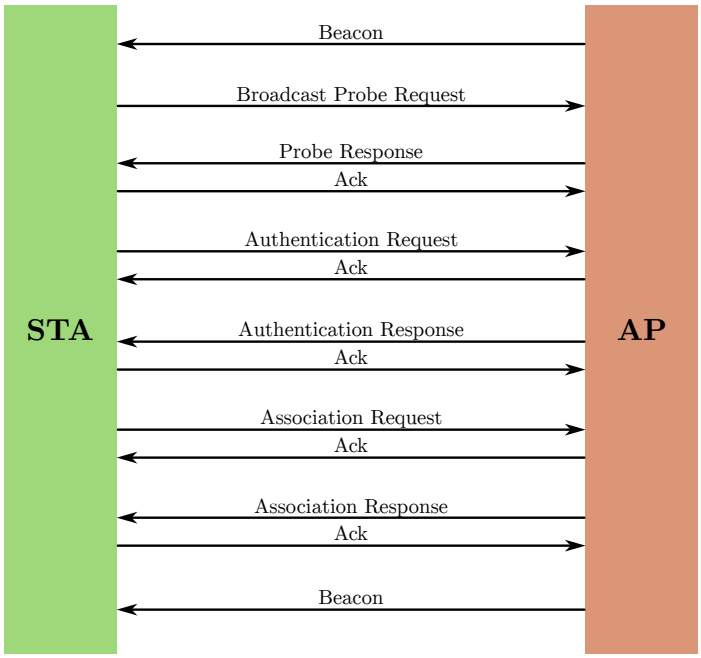


Figure 3.3.: Connection Establishment

are encoded and modulated onto an RF waveform, which is then transmitted via the antenna. On arrival of data, the receiver circuit decodes the data, extracts the MAC frame, and provides it to the MAC layer. The first generations of Wi-Fi used to transmit in the 2.4 GHz band, but due to it being used by many other devices, such as cordless phones, computer peripherals, Bluetooth devices or microwave ovens, Wi-Fi 4 started using the 5 GHz band, in order to improve connectivity. [Sank 21, page 5,34,35] With every generation of Wi-Fi, the PHY layer components are modified to improve performance and efficiency, but due to the PHY layer not being majorly significant for cybersecurity, it will not be described further in this thesis.

3.4. Security

Through the widespread adoption of Wi-Fi technology, for example in smart homes, infrastructure, public spaces, industry, and governments, the security of Wi-Fi networks plays an ever increasing role. Therefore, principles of cybersecurity need to be applied in order to guarantee that networks are safe, reliable, and performant. Because of its wireless nature, "[i]t is difficult to confine the data-carrying radio waves to be within the physical security perimeter of [an] organizsation" [Sank 21, page 103], so anyone in the vicinity of a Wi-Fi network could just intercept and manipulate network traffic. To ensure integrity and confidentiality, Wi-Fi two main security mechanisms, which are encryption and authentication. [Sank 21, page 103]

Note: Since this thesis is focused on private networks, the following summary of Wi-Fi security mechanisms will not include enterprise authentication protocols.

3.4.1. Encryption

The first Wi-Fi security standard, called Wired Equivalent Privacy (WEP) uses a shared key, with a RC4 stream cipher algorithm for message encryption and a 32-Bit cyclic redundancy check (CRC) as an integrity check (IC). Due to several vulnerabilities of WEP, including the RC4 algorithm, a lack of authentication and no key management, the second standard, Wireless Protected Access (WPA), was developed.

WPA still uses the RC4 cipher for compatibility reasons, but the encryption mechanism was updated, now called Temporal Key Integrity Protocol (TKIP), it uses a pairwise master key, which is generated for each STA after it has authenticated successfully. TKIP also dynamically changes the encryption key for each packet, which in combination with a longer 128-Bit pre-shared key (PSK) fixed some of the weaknesses of WEP. [Oriy 17]

WPA2 could be considered as the first proper Wi-Fi security standard since it replaced the weak RC4 algorithm with AES-CCMP encryption. AES-CCMP encrypts data blockwise and uses a combination of substitution and transposition algorithms to achieve high security. WPA2 security also added protected management frames, so not only data frames, but also management frames are encrypted. This helps to secure networks against DoS attacks with deassociation or deauthentication frames.

The latest Wi-Fi security standard is WPA3, which aims to fix vulnerabilities of WPA2 and future-proof the technology. Most changes for WPA3 have been made in authentication mechanisms, WPA3 added the optional support of AES-GCMP encryption, which is more performant and safer than AES-CCMP. [Sank 21, page 103-117]

3.4.2. Authentication

WEP authentication is only one-sided, where STA has to encrypt a package with the correct key, in order to complete association with the AP. The same key is used for encryption, making the whole system vulnerable when either the authentication or the encryption is exploited.

WPA and WPA2 networks use an authentication mechanism that is two-sided, where STA and AP exchange identity and capability. After entering the correct passphrase a password-based key derivation function then generates the PSK for encryption. This works by deriving the PSK "as a function of the Wi-Fi password, SSID, and SSID length using a password-based key derivation function" [Sank 21, page 106].

Because the PSK is transmitted during the handshake on connection establishment, PSK is potentially vulnerable to offline and dictionary attacks when a weak passphrase is used. To future-proof the technology, WPA3 replaces PSK with Simultaneous Authentication of Equals (SAE). SAE works by choosing an element from a finite cyclic group (e.g., a point on an elliptic curve) based on the password and performing a Diffie-Hellman exchange to generate a pairwise master key. This mechanism allows authentication without the transmission of any information related to the password and is performed for every session to ensure forward security. [Oriy 17], [Sank 21, page 115-116], [Hark 08]

Note: A full summary of the different security specifications of each Wi-Fi generation, can be found in table A.1.

Chapter 4.

Documentation

This chapter documents the hardware and software of the Raspberry Pi Cybersecurity Lab (RCSL), wherein the hardware section describes the hardware components, which the device is built from, and the software section explains the architecture of the software and its functionality.

4.1. Hardware

The figure below shows the hardware components used in the project, which are a Raspberry Pi (RPI), a USB network card, an ESP32 microcontroller, a display and a rotary encoder.

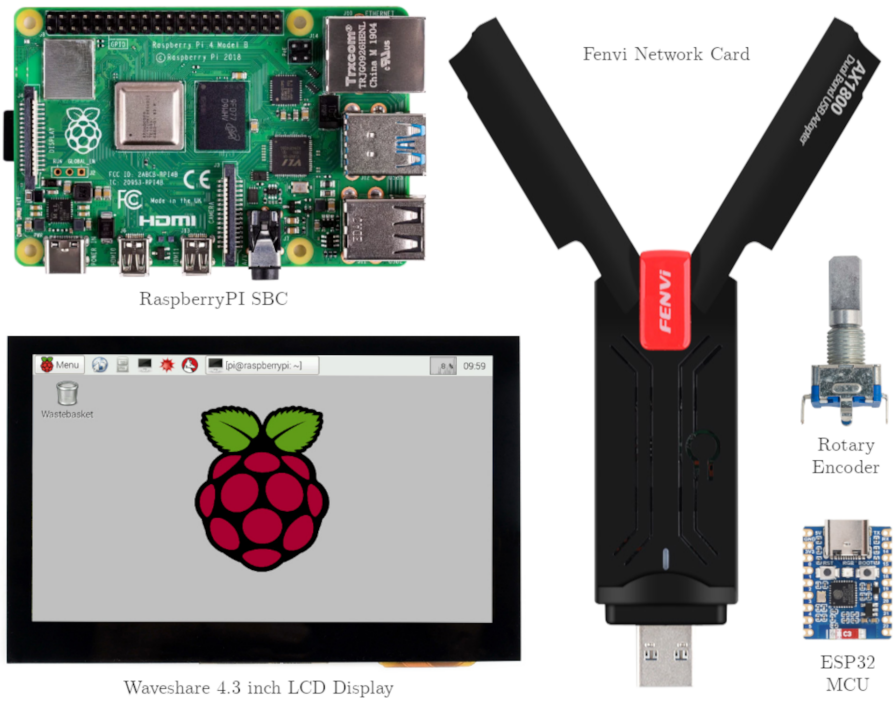


Figure 4.1.: Hardware Components

As the main computer, the Raspberry Pi 4B with 8GB of RAM was chosen due to its ease of use, Linux based operating system and widely available documentation.

The RPI has an integrated Wi-Fi and Bluetooth card, as well as onboard Ethernet networking. However, to support all Wi-Fi standards, an external network card is needed since the internal card of the RPI does not support the ciphers used by WPA3. The Fenvi AX1800 USB Wi-Fi card was chosen for its compatibility with the Linux kernel, according to [Morr], and low cost.

For some types of attack, such as cracking WPA/WPA2[1], a Wi-Fi station, which is connected to the test network is needed, which in this case is an ESP32 microcontroller unit (MCU). Selecting from the large offer of ESP32 MCUs, again size was taken into account and the ESP32-C6 Mini by Waveshare was chosen for its small size and good feature set. It connects to the RPI's GPIO[2] pins using jumper cables (see fig. A.2), which supply the ESP32 with power and establish an UART[3] connection.

Another device connected to the RPI's GPIO header is the rotary encoder. On its five pins, it receives 3.3V supply voltage and outputs a gray code, indicating the direction of rotation, on two of its outputs. The fifth pin is used as the input for the push-button action of the encoder.

The hardware components are mounted in a 3D printed case, shown in fig. 4.2.

Figure 4.2.: RCSL within its case

[1] see chapter 6
[2] General Purpose Input/Outputs can be configured to read or write data
[3] Universal Asyncronous Receiver/Transmitter is a bus standard for serial communication between embedded devices

4.2. Software

Note: The project does not focus on performance or software quality. It is written to implement the wanted functionality and should be seen as experimental, since it is neither optimised for performance nor have proper error handling mechanisms been implemented.

4.2.1. Overview

The software of the project is mostly modular, and it consists of a central program, which runs the user interface (UI) and then executes various shell scripts[4], with which the functionality is implemented. As can be seen in fig. 4.3, besides the UI program, there are programs for MQTT communication and input processing, also the services of the Network Manager and the Juice Shop project are used.

Note: Only the UI program is written object-oriented and only Menu and MenuOption are object classes.

4.2.2. UI Program

Upon booting the device, a systemd service executes start.sh, which starts the encoder program and the UI program, with the output of the encoder program piped into the input of the UI. The UI displays the main menu, depicted in fig. 4.4, which the user can interact with by using the rotary encoder. Each page of the UI is a "Menu" object, each menu has a title, options stored inside a vector of the type "MenuOption", a variable to track the currently selected option, a method to add options to a menu, and methods for navigation, input capture and displaying. MenuOption objects have a name, as well as an action of the std::function type, which can be executed by calling the execute() method. In the main of the program, the constructors for menus and the addOption() methods are being used to create the different menus and their submenus. An overview of the menus and options, currently present inside the UI program can be seen in appendix A.3.2. The program starts by calling the navigate() function of the main menu, which then displays all available submenus and menu options. All submenus work in the same way as the main menu, getting their respective navigate() called upon selection. During navigation, the UI periodically prints the available options, using Unicode to highlight the currently selected option as well as to give contrast to the title of the menu (see fig. 4.4).

More examples for the UI can be found in the appendix, for example, fig. A.3.

[4] chain of commands to execute in the terminal of a Linux/Unix system

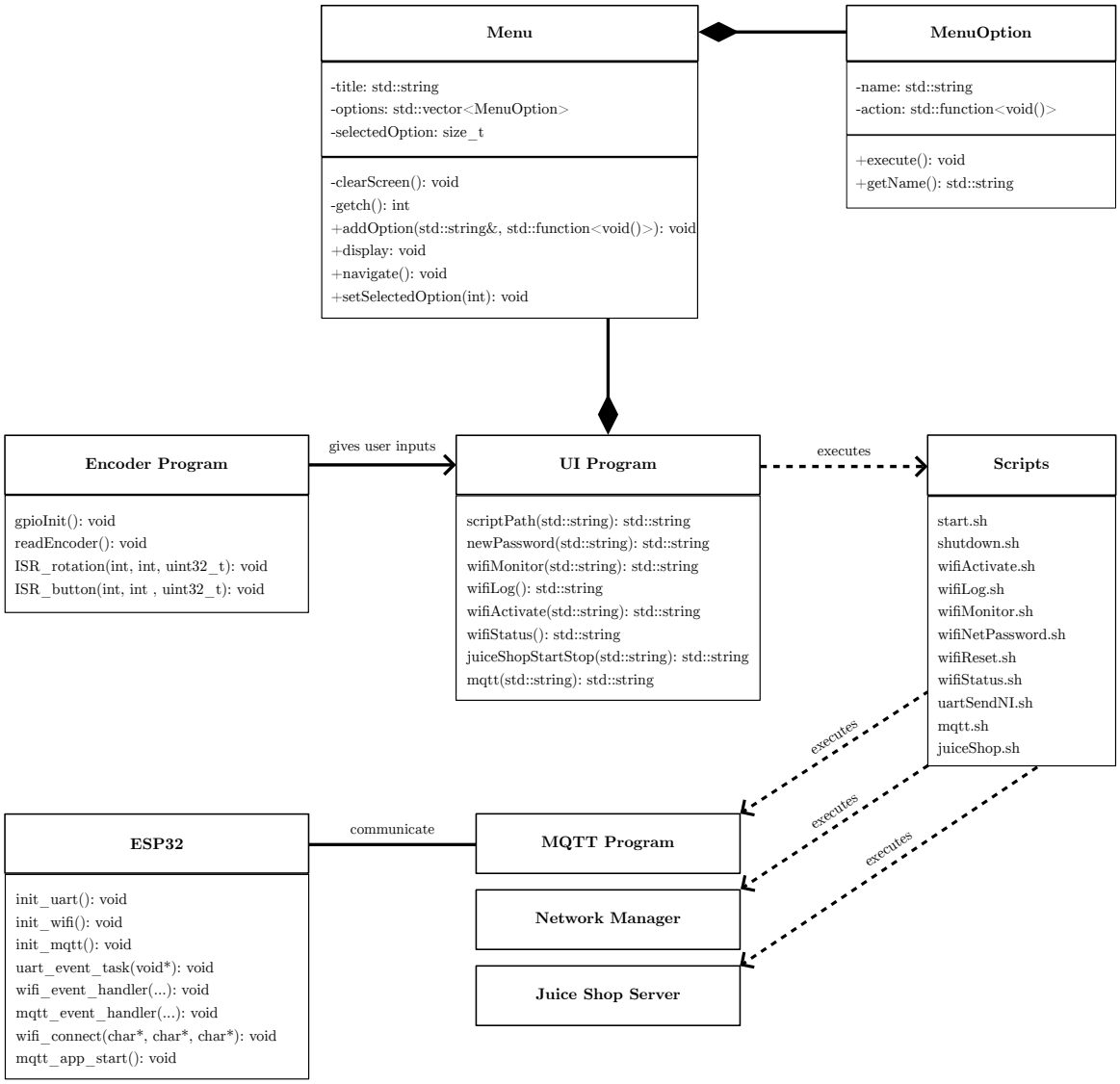


Figure 4.3.: Software Architecture

When an action, like activation of a network, is executed, main calls the according shell script with the system() function (see lis. 4.2). Scripts, which require an input, are called with a specific function that pipes the according information into the script (see lis. 4.1).

```
1 std::string wifiActivate(std::string connection){
2     std::string command = "echo " + connection + " | " + scriptPath("wifiActivate");
3     return command;
4 }
```

Listing 4.1: wifiActivate function in the UI program

```
1       wifiActivateMenu.addOption("WEP", [](){system(wifiActivate("WEP").c_str());});
```

Listing 4.2: use of wifiActivate for WEP network

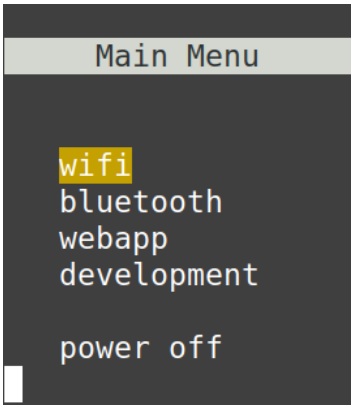


Figure 4.4.: Main Menu

4.2.3. Encoder Program

For input, the RCSL uses a rotary encoder with a push-button, turning the encoder moves the menu selection up or down, the push-button executes the selected option (see fig. 4.5). The encoder is connected to the Raspberry Pis GPIO interface, its rotation is segmented with clicks, similar to the scroll wheel of a computer mouse. Like shown in fig. 5.2, rotation between two clicks outputs one complete Gray code sequence on the signals DT and CLK, which is processed by the encoder program. This happens within the rotation handling interrupt service routine (ISR), shown in lis. 4.3, which is executed when DT outputs a high-low transition.

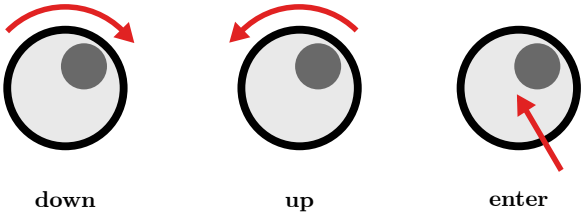


Figure 4.5.: Navigation Scheme

The ISR checks the state of CLK, which determines the direction the encoder was turned: CLK=0 means the encoder has been turned clockwise, CLK=1 means counter-clockwise (see fig. 5.2). Subsequently, a command in the form of a string, representing an ANSI Escape Code, is printed. Escape Codes are used as an output for peripheral buttons, which are not a character, for example, `"\033[A"` describes the pressing of the downwards button on the keyboard. As previously explained, the output of the encoder program is piped into the input of the UI, therefore rotating the encoder leads to the UI receiving the codes for up and down. The push-button action is processed similarly, with an ISR outputting a newline symbol to the UI.

```
void ISR_rotation(int gpio, int level, uint32_t tick) {
    if (level == PI_LOW) {
        volatile int clk = gpioRead(CLK);
        if (clk == 0) {
            std::cout << "\033[A" << std::flush;  //Arrow Down
        } else {
            std::cout << "\033[B" << std::flush;  //Arrow Up
        }
    }
}
```

Listing 4.3: Rotation handling ISR

Because the commands used for navigation are ordinary ANSI escape codes, the program can be easily tested on any computer by navigating the UI with the keyboard.

4.2.4. Shell Scripts

The wifiActivate.sh script is used to open Wi-Fi hotspots, where the user can choose the type of Wi-Fi standard being used. It receives an input command and then establishes a WEP, WPA, WPA2 or WPA3 network. On first use, it sets up a network configuration, which just gets activated on repeated use, as seen for WEP networks in the code extract 4.4.

```
#!/bin/bash

scriptdir="$(dirname "$0")"

read -n 4 connection

if [ "$connection" == "WEP" ]; then
  exists=$(nmcli connection show | sed -n '/Wifi-WEP /p')
  if [ -z "$exists" ]; then
    sudo nmcli device wifi hotspot con-name "Wifi-WEP" ssid "WEPnetwork" password "
        TestSetup123" ifname wlan1
    sudo nmcli connection down Wifi-WEP
    sudo nmcli connection modify Wifi-WEP 802-11-wireless-security.key-mgmt none
    sudo nmcli connection modify Wifi-WEP 802-11-wireless-security.group wep104,
        wep40
    sudo nmcli connection modify Wifi-WEP 802-11-wireless-security.auth-alg shared
    sudo nmcli connection modify Wifi-WEP 802-11-wireless-security.wep-key0 "test1"
    sudo nmcli connection modify Wifi-WEP 802-11-wireless-security.pmf 1
  fi
    sudo nmcli connection up Wifi-WEP
```

Listing 4.4: extract of wifiActivate.sh

After a Wi-Fi hotspot has been activated, a command with the connection details is sent to the ESP32 over UART. This functionality is implemented by the uartSendNI.sh script, which gets called by wifiActivate.sh (see line 17 of lis. 4.5). Receiving the network information, the ESP32 will then connect to the provided network and act as a communication partner for the RPI.

With the wifiReset.sh script, the currently active network connection is turned off, which gets called by the deactivate option in the Wi-Fi menu.

To display the current status of the network, meaning its security standard, SSID and password, the wifiStatus.sh script is used. It uses the **`nmcli dev wifi show-password`** to output to the UI, which then looks like the following:

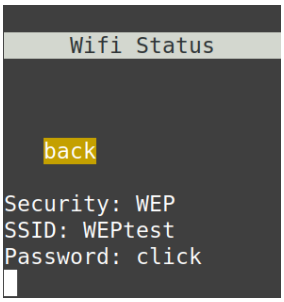


Figure 4.6.: Wi-Fi status menu

When activating monitoring, wifiMonitor.sh uses the **`iw event -T`** command to log devices connecting and disconnecting from the network. Selecting the "monitor" option, the user is presented with the options of de-/activating the network monitoring as well as viewing or deleting the monitoring log.

Once a network has been cracked, the wifiNewPassword.sh script can be used to give a new random password to the selected network inside the change password menu. The script then pulls a random password from a password list (see lis. 4.5), containing popular passwords, that have been filtered to work with the specification of each network standard. WEP networks use passwords with 5 or 13 characters, while WPA and WPA2 networks need passwords with at least 8 characters, and WPA3 networks work with passwords of every length.

```
1  randNumber=$((RANDOM % $(wc -l < $pwddir/passwordsWEP.txt) +1))
2  newpassword=$(sed -n "${randNumber}p" $pwddir/passwordsWEP.txt)
3  sudo nmcli connection modify Wifi-WEP 802-11-wireless-security.wep-key0
       $newpassword
```

Listing 4.5: extract from wifiNewPassword.sh

4.2.5. Bluetooth

Note: the Bluetooth features, originally planned, have not been implemented due to time constraints and may be implemented in the future.

4.2.6. Juice Shop

Inside the webapp menu, the Juice Shop and MQTT communication can be found.

Juice Shop is a project by the Open Worldwide Application Security Project (OWASP). It is "a web application with a vast number of intended security vulnerabilities, [which] is supposed to be the opposite of a best practice or template application for web developers: It is an awareness, training, demonstration and exercise tool for security risks in modern web applications" [Kimm 25]. When activating the Juice Shop, the juiceShop.sh script will start up the server for the OWASP Juice Shop project and open a hotspot, which can be used to access the simulated webstore.

Note: further information about the project can be found in [Kimm 25], which is the official companion to the Juice Shop.

4.2.7. MQTT Communication

When a Wi-Fi network has already been activated from the Wi-Fi menu, a simulated MQTT communication can be started from the MQTT menu. Starting a conversation will again send a command via UART to the ESP32, prompting it to connect to the MQTT broker, running on the RPI. Following up, uart.sh starts the UART program, which begins an infinite MQTT conversation with the ESP32.

4.2.8. ESP32

The code for the ESP32 is based on the code examples provided by Espressif Systems on the official ESP-IDF GitHub [Syst]. It includes event handling for UART, Wi-Fi and MQTT and a client for MQTT. The UART event handler has been modified to process the commands sent by the RPI and passes connection information to the Wi-Fi and MQTT instances.

4.2.9. Miscellaneous

The development menu contains options used during development, these can be adjusted on the fly inside the development.sh script.

When selecting the shutdown option, all running processes are stopped, and the shutdown.sh script calls **`shutdown now`** to turn off the device.

Chapter 5.

Development

5.1. Project Inception

The initial concept for the project revolved around the use of a collection of SD-Cards, containing operating systems tailored for specific use cases. The user would be to able insert the wanted SD-Card into the RPI, which then would execute a script on startup to configure the operating system for a specific scenario.

With the usability of this system appearing quite cumbersome, it was instead opted to develop a user interface (UI), which would execute the wanted shell scripts during runtime. To implement the UI, it was decided on the use of a simple terminal-based interface, which would be printed directly into the shell. This approach would facilitate straightforward navigation through the use of a dial and push-button combination, providing an intuitive means for users to interact with the system. The program should be written in C or C++ due to my familiarity with both languages.

From this concept the following 3D model was created to visualize the project idea:

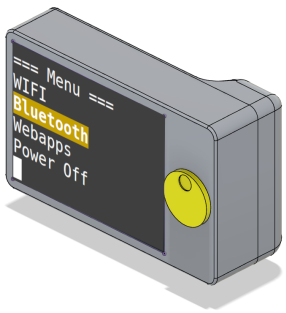


Figure 5.1.: Concept Model

5.2. User Interface

To display the UI, the menus need to be printed into the terminal, for which a simple loop, that periodically clears the screen and prints the options of a menu, was decided on. Subsequently, a fitting software architecture needed to be designed, which can model the variety of menus and their respective options with individual actions. The menus need to display options, which require the capability of executing various actions, including the transition to other menus and the execution of shell scripts and programs. Since the command design pattern at this point was unknown to me, the modeling of the menus with a uniform class turned out quite challenging. Therefore, it was decided to model the menus as classes, containing objects of the MenuOption class, and utilize the std::function for action execution.

In order for the device to be usable with the built in screen and selector dial, the RPI needs to be configured to run and display the UI program on startup. Execution of the program on startup is implemented with the help of a systemd service. For the setup, an according configuration file needs to be added to */etc/systemd/system*, which is activated by running the **`systemctl enable [service].service`** command.

Instead of a BIOS[1], the RPI employs a configuration file (*/boot/firmware/config.txt*), which is read on startup to initialize the hardware and operating system. To this file the following lines were added:

```
dtoverlay=waveshare35a
framebuffer_width=200
framebuffer_height=120
```

Line one sets up a hardware overlay to initialize the display, lines two and three set the resolution of the display. With the resolution set to the factory specification of 800x480, the UI is displayed quite small, thus poorly suited for the use case. Since the OS runs in terminal without a graphic server, there is no simple option for scaling the UI. The scale problem was fixed with a workaround, which involves the resolution inside the *config.txt* to be set to 200x120, therefore forcing the UI to scale. The increased scale leads to the menus being clearly readable and comfortable to use, but limits the readability of console output.

Initially it was planned to use the display in vertical orientation, however, attempts to rotate the terminal by configuration inside *config.txt* or the kernel command line (*/boot/firmware/cmdline.txt*) were unsuccessful.

[1] Binary Input Output System: boots before the OS

5.3. Network Management

For network management, it was decided on use of the Network Manager Command Line Interface (nmcli) tool, which is easy to use and offers a sizable set of configuration options and utilities. At the start of the development of the shell scripts for Wi-Fi activation, it was unclear, how to configure a connection in line with the specification of the different Wi-Fi standards. After research on the different security mechanisms and the nmcli documentation was conducted, the solution appeared to be the manual configuration of the encryption algorithm, key management, and information element for each connection.

The script developed for changing Wi-Fi passwords was designed to utilize a password list, from which a random password would be extracted for the connection. However, large password lists, such as the RockYou list included in Kali Linux[2], proved impractical due to their considerable size, complicating handling. A password list containing 100,000 entries was selected as the data source. Given the varying requirements for passwords across different Wi-Fi standards, as detailed in chapter 4, the list contained a number of unsuitable entries. It was subsequently filtered using a simple script (see listing 5.1) to generate password lists that conformed to the specific requirements of each standard.

```
#!/usr/bin/bash

lines=$(wc -l < 100kPasswords.txt)
i=1
while [ $i -le $lines ]; do
    currPassword=$(sed -n "${i}p" < 100kPasswords.txt)
    currCharCount=${$currPassword}
    if [ $currCharCount -ge 8 ] && [ $currCharCount -le 63 ]; then
        echo "$currPassword" >> passwordsWPA.txt
    fi
    if [ $currCharCount -eq 5 ] || [ $currCharCount -eq 13 ]; then
        echo "$currPassword" >> passwordsWEP.txt
    fi
    ((i++))
done
```

Listing 5.1: script for password extraction

For the Wi-Fi status function, the **`nmcli dev wifi show-password ifname [device name]`** command is employed to display the type of network, its SSID and password. Unfortunately this command is not functional for WEP networks, therefore a script was created, which extracts the SSID and password from */etc/NetworkManager/system-connections*, formatted to output alike the nmcli utility.

In the implementation of the Wi-Fi monitor function, the objective was to create a logger that would enable users to trace the events of an attack, such as the dis-/connection of devices and any changes to the network. The command **`iw event -T`** provides timestamps and

[2] see section 6.2

MAC addresses for devices connecting to or disconnecting from the network. Additionally, **`nmcli monitor`** reports any modifications made to the network status but does not include timestamps in its output. Attempts to add timestamps by using **`awk`** or **`ts`** were unsuccessful, as nmcli monitor must be executed in the background to allow the UI to remain operational. Furthermore, a challenge that remains unresolved is the integration of the outputs from both programs into a single log file.

As described in section 6.3, some critical configuration errors were discovered during the testing process. In the pursuit of fixing the configuration errors, the WPA network was accidentally configured to use enterprise authentication in the form on 801.2X. Switching the configuration back to WPA-PSK, the Wi-Fi connection continued trying to reach the server for enterprise authentication during activation, timing out with the error message:

```
Error: Connection activation failed: 802.1X supplicant took too long to authenticate
Hint: use 'journalctl -xe NM_CONNECTION=7fb3feca-d2b4-424a-85ee-e0aa40f1c90a +
    NM_DEVICE=wlan1' to get more details.
```

After allocation of significant time and the discovery of more unexpected behavior with **`nmcli`**, it was decided to create a new OS setup on another SD card.

For the correct setup of the WEP network, the following lines were added or modified:

```
  sudo nmcli connection modify Wifi-WEP 802-11-wireless-security.group wep104, wep40
  sudo nmcli connection modify Wifi-WEP 802-11-wireless-security.auth-alg shared
  sudo nmcli connection modify Wifi-WEP 802-11-wireless-security.pmf 1
```

With the renewed OS setup, the connection timeout for the WPA supplicant seemed to be resolved, and WPA and WPA2 connections were correctly established.

However, on the second pentest of the WPA network, the supplicant error appeared once again, even though enterprise authentication was never activated, which leads to the assumption that this error is being caused by a software bug.

The final solution found involves the WPA supplicant being disabled by executing **`systemctl disable wpa_supplicant`**. This allows the activation of the WPA network, although the error still appears occasionally. A reliable fix for the problem could not be determined at this point.

5.4. Rotary Encoder

To use the encoder dial as a scroll wheel, its signals need to be processed and translated to instructions for the main program to execute. After the encoder was soldered to a simple circuit of pull-up resistors and connected to the GPIO pins of the RPI, the encoder program was developed for processing the encoder's outputs.

Off the shelf rotary encoders incorporate clicks into their rotation, similar to the scrolling wheel of a computer mouse, and output two digital signals. These signals are called DT and CLK, which, when turned, oscillate with 90 degrees of phase shift, resulting in a Gray code with two bits (see fig. 5.2). Through saving the previous state and comparing it to the new state, once the encoder was turned, the direction of rotation can be determined. According to various online source, this kind of encoders switch one state on each click, like the dotted arrow in fig. 5.2 depicts.

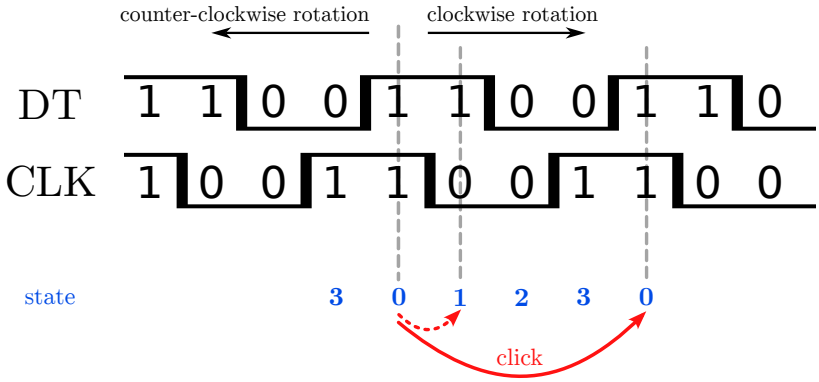


Figure 5.2.: Encoder Processing

During development of the encoder program, the output showed unexpected behavior, which was discovered to originate from the encoder switching through states 0 to 3 between clicks, as depicted by the red arrow in fig. 5.2. The discovery of this behavior helped to simplify the processing since this pattern allows a single query of the signal states to determine the direction of rotation. As seen in listing 4.3, the processing was then applied in a single ISR, comparing the DT and CLK signals.

5.5. ESP32

To establish communication between the RPI and the ESP32, some of the GPIO pins needed to be initialized for UART, which was achieved by adding a hardware overlay to the *config.txt* file. The code of the ESP32 in large part consists of code examples provided by the ESP-IDF GitHub ([Syst]). From these examples the code was compiled from snippets the Wi-Fi station example, MQTT web socket example and UART events example. These snippets were modified to work in conjunction with the scripts present on the RPI.

When closing a network and reopening another, instructing the ESP32 to switch to the new network, led it to regularly crash with the error **`itwt_stop_process`**. After investigation, the error seemed to originate from a Wi-Fi power-saving mechanism present in Wi-Fi 6, called Target Wake Time (TWT). Various attempts at solving the error by disabling the power saving were attempted but resulted unsuccessful.

The solution, finally implemented, involved the code being restructured to reset the MCU on a UART command, preceding the establishment of a network connection.

Chapter 6.

Pentesting

This chapter describes the pentesting of the Wi-Fi environments, established with the RCSL. It starts by explaining some common (Wi-Fi-) network attacks, of which the cracking of WEP and WPA/WPA2 are then tested on the RCSL and the results documented.

6.1. Attacks

Wi-Fi attacks can take many different forms, depending on the type of attack and which mechanism it tries to break. There are attacks aimed at breaking the Access Controls, Integrity Controls, Confidentiality or Availability. [Oriy 17] The following selection of attacks quite nicely depicts the different vectors an attacker can take to hack a Wi-Fi network and its STAs.

6.1.1. War Driving

War Driving describes the act of gathering information on Wi-Fi networks in larger numbers in order to locate specific or weak targets. This is done by traversing areas of interest with data collection equipment, which collects information about every network coming into range. After capturing and mapping as many Wi-Fi networks as possible, a more sophisticated attack can be launched on the captured networks, or the data can be sold on the black market. [Oriy 17]

6.1.2. Rogue Access Point and Evil Twin

When using Rogue Access Points, attackers place extra APs onto a network, trying to stay undetected and get STAs to connect to the malicious AP. This can, for example, be done by connecting an AP to an open and unprotected LAN port inside a public or company building. Once a STA has connected to the rogue AP, it acts as a MitM, allowing the attacker to manipulate or eavesdrop on the traffic.

Evil Twins have some similarity to Rogue Access Points in that they also pretend to be a legitimate part of an existing network. Different to Rogue APs, Evil Twins do not connect to the legitimate network, but instead spoof wireless networks, providing a stronger signal, to again get STAs to connect to it. Some attackers might also use a deauthentication flood to get STA disconnected from their genuine AP and auto connect to the malicious one.

Because they employ one-way authentication, WEP and OWE networks are vulnerable to these attacks. [Sank 21, page 105,119]

6.1.3. Cracking WEP

As touched on in chapter 3, WEP has several vulnerabilities, most notable the encryption algorithm, the one-sided authentication mechanism, and the lackluster MIC.

WEP encryption is based on the RC4 algorithm and works "by exclusive-ORing the data stream with a pseudo-random stream of bits generated based on the WEP key" [Sank 21, page 104] and a 24-bit initialization vector (IV). With 24 bits, there are about 16.5 million combinations for the IV, which can be exhausted in a few hours of network traffic. The probability of the same key being reused is at about 50% after around 5000 frames have been captured, opening the possibility for reverse engineering by analysis of the frames [Oriy 17].

The MIC with CRC-32 is not sufficient in ensuring message integrity, allowing manipulated frames, injected into the network, to be accepted as non-compromised. [Oriy 17]

Cracking a WEP network involves the reverse engineering of the password, aided by exploiting the authentication mechanism and the MIC. The process begins by scouting the target network, commonly conducted by capturing all traffic on all channels from networks in the vicinity. In normal operation, a NIC only receives the traffic addressed to its MAC address. To disable this filtering, the network card is put into monitor mode, allowing it to capture all packets transmitted in the vicinity, and discover all APs present. After the target network has been determined, its traffic is captured, and after enough frames have been transmitted, the password can be reverse engineered.

However, the process can be accelerated by injecting spoofed ARP frames to artificially increase traffic. This technique is called a replay attack and involves the capture of several authentic ARP frames, subsequently modifying and injecting them, which "the AP will re-broadcast [...] and as a result generate [new IVs]" [Oriy 17].

Sending ARP frames without association to the AP will lead to them being rejected, therefore another spoofed frame is injected to fake authentication and association with the AP.

6.1.4. Cracking WPA/WPA2

WPA and WPA2 networks come with improved encryption in the form of TKIP and SAE-CCMP, which cannot be cracked in the way, WEP can [Oriy 17]. To crack a WPA network, it is common to attack the authentication mechanism because, as described in chapter 3, it can be vulnerable to offline attacks.

An attacker starts off by capturing the handshake between an AP and STA during authentication, from which the PSK is extracted. In order to capture a handshake, the attacker would have to wait for a device to authenticate with the network, which depending on the number of STAs, could take quite some time. A deauthentication attack is performed, injecting frames, which force the STAs to reconnect to the AP and therefore making the capture of a handshake predictable.

Since the PSK is derived from the password and other information, which is openly available, the password can be brute forced by generating PSKs and comparing them to the captured one. The brute force attack can be accelerated by performing a dictionary attack. However if the network uses a strong password, which is not contained in the password dictionary, the attack will be unsuccessful in the way that it takes too long to brute force the passphrase. Furthermore, APs with PMF enabled can not be attacked with this method, since encrypted deauthentication frames cannot be spoofed.

6.2. Methods

The pentests were conducted using the Aircrack-ng suite with Kali Linux. Kali Linux is a Linux distribution, which comes with a vast collection of tools for pentesting, security auditing and computer forensics. The Aircrack-ng suite is tailored towards network security assessment and is included with Kali Linux.
Note: For more information, consult the Kali and Aircrack-ng documentation [Airc][Kali].

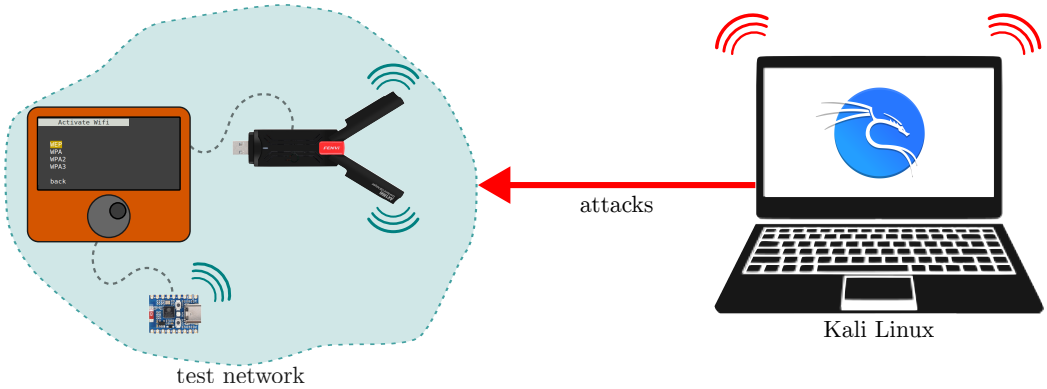


Figure 6.1.: illustration of the test setup

All tests were run on a Dell Vostro 3550 with an Intel Core i3 CPU from 2011 and 8GB of RAM, using the Alfa AWUS036AXM network card.

6.2.1. Cracking WEP

The RSCL is used to establish a WEP network to be cracked (see fig. 6.1). Starting the target scouting, **`airmon-ng start wlan1`** is used, which puts the card into monitor mode and creates a new network interface, called "wlan1mon". With the card being able to sniff all traffic in the area, the target network can be searched by using **`airodump-ng wlan1mon`**.

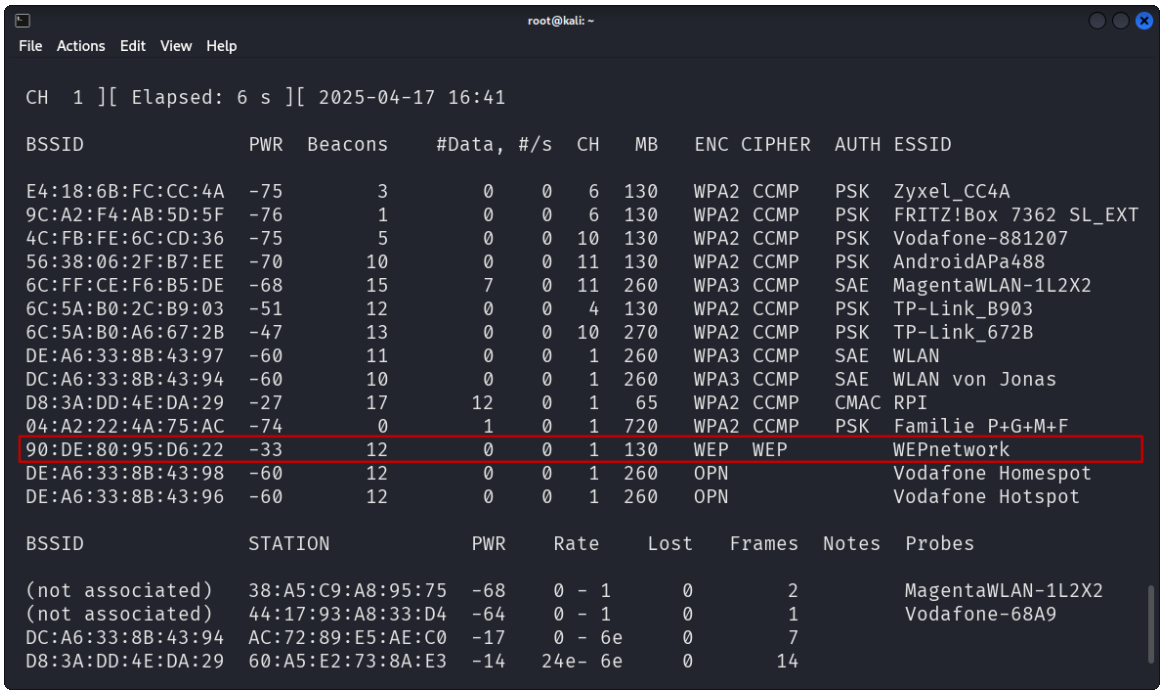


Figure 6.2.: Screenshot of airodump-ng

As seen in fig. 6.2, the program scans on all channels for networks within reach and lists them in the table on top. The target network, marked with the red border, is listed with its respective BSSID, RSSI, recorded Beacon Frames, encryption standard, and cipher.

In the second table, all STAs in reach are listed and to which AP they are associated. If a STA is actively scanning for networks, it is also listed with the network name it is probing for.

After the target network has been identified, airodump-ng is set up with filters to capture all traffic related to the AP and write it into an output file as seen in fig. 6.3.

`--channel` filters the captured frames to only collect frames on the APs channel
`--bssid` filters the frames related to the BSSID of the target

Figure 6.3.: Screenshot of airodump-ng filtered for the target network

`--write` specifies the output file
All devices associated with the network can be seen in the lower table, which at that time is only the ESP32 with the address **`40:4C:CA:5C:F9:CC`**.

For the replay attack, a fake association with the AP is established, using **`aireplay-ng`** as seen in fig. 6.4.
`--fakeauth 0` defines the authentication to be performed infinitely until a connection is established.
`-e` sets the name of the target network
`-a` sets the BSSID of the target network
`-h` sets the MAC of the device to be associated with the target network

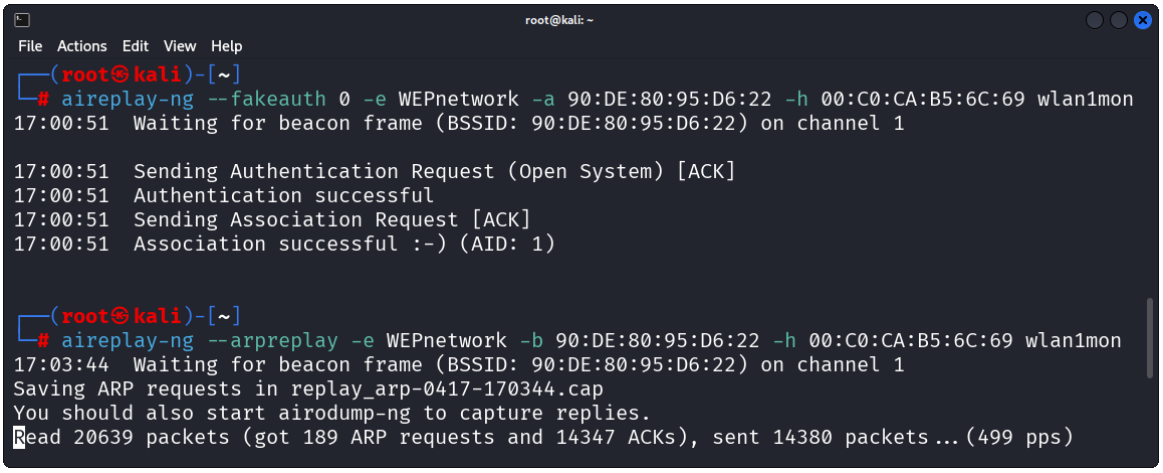


Figure 6.4.: Screenshot of aireplay-ng fake authentication and ARP replay attack

Upon successful association, as seen in fig. 6.4, ARP requests can be injected.
`--arpreplay` selects the replay attack for ARP frames

-b sets the BSSID of the target network
-h sets the MAC address of the attacker

Creating and capturing traffic, IVs are collected, and the cracking of the password can be started simultaneously. As seen in fig. 6.5, **aircrack-ng** is executed and reads the output of airodump-ng, attempting to crack the password for every 5000 IVs collected.

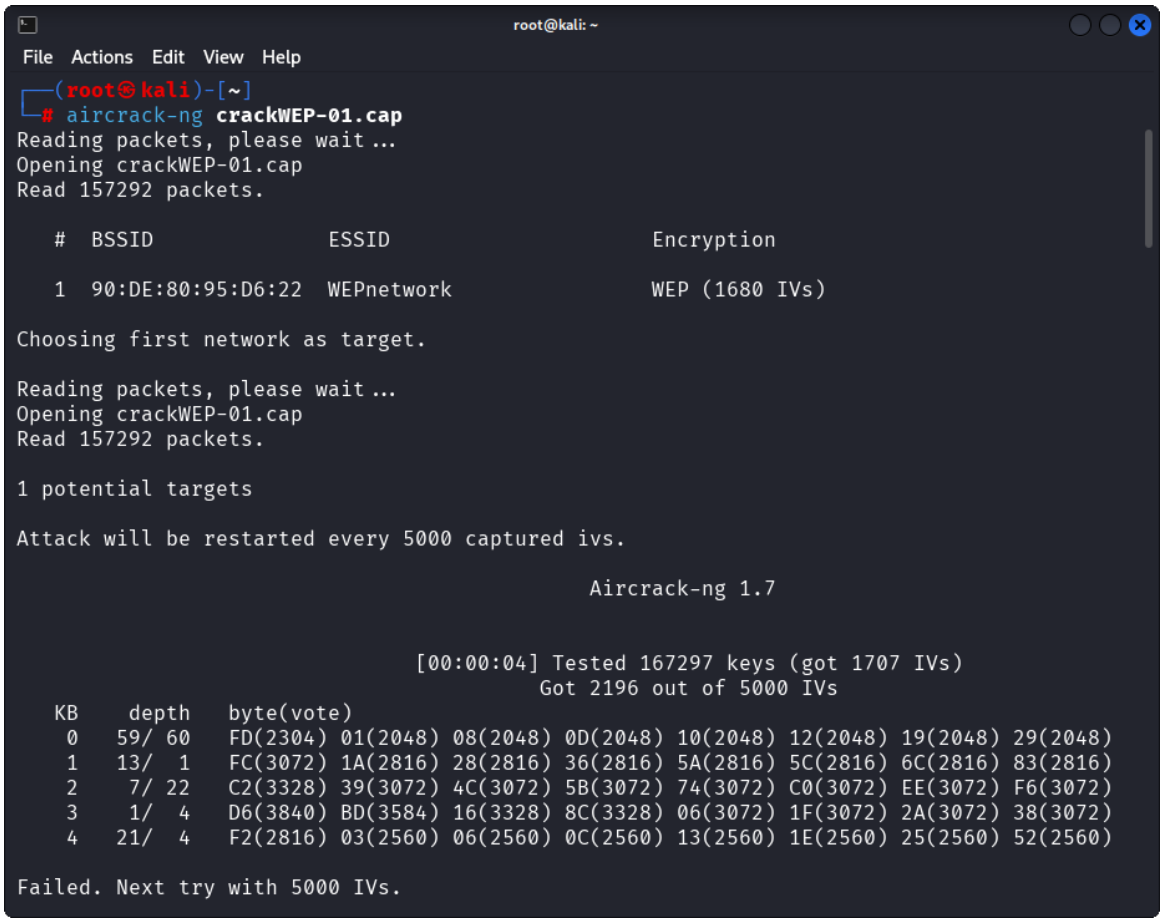


Figure 6.5.: Screenshot of the cracking process with aircrack-ng

6.2.2. Cracking WPA/WPA2

This attack starts off by scouting the target network with **airodump-ng** as well, in this case it is called "WPAnetwork".

Once the desired AP has been discovered, **airodump-ng** is used to capture the (re-)authentication handshake. To ensure the functionality of frame injection, it was tested with airodump, using the **--test** flag. The output, depicted in fig. A.4, means that normal frames are ignored, due to the STA not being authenticated. If PMF is not active, deauthentication frames can still be injected.

The deauthentication attack is performed with **aireplay-ng** to force the re-authentication of connected STAs (see fig. 6.6).

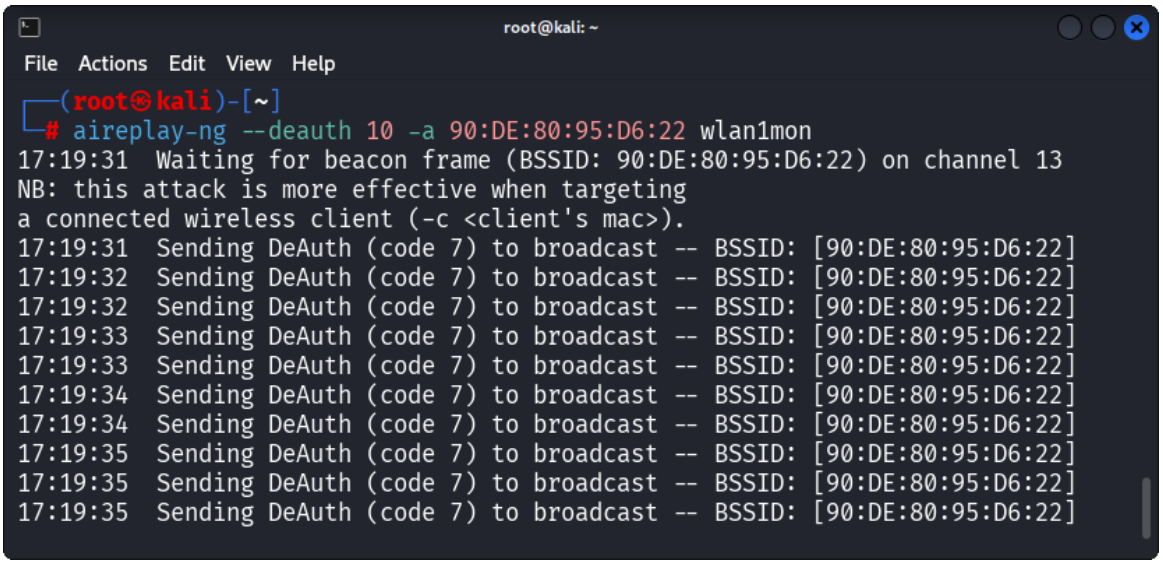


Figure 6.6.: Screenshot of the Deauthentication Attack

--deauth 10 promts the program to transmit 10 deauthentication frames
-a specifies the APs BSSID, leading to the frames being broadcast to all STAs
-c can be used to specify the MAC address of a STA, to send the frames directly to a specific STA, as recommended by the program

After capturing the authentication handshake, a dictionary attack is performed to crack the password, which in this case utilizes the "rockyou.txt" password list, built into Kali Linux, to brute force the password.

Aircrack-ng starts to iterate through the password list, computes the PSK for each password and compares it to the captured PSK. The brute forcing is started with:
aircrack-ng -b [BSSID] -w [passwordlist] WPAcrack-01.cap

-b specifies the AP to be targeted
-w specifies the password file, in this case the rockyou.txt, found in */usr/share/wordlists*

6.3. Results

Navigation of the UI seems intuitive, and switching between menus works fluidly. Execution of the various scripts is reliable, if implemented correctly. However, the ESP32 has been observed to crash occasionally during Wi-Fi connection establishment. The circumstances and cause for the occurrence of these sporadic crashes are unclear at the moment, and a fix is not foreseen for the scope of this thesis.

Upon conducting the tests, it was discovered that the configurations for the WEP, WPA and WPA2 networks were not correctly set up. The faulty configuration resulted in the network, supposedly encrypted with the WEP encryption standard, showing up as a WPA network in **airodump-ng**. Also, the WPA and WPA2 networks were identified by airodump as WPA3 networks, using SAE as the authentication mechanism. The section 5.3 details the process of resolving the configuration errors and allowing the following tests to be conducted.

The webapp functions, specifically the MQTT communication and Juice Shop server, seem to work without complications, but were not tested thoroughly.

6.3.1. Cracking WEP

The first attempt at cracking the WEP network was unsuccessful because only the ESP32 being connected to the network, ARP request injection seemed unable to generate significant traffic, therefore the collection of IVs proceeds very slowly.

As a workaround for test two, the RPI was connected to the internet and a second STA in form of a Smartphone was put on the network. The smartphone streamed a video, which generated sufficient traffic. This time, the password could be cracked after around three minutes and only 214 captured IVs.

For control, a third test was conducted, for which the password of the hotspot was changed with the new password function. Again, the password could be cracked in about two minutes, with around 15000 IVs collected.

An investigation into the first test revealed that the ARP injection did not function properly, therefore a fourth test without the video stream was conducted. With working ARP injection, the IV collection proceeded quicker than in test one, however it still took around 20 minutes of frame capture and six minutes of cracking, as depicted in fig. 6.7, to find the password.

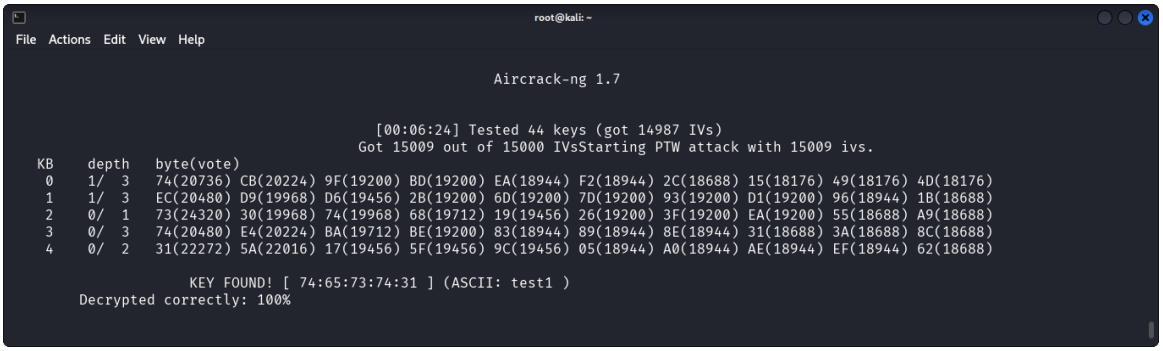


Figure 6.7.: Screenshot of the fourth cracking attempt

6.3.2. Cracking WPA/WPA2

On the first test, cracking the password was unsuccessful because the handshake was not captured, leading to aircrack-ng exiting with the output depicted in fig. 6.8.

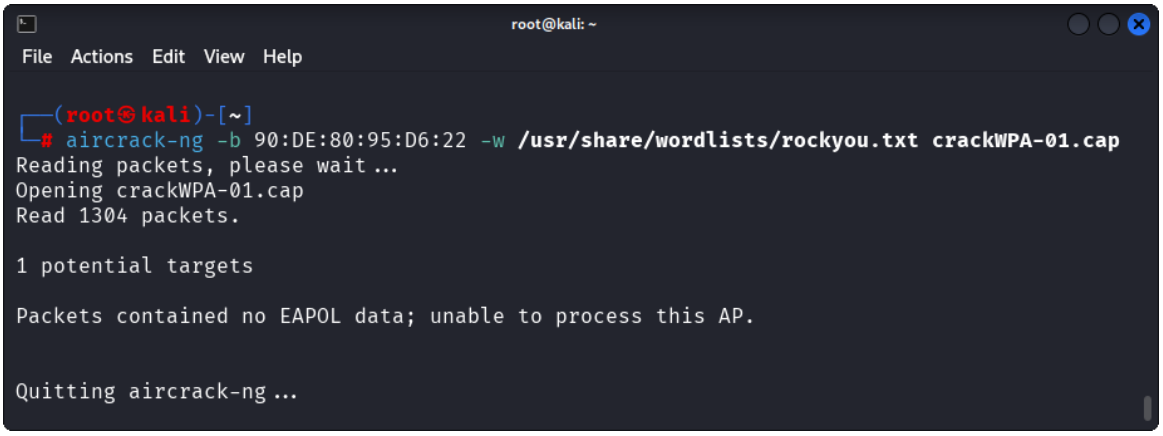


Figure 6.8.: Screenshot of the failed handshake capture

On the second try, a handshake was captured by manually reconnecting the EPS32 to the network. Capture of a handshake with the use of the deauthentication attack could not be achieved. With the password "TestSetup123", created on creation of the connection, the network could not be cracked with the RockYou password list. Since this password is not contained in the list, after two hours, the password could not be found, as to be seen in fig. A.5.

After setting a new random password for the third try, cracking the network, shown in fig. 6.9 was possible in 22 seconds. Multiple follow-up tries with other passwords also resulted in the successful cracking of the network, with the limitation of needing to manually (re-)connect the ESP32 to the network.

Cause for the failing deauthentication attacks is most likely PMF being active for the network. Even though the configuration was double-checked, the previous unexpected behavior of Network Manager leads to the suspicion that the configuration is not being applied correctly.

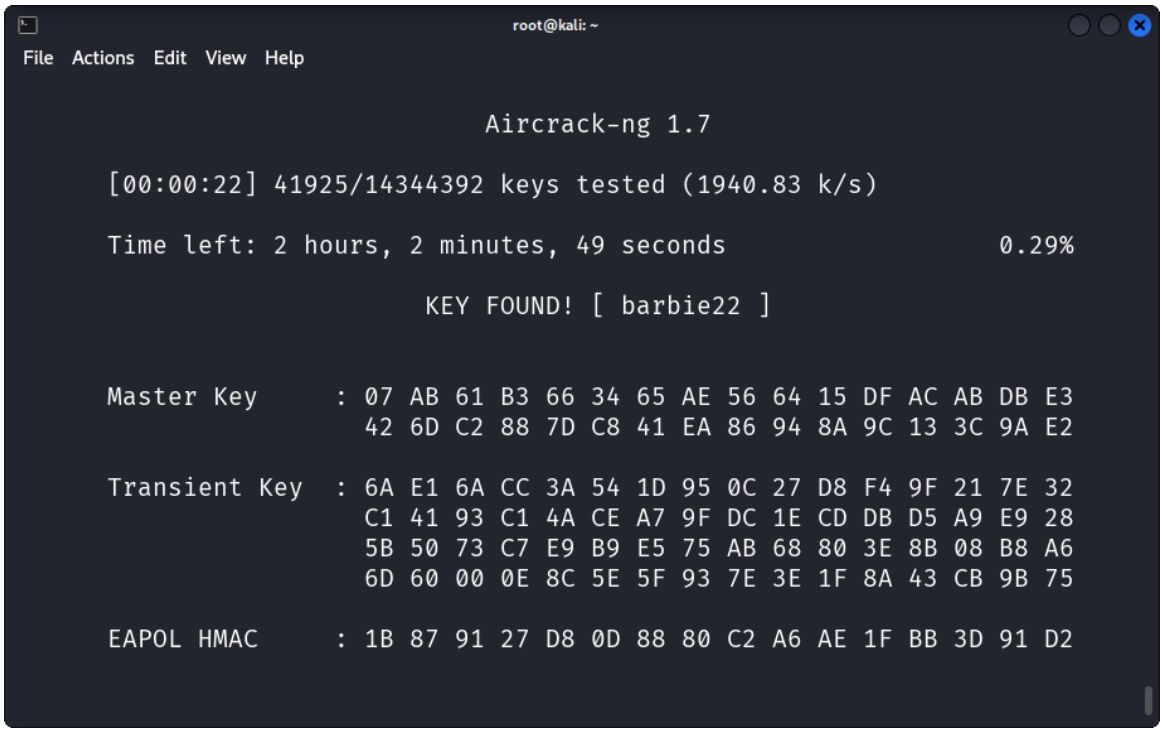


Figure 6.9.: Screenshot of the successful cracking of WPA

Chapter 7.

Cybersecurity Education

In this chapter, the importance of cybersecurity education is highlighted, and the potential use cases for the RCSL in education are layed out.

7.1. Importance of Cybersecurity Education

The advancement of digital information technology, accelerated by trends like Artificial Intelligence, Cloud Computing or the Internet of Things (IoT), has reached a high pace, which companies and institutions are trying to keep up with. In the pursuit of keeping up with this pace, convenience is often the main focus, whereas security and privacy are left as an afterthought. [Salm 17, chapter 1] This leads to the implementation and operation of vulnerable systems, presenting a larger attack surface for cybercriminals.

Experts estimate that cybercrime causes annual damages of 110 billion dollars worldwide and most companies, relying heavily on their IT infrastructure, can only operate for a few days without these systems [Hell 23, page 3],[Salm 17, page 12]. cyberattacks in the form of data piracy, or blackmailing have become commonplace and cause ever-increasing financial damages for companies, the public, and governments, as seen in the example depicted in fig. 7.1.

Cybersecurity awareness and education play a large role in ensuring the integrity of networks and systems. People without awareness and training in cybersecurity oftentimes tend to circumvent or ignore security mechanisms since following the principles of cybersecurity is often connected to effort [Hell 23]. Awareness training should educate about the potential threads in the cyberspace and security measures to apply for mitigating the risk of falling victim to cybercrime. Futhermore, the training should sensitize for common schemes of Social Engineering. This helps to protect against threads like Social Engineering, Malware, or network attacks, such as MitM or Evil Twin. [Pogu 13, page 24-25], [Mari 24]

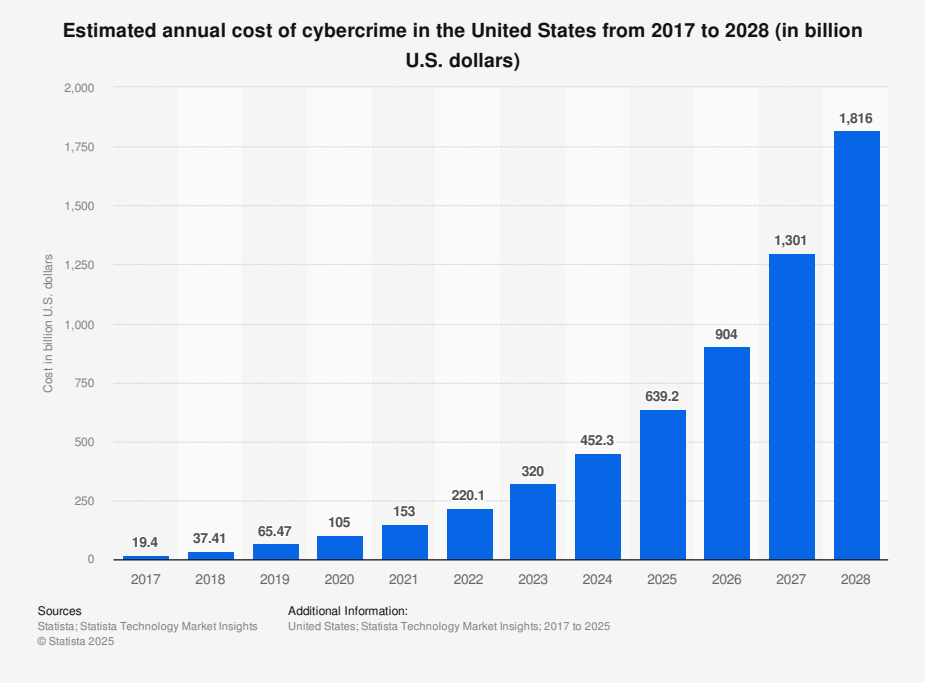


Figure 7.1.: Estimated annual cost of cybercrime in the United States from 2017 to 2028 [Stat 24]

7.2. Results Evaluation

The general usability of the RCSL proved to be in line with the goals set for the project with intuitive and smooth operation. The software architecture is mostly modular, allowing future expansion, and executes correctly implemented functions reliably. For improved development and problem analysis as well as performance, a rewrite of the code with adherence to software engineering best practices would be adviceable.

Concluding the tests, it appears the RCSL can provide a relatively reliable environment to perform network attacks in. However, the generation of artificial network traffic could aid the speed of cracking WEP, as well as depict a more realistic scenario, and might be considered for future improvements. The error caused by wpa_supplicant, described in section 5.3, seems to have been fixed, but should also be revisited in order to implement a more dependable fix.

7.3. Potential Use Cases

Confrontation with realistic scenarios and hands-on practice are effective tools for raising awareness and teaching cybersecurity. Authentic experiences help to "reinforc[e] the importance of [security]" [Mari 24] and connect previously learned theory to the experience

in real world application. Providing context for learning is essential, as it helps to build mental schemas and understanding of the core concepts, enhancing the ability to transfer the knowledge to different scenarios. [Cybe 23, page 61]

The RCSL is able to create test environments for various applications and could be used as tool in education for demonstration purposes and to allow students to get hands-on experience.

A lecturer could use the device to demonstrate the process of cracking a Wi-Fi network and performing subsequent attacks like MitM to eavesdrop on the traffic. The MQTT communication between the RPI and ESP32 could be a suitable target for such an attack. Demonstration of the ease with which an improperly secured network can be cracked and data stolen or manipulated would most likely give a vivid impression of the threats present in the cyberspace. This memorable experience intern might lead to the perception of the importance of cybersecurity being increased, aiding the spread of awareness.

For cybersecurity students, the RCSL could serve as a platform to practice pentesting techniques on, providing context for theoretical concepts and practical experience. The attacks performed in chapter 6 are an obvious choice as well as various network attacks, such as MitM, DoS, or TLS/SSL attacks. Besides pentesting, future expansions of the project could also include the implementation of protective measures, such as IDS systems, allowing students to gain experience in their application.

The Juice Shop could be employed for teaching security practices in software engineering and web development, providing context for the adherence to principles of secure development.

With the future addition of features, like the ones suggested in chapter 9, the RCSL could be expanded to cater for numerous further use cases.

Chapter 8.

Summary

Increasing damages caused by cybercrime in recent years have shown the need for improving cybersecurity for companies, governments, and citizens. The mismatch of demand and supply of trained security professionals represents a significant challenge in curbing the numbers of cybercrime predicted for the near future. As a result, there is an urgent need for training students in the complex and multi-faceted discipline of cybersecurity. Besides training professionals, raising awareness with employees and the public also plays an important role in protection against cybercrime.

The development of the Raspberry Pi CyberSec Lab aims to create a testing platform to aid in raising awareness and in education.

With the use of common components and a mostly modular software architecture, the RCSL lays the groundwork for a useful platform. Testing showed that the RCSL is capable of providing environments for practicing network security, although certain flaws were revealed. In summary, the goal of creating and testing a device, which combines usability with expandability, was mostly achieved.

Implementing the expansions and improvements proposed in chapter 9, the RCSL could become a useful asset in teaching cybersecurity.

Chapter 9.

Outlook

9.1. UI Program

The UI Program is using the navigate() function, explained in chapter 4 to periodically reprint the display and read the user input. To improve performance, the function could be made to only refresh the output on change, meaning an interrupt on user input could trigger the reprinting of the menu and selection.

A menu manager class could be added, which contains a menu stack and methods to manipulate it. Instead of implementing navigate() as a method of the menu class, it could be a method of the menu manager to display the menu currently on top of the menu stack.

The structure and therefore readability of the code could furthermore be improved by replacing the lambda functions and implementing the functions as methods with the command design pattern.

As described in section 5.2, the setup without a graphic server does limit the scaling of the UI for adjusting the resolution in the display configuration in config.txt. The UI could be reworked with a graphical interface to allow individual scaling of the menu items and the terminal output to improve the output's readability.

9.2. Input Processing

The encoder program could be merged into the UI program and run as a separate thread, which sends the preprocessed inputs to the parent thread. This change would allow the parallel input via the rotary encoder and a keyboard, which could be useful when a developer is connected to the RPI over SSH.

9.3. Features

There are several features that can be implemented in the future to greatly improve the devices versatility:

- Adding battery power would improve the portability of the device, but would most likely also require a computing unit, more efficient than the RPI.
- In the field of wireless communication, Bluetooth, NFC and 2.4Ghz RF communication environments could be set up for pentesting.
- Pentesting with SSL/TLS attacks and malware injection could be implemented as well as the addition of dummy files to test the cracking of different cryptographic techniques.
- The addition of defensive features like an Intrusion Detection and Prevention System would be a great way of teaching the application of cybersecurity principles.
- An environment for practicing incident response would be helpful in teaching security techniques and could allow students to be split up into groups of attackers and defenders, endorcing playful learning.

Appendix A.

Supplemental Information

A.1. User Manual

Anleitung zum Raspi-CyberSec-Lab

Jonas Schmitt

28. April 2025

1 Allgemeine Informationen

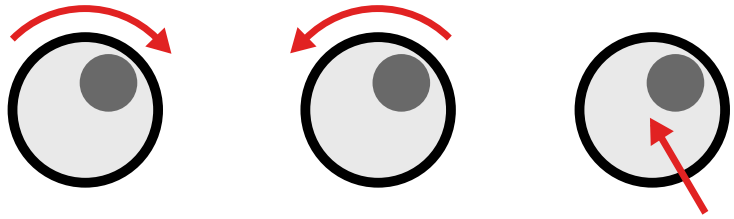
Das RaspberryPI Cybersecurity Lab (RCSL) ist eine Platform zum Testen verschiedener Cyberangriffe, das Projekt ist Teil meiner Bachelorarbeit in Elektro- und Informationstechnik an der Technischen Hochschule Nürnberg. Es hat verschiedene Funktionen um Umgebungen zu schaffen, welche für das Ausprobieren von Angriffen aus den Bereichen Wifi Netzwerke, Webseiten / -applikationen, MQTT Kommunikation und Bluetooth ausgelegt sind.

2 Benutzeranleitung

Zum Einschalten des Geräts, schließen Sie das Gerät mit dem zugehörigen Netzteil an den Strom an. Das Gerät fährt nun hoch, wobei der Bootvorgang zu sehen ist (das RaspberryPI Logo wird angezeigt). Nach erfolgreichem Bootvorgang wird das Hauptmenü angezeigt.

2.1 Navigation

Zur Navigation wird das Drehrad, wie in folgender Abbildung beschrieben, genutzt:



nach unten **nach oben** **bestätigen**

Die farbige Hinterlegung zeigt an, welcher Menüpunkt aktuell ausgewählt ist. Mit dem Drücken des Knopfes wird die Auswahl bestätigt und das Untermenü geöffnet, bzw die Aktion ausgeführt.
Zum verlassen eines Menüs wird die Option "back"gewählt.

2.2 Menü Übersicht

Dem Nutzer steht der folgende Menübaum zu Verfügung:

- **wifi** - Optionen zu Wifi Netzwerken
  - **activate** - Aktivieren eines Hotspots
    Auswahl der Netzwerkart
  - **deactivate** - Abschalten des aktiven Hotspots
  - **status** - Anzeigen der aktuellen Netzwerkeinstellungen
  - **change password** - Setzen eines neuen Passworts
    Auswahl der zu änderenden Netzwerkart
  - **monitoring** - Einschalten und Anzeigen des Netzwerkmonitoring
    on - Einschalten
    off - Ausschalten
    show log - Log anzeigen
    delete log - Log löschen
- **bluetooth** - Optionen zu Bluetooth
- **webapp** - Optionen zu Webapplikationen
  - **Juice Shop**
    on - Server Einschalten
    off - Server Ausschalten
  - **MQTT**
    on - MQTT Konversation starten
    off - MQTT Konversation beenden
- **development** - Anpassbare Optionen für Entwickler
- **power off** - Ausschalten des Geräts

3 Installation

Das Projekt wurde mit folgender Hardware umgesetzt:

- RaspberryPI 4B mit 8GB RAM
- Fenvi AX1800 USB Netzwerkkarte
- Waveshare ESP32c6 Microcontroller
- Waveshare 4.3 zoll LCD
- Drehgeber mit Druckknopf

Als Betriebssystem des RaspberryPI dient RaspberryPI OS in der light Version (nur Kommandozeile). Zum Kompilieren und Flashen des ESP32 Codes wird das ESP-IDF benötigt, dieses kann in Visual Studio Code als Erweiterung installiert werden.

Zuerst muss das Github Repository des Projekts auf den RaspberryPI geklont werden, dies kann mit folgendem Kommando erfolgen:

```
git clone https://github.com/Der-Erzfeind/Raspi-CyberSec-Lab-Project.git
```

Es wird empfohlen das Repository in das standardmäßige Nutzerverzeichnis zu klonen. Folgende Aktionen müssen im Anschluss ausgeführt werden:

- Kompilieren von main.cpp, encoder.cpp und mqtt.cpp
- Alle Skripte ausführbar machen
- Installation und Konfiguration von Mosquitto
- Installation des JuiceShops (und node.js)
- Einrichten des systemd services zum Ausführen von start.sh beim Start des Geräts
- Kompilieren und Flashen des ESP32 Codes

4 Entwicklung

Das RCSL öffnet nach dem Bootvorgang einen Hotspot, der für den Aufbau einer SSH Verbindung, wie folgend, genutzt werden kann:

1. Mit dem Hotspot verbinden,
   SSID: RPI
   Passwort: CyberSec
2. SSH Verbindung aufbauen mit **`ssh pi@10.40.0.1`**
   Passwort: admin

Im Nutzerverzeichnich befinden sich die Ordner mit den Daten zum Projekt und dem Juice Shop. In der Datei */Raspi-CyberSec-Lab-Project/Skripte/devMenu.sh* können die Entwicklungsfunktionen, welche im Menü "development"verfügbar sind, angepasst werden.

A.2. WiFi Basics

A.2.1. Architecture

IBSS networks don't have any APs, instead the STAs are connected directly to each other. WDS can be used when the reach of an AP needs to be extended, but there is no wired connection available. A secondary AP can be installed, which receives backhaul connection from the first AP. Mesh Networks are complex BSS networks with multiple APs, which the STAs can hop between. This type of network can improve the signal coverage and resilience and is often used in companies or public institutions. [Sank 21, page 8-9]

A.2.2. MAC Frames

In a BSS network the frame header is configured for the following functions:

- **Frame Control:** serves different control functions

  Protocol Version: describes which version of the MAC protocol this frame uses

  Type: describes the type of MAC frame is being used: there are data, management, control and acknowledgment frames

  Subtype: describes the subtype of the frame, e.g. association request or beacon frame

  From/to ds: indicates the direction of travel

  More Fragment: indicates more following fragments

  Retry: indicates that the current frame is a retransmission

  Power Management: gives information about the power state of the STA

  More Data: indicates more frames are to be transmitted

  Protected Frame: indicates whether a frame is encrypted

  Order: set to 1 if the order of processing is important

- **Duration:** indicates the time of the transmission, therefore how long the medium is occupied

- **Address 1:** contains the receiver address (RA), which can be the MAC address of the destination STA or the BSSID of the AP, depending on the direction, the frame travels

- **Address 2:** contains the transmitter address (TA), which can be the BSSID or the source address.
- **Address 3:** contains either the source or destination address
- **Sequence Control:**

  Fragment Number: ensures the correct processing sequence of the fragments

  Sequence Number: number of the frame

A.2.3. Wi-Fi Security

	WEP	WPA	WPA2	WPA3
Certification start	-	2003	2004	2018
Mandatory since	-	2006	2006	2019
Underlying standard(s)	802.11	802.11i(part. compliance)	802.11i, 802.11w(PMF)	802.11s(SAE), 802.11w
Information element	None	WPA IE	RSN IE	RSN IE
Encryption	RC4	TKIP	TKIP, AES-CCMP	AES-CCMP, AES-GCMP
Authentication	None	PSK, 802.1X	PSK, 802.1X	SAE, 802.1X
Usability enhancement	None	None	WPS	DPP
Open Wi-Fi security	None	None	None	OWE

Table A.1.: Evolution of Wi-Fi Security [Sank 21, page 123]

A.3. Documentation

A.3.1. Hardware

Figure A.1.: RCSL front

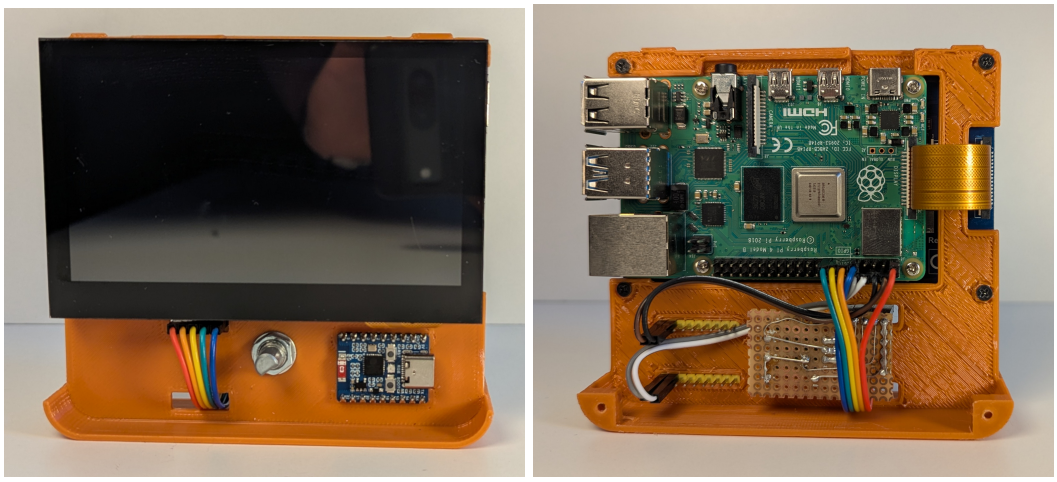


Figure A.2.: RCSL without case

A.3.2. Software

Menus Overview

- **wifi:** options for wifi
  - **activate:** open up a hotspot - runs wifiActivate.sh

    choose the wifi standard
  - **deactivate:** turn off the active network - runs wifiReset.sh
  - **status:** display the current networks settings - runs wifiStatus.sh
  - **change password:** set new password for a network - runs wifiNewPassword.sh

    choose the wifi standard
  - **monitoring:** settings for wifi monitor - runs wifiMonitor.sh

    on/off/show log/delete log
- **webapp:** options for webapps
  - **Juice Shop:** settings for the Juice Shop - runs juiceShop.sh

    on/off
  - **MQTT:** settings for MQTT communication - runs mqtt.sh and the mqtt program

    on/off
- **development:** runs development options from development.sh
- **power off:** power off the system - runs shutdown.sh

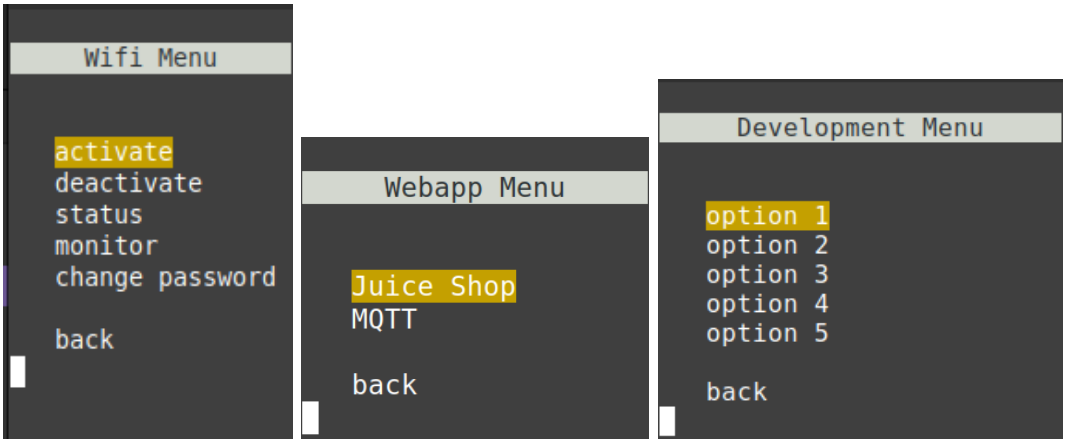


Figure A.3.: Device menus

A.4. Pentesting

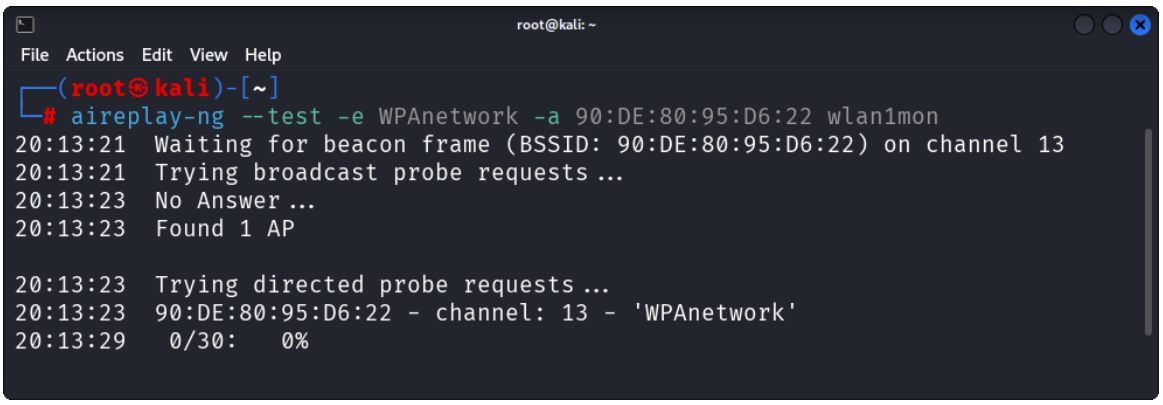


Figure A.4.: Screenshot of testing frame injection

Figure A.5.: Screenshot of the failed attempt to crack WPA

List of Figures

List of Tables

List of Listings

Bibliography

[Airc] Aircrack-ng. "Aircrack-ng Documentation". https://www.aircrack-ng.org/doku.php. Accessed: 2025-04-14.

[Cybe 23] *Cybersecurity Teaching in Higher Education*. Springer Cham, 2023.

[Geeka] GeeksforGeeks. "IEEE 802.11 mac Frame". https://www.geeksforgeeks.org/ieee-802-11-mac-frame/. Accessed 2025-01-15.

[Geekb] GeeksforGeeks. "Intrusion Detection System (IDS)". https://www.geeksforgeeks.org/intrusion-detection-system-ids/. Accessed: 2025-03-21.

[Hark 08] D. Harkins. "Simultaneous Authentication of Equals: A Secure, Password-Based Key Exchange for Mesh Networks". In: *2008 Second International Conference on Sensor Technologies and Applications (sensorcomm 2008)*, pp. 839–844, 2008.

[Hell 23] R. Hellmann. *It-sicherheit: Methoden und Schutzmaßnahmen Für Sichere Cybersysteme*. De Gruyter Oldenbourg, 2023.

[Kali] Kali Linux. "Kali Linux Documentation". https://www.kali.org/docs/. Accessed: 2025-04-14.

[Kimm 25] B. Kimminich. *Pwning OWASP Juice Shop*. Leanpub, 2025.

[Mari 24] H. S. Mariano and D. C. Café. "Measuring Public Wi-Fi Security Awareness via Captive Portal Connections Using a Microcontroller". In: *9th Workshop on Communication Networks and Power Systems (WCNPS 2024)*, University of Brasília (UnB), Brasília, Brazil, 2024.

[Morr] MorrowNR. "USB WiFi Adapters that are Supported with Linux In-Kernel Drivers". https://github.com/morrownr/USB-WiFi/blob/main/home/USB_WiFi_Adapters_that_are_supported_with_Linux_in-kernel_drivers.md#axe3000---usb30---24-ghz-5-ghz-and-6-ghz-wifi-6e. Accessed: 2025-03-20.

[Oriy 17] S.-P. Oriyano. *Kali Linux Wireless Penetration Testing Cookbook: Identify and assess vulnerabilities present in your wireless network, Wi-Fi, and bluetooth enabled devices to improve your wireless security*. Packt Publishing, 2017.

[Pogu 13] W. Poguntke. *Basiswissen IT-Sicherheit*. W3L-Verlag, 2013.

[Salm 17] A. Salmon, W. Levesque, and M. McLafferty. *Applied Network Security: Master the art of detecting and averting advanced network security attacks and Techniques*. Packt Publishing, 2017.

[Sank 21] S. G. Sankaran and S. R. Gulasekaran. *Wi-Fi 6: Protocol and Network*. Artech House, 2021.

[Stat 24] Statista. "Estimated annual cost of cybercrime in the United States from 2017 to 2028". https://www.statista.com/forecasts/1399040/us-cybercrime-cost-annual, 2024. Accessed: 2025-05-05.

[Syst] E. Systems. "ESP-IDF". https://github.com/espressif/esp-idf/tree/master. Accessed 2025-03-30.

[Watj 18] D. Wätjen. *Kryptographie - Grundlagen, Algorithmen, Protokolle*. Vol. 3. Auflage, Springer Vieweg, 2018.

[Wiki] Wikipedia. "Wi-Fi". https://en.wikipedia.org/wiki/Wi-Fi. Accessed 2025-01-13.