UNIVERSITY OF BARI ALDO MORO

COMPUTER SCIENCE DEPARTMENT

MASTER'S DEGREE COURSE IN COMPUTER SCIENCE

SECURITY ENGINEERING CURRICULUM

IOT SECURITY EXAM

CASE STUDY

WARDRIVING TOOL-WIFI SNIFFER AND CRACKER

Project Members:
Alessandro Aldo Boffolo | matr. 841260 | `a.boffolo@studenti.uniba.it`
Narcis Paviliuc | matr. 838039 | `n.paviliuc@studenti.uniba.it`
Simone Papa | matr. 802282 | `s.papa6@studenti.uniba.it`
Licia Congedo | matr. 828064 | `l.congedo4@studenti.uniba.it`
Cristiana Sorrenti | matr. 788565 | `c.sorrenti3@studenti.uniba.it`
Gabriele Leone | matr. 807379 | `g.leone94@studenti.uniba.it`

ACADEMIC YEAR 2024/2025

Contents

Chapter 1

Introduction

1.1 Context

In recent years, the Internet of Things (IoT) has revolutionized the way we interact with the digital world, bringing connectivity and intelligence to a wide range of devices from smart TVs to home appliances, from wearables to industrial sensors. This connected ecosystem has improved efficiency, automation and accessibility across many sectors, from home automation to healthcare and Industry 4.0.

However, the expansion of the IoT has also brought with it significant cybersecurity challenges, particularly in relation to wireless communications. Most IoT devices connect to the internet via Wi-Fi networks, making them potentially vulnerable to attacks based on traffic interception and manipulation. In fact, Wi-Fi networks are a critical point because:

- They transmit data into the airwaves, which can be easily intercepted by a nearby attacker;
- They often use weak or poorly configured authentication protocols;
- They are used by devices that rarely receive security updates.

In this scenario, sniffing and cracking attacks become particularly relevant. Sniffing involves passively intercepting wireless traffic, while cracking aims to break network security keys (WEP, WPA, WPA2), allowing the attacker to access traffic or infiltrate the network itself.

Understanding these techniques is not only useful for offensive purposes in the context of penetration testing, but is also essential for designing effective defence systems and improving security awareness of wireless networks in the IoT ecosystem.

1.2 Motivation

The main motivation for this study is the increasing importance of wireless network security, especially in the context of the IoT, where data transmission is almost always over Wi-Fi. Unlike wired networks, wireless networks inevitably expose transmitted data to the surrounding environment, allowing interception by attackers even in the absence of physical access to the infrastructure.

This case study was designed to understand and demonstrate in practice the real vulnerabilities of Wi-Fi networks, which are typically downplayed by both home users and network administrators. Using tools such as Airodump-ng, Aireplay-ng and Aircrack-ng, the intention is to investigate the most common sniffing and cracking practices of an attacker, with the aim of demonstrating how easy it is - in certain situations - to break the security of a network.

The choice of this topic is also motivated by the need for future security professionals to master offensive techniques in order to improve defensive strategies. Only by understanding the methods and tools used by attackers can we design more resilient systems and develop truly effective countermeasures.

1.3 Objectives

The main objective of this case study is to analyse the vulnerabilities of Wi-Fi networks and illustrate, in a controlled environment, the main attack techniques used to compromise their security. In particular, it aims to:

- Study the operation of the main wireless network security protocols (WEP, WPA, WPA2).
- Simulate real-world attack scenarios using traffic sniffing and network key cracking.
- Use tools from the Aircrack-ng suite (such as Airmon-ng, Airodump-ng, Aireplay-ng, and Aircrack-ng) to perform attack steps automatically and repeatedly.
- Demonstrate how an attacker can gain unauthorised access to the network, potentially compromising connected IoT devices.
- Reflect on the security implications and propose concrete countermeasures to mitigate these risks.

The work also aims to develop an automated tool that encapsulates all phases of the attack, facilitating hands-on cybersecurity demonstrations and training.

Chapter 2

Theoretical Background

2.1 Wi-Fi network

The term Wi-Fi refers to a set of wireless communication technologies, based on the IEEE 802.11 standard, designed to connect devices in a wireless local area network. The IEEE 802.11 standard, developed since 1997, defines specifications for physical layer communication (PHY) and medium access control (MAC) in wireless local area networks (WLANs).

A Wi-Fi network typically consists of one or more access points (APs) that act as a bridge between wireless devices (clients) and a wired network. Client devices (laptops, smartphones, IoT devices, etc.) connect to the AP via radio waves operating in the 2.4 GHz or 5 GHz frequency bands, and more recently 6 GHz (802.11ax standard - Wi-Fi 6E).

2.1.1 Architecture and operation

The 802.11 standard has two modes of operation:

- Infrastructure mode: the client device communicates with an AP, which handles network traffic.
- Ad hoc mode: devices communicate directly without an AP.

There are three stages to the connection process:

1. Scanning: the device searches for available networks using beacon frames sent by APs.
2. Authentication: an authentication mechanism is established (open, WEP, WPA, WPA2, WPA3).
3. Association: the client connects to the network and starts transmitting data.

Version	Year	Maximum theoretical speed	Frequency	Commercial name
802.11b	1999	11 Mbps	2.4 GHz	Wi-Fi 1
802.11a	1999	54 Mbps	5 GHz	Wi-Fi 2
802.11g	2003	54 Mbps	2.4 GHz	Wi-Fi 3
802.11n	2009	600 Mbps	2.4 / 5 GHz	Wi-Fi 4
802.11ac	2013	until 1.3 Gbps	5 GHz	Wi-Fi 5
802.11ax	2019	beyond 10 Gbps	2.4 / 5 / 6 GHz	Wi-Fi 6 / 6E

TABLE 2.1: Evolution of standard IEEE 802.11

2.1.2 Security in 802.11 networks

Wi-Fi networks support different security protocols to ensure authentication and data encryption:

- WEP (insecure)
- WPA / WPA2 (the most common standard today)
- WPA3 (new standard with protection against dictionary attacks and forward secrecy)

Despite the evolution of standards, several networks still use weak protocols or misconfigurations, making them vulnerable to attacks such as sniffing and cracking, the central themes of this case study.

2.2 Wi-Fi Cryptography

Over the years, the security of Wi-Fi networks has evolved significantly, moving from weak and vulnerable protocols to more robust modern solutions. Encryption protocols, initially designed to protect the confidentiality and integrity of wireless communications, have been updated to respond to the increasing threats and vulnerabilities that have emerged over time.

2.2.1 WEP

The first protocol used was WEP (Wired Equivalent Privacy), introduced in 1999 with the IEEE 802.11 standard. Based on the RC4 algorithm and static keys (40 or 104 bits), it combined a secret key with a 24-bit initialisation vector (IV). However, the shortness of the IV and weak key management made WEP highly vulnerable to sniffing techniques and replay or brute force attacks. As a result, it was soon considered outdated and insecure.

2.2.2 WPA

As a result of WEP's weaknesses, WPA (Wi-Fi Protected Access) was introduced in 2003 as an interim solution until a new, definitive standard was available. WPA used the Temporal Key Integrity Protocol (TKIP), which improved key management by dynamically regenerating keys with each packet. Despite the increase in security over WEP, WPA was also vulnerable, especially when weak passwords were used, exposing the network to possible dictionary attacks.

2.2.3 WPA2

The real breakthrough came with WPA2, ratified in 2004, which introduced the use of the Advanced Encryption Standard (AES) algorithm and Counter Mode with Cipher Block Chaining Message Authentication Code Protocol (CCMP). This combination provided a high level of security for wireless communications for several years. However, WPA2 also had vulnerabilities, such as the KRACK (Key Reinstallation Attack), which exploited weaknesses in the four-way handshake, especially on older devices.

2.2.4 WPA3

In 2018, WPA3, the current recommended standard, was developed to address these issues and the need for increased security in modern networks. This protocol introduces Simultaneous Authentication of Equals (SAE)-based authentication, which makes offline dictionary attacks impossible, and also guarantees forward secrecy, the protection of previous sessions in the event of a future key compromise. In addition, WPA3 enhances protection for IoT devices with a simplified configuration system known as the Device Provisioning Protocol (DPP). Although not yet fully supported by all devices, WPA3 is the standard that wireless networks are beginning to migrate to.

2.3 Sniffing and Cracking

In the context of wireless network security, sniffing and cracking are two phases that are essential to Wi-Fi traffic analysis and compromise.

2.3.1 Sniffing

Sniffing is the method by which an attacker captures packets transmitted on a wireless network. This is done by using a network interface configured in monitor mode, which allows him to capture all traffic passing through the air, regardless of destination. During this phase, useful information such as MAC addresses of clients and access points, network identifiers (SSIDs), channels used, type of encryption and authentication packets (such as handshakes in WPA/WPA2) are collected. The most commonly used sniffing tools are airodump-ng, Wireshark and tcpdump.

2.3.2 Cracking

Cracking, on the other hand, is when the attacker attempts to gain access to the target network using the information gathered in the sniffing phase. Assuming that networks have been protected with outdated protocols such as WEP, cracking can be achieved very quickly using statistically based IV attacks. For WPA/WPA2 networks, however, cracking focuses primarily on the four-way handshake, which can be captured by inducing a client to disconnect from the access point (via a deauthentication attack) and then forcing a new connection. Once the handshake is obtained, dictionary or brute force attacks are used to find the access password.

2.4 Main tools

The Aircrack-ng toolkit, a widely used open source wireless penetration testing suite, was used to sniff and crack Wi-Fi networks. This suite is a set of tools, each with a well-defined purpose, that work together to analyse, eavesdrop on and potentially compromise WEP- or WPA/WPA2-protected networks.

- **Airmon-ng**. This is the software used to enable the monitor mode on the wireless network interface. This mode is essential for sniffing, as it allows the wireless card to capture all packets in the area, even those not destined for the device itself. In addition, airmon-ng allows you to control processes that might interfere with the smooth operation of monitor mode.
- **Airodump-ng**. Once monitor mode is enabled, airodump-ng allows you to scan the wireless environment. It can identify nearby access points, connected hosts, the type of encryption used, the channel and other key information. It can also be used to capture specific packets, such as WPA/WPA2 handshakes, and store them in .cap files for later analysis.
- **Aireplay-ng**. This tool allows packet injection and the execution of active attacks. One of the most common is the deauthentication attack, which forces a device to temporarily disconnect from the network. This forces the client to reconnect, allowing airodump-ng to capture the four-way handshake necessary to continue cracking a WPA/WPA2 network.
- **Aircrack-ng**. This is the final tool used for the actual attack. Using the captured packets (typically a .cap file containing the handshake), aircrack-ng attempts to recover the network access key using a dictionary attack. Depending on the quality of the handshake and the strength of the password, the success of the cracking can vary greatly.

The coordinated use of these tools provides a complete workflow for compromising a wireless network and is essential for auditing and evaluating the security of Wi-Fi networks in compliance with current regulations.

2.5 Legal and Ethical Aspects

The analysis and experimentation of sniffing and cracking techniques in wireless networks raises serious legal and moral concerns. Indeed, it is important to note that even if such activities can be carried out for educational or auditing purposes, their use on unauthorised networks is a criminal offence in the vast majority of countries.

For example, the Italian Penal Code (articles 615-ter and 640-quinquies) and the European GDPR regulation prohibit abusive access to computer systems and the violation of the confidentiality of transmitted data. Passive interception of packets from another's network may also constitute a breach of privacy, unless with the express consent of the network owner or in the context of regulated activities (e.g., penetration testing under mandate).

From an ethical perspective, the use of attack techniques must always be guided by principles of responsibility and constructive purpose. The primary objective is not to compromise systems, but rather to understand their vulnerabilities in order to strengthen their security. For this reason, all testing activities must be carried out on laboratory networks, simulators or specially prepared infrastructures, with the explicit consent of the owners.

In the academic environment, these practices are an important tool available for the training of future security specialists, but it is essential that each time they are misused, they are accompanied by a critical reflection on their legal, moral and social implications.

Chapter 3

Methods

3.1 Case study phases

3.1.1 Enabling Monitor Mode

To begin the wireless traffic sniffing activity, the network interface must be configured in **monitor mode**. This mode allows the Wi-Fi card to intercept all packets in the vicinity, including those not directed to the device itself. It is essential for capturing handshakes and analyzing traffic.

The monitor mode was enabled using the following command:

```
airmon-ng start wlan0
```

After executing the command, the interface `wlan0` is converted to a monitor mode interface, usually named `wlan0mon`. The activation can be verified using:

```
iwconfig
```

Note: It is good practice to stop any processes that may interfere with monitor mode (such as `NetworkManager` or `wpa_supplicant`). This can be done automatically with:

```
airmon-ng check kill
```

Once the interface is in monitor mode, it is possible to proceed with the sniffing phase using tools like `airodump-ng`.

3.1.2 Network Scanning

Once the interface is set to monitor mode, it is possible to scan the surrounding wireless networks using `airodump-ng`. This tool enables the detection of access points (APs), connected clients, encryption types, MAC addresses, and channels.

The basic command used is:

```
airodump-ng wlan0mon
```

This command launches a real-time scan using the `wlan0mon` interface (previously activated by `airmon-ng`). The output consists of two tables: the upper table lists all discovered APs, while the lower one displays associated client devices.

To narrow the scan to a specific channel and save the captured packets (for later use in WPA/WPA2 cracking), the following command is used:

```
airodump-ng -c <channel> --bssid <AP_MAC> -w capture wlan0mon
```

Where:

- `-c` specifies the channel of the target access point,
- `-bssid` defines the MAC address of the target AP,
- `-w` sets the prefix of the output file (e.g., `capture.cap`).

This targeted scanning allows for efficient capture of WPA handshakes and minimizes interference from irrelevant traffic.

3.1.3 Handshake Capture

Capturing the WPA/WPA2 handshake is a critical step in the process of cracking a wireless password. A handshake occurs when a client (station) connects or reconnects to an access point (AP). This process can be passively observed or actively triggered by deauthentication.

Once airodump-ng is running with the proper parameters (as shown in Section 4.2), the handshake can be captured when a client connects to the network. To speed up the process, a deauthentication attack is often used to force a client to disconnect and reconnect.

The deauthentication is performed with the following command:

```
aireplay-ng --deauth 10 -a <AP_MAC> -c <Client_MAC> wlan0mon
```

Where:

- `-deauth 3` sends 3 deauthentication frames (if you put 0, it sends deauthentication frames until the handshake is captured),
- `-a` specifies the MAC address of the access point,
- `-c` specifies the MAC address of the client,
- `wlan0mon` is the monitor interface.

If successful, the client will automatically reconnect, and the handshake will be captured by `airodump-ng`. A confirmation message such as `"WPA handshake: <AP_MAC>"` will appear in the top right corner of the terminal.

The handshake file (e.g., `capture.cap`) can later be used for offline password cracking.

3.1.4 Capturing Device Location via ADB

During the handshake capture process, an additional step was performed to geolocate the device performing the attack. This was achieved by retrieving the GPS location of an Android smartphone connected via USB and controlled using the Android Debug Bridge (ADB).

The purpose of this step was to correlate the physical location of the attacker with the Wi-Fi network being targeted. This could be useful in penetration testing scenarios where the geographical distribution of vulnerabilities is analyzed, or in forensic investigations to log attacker movements.

Procedure

- An Android smartphone was connected to the attacker's machine via USB with developer mode and USB debugging enabled.
- The following ADB command was used to obtain the current GPS coordinates:

  ```
  adb shell dumpsys location | grep "mLastLocation"
  ```

- The coordinates (latitude and longitude) were extracted from the output and stored for documentation purposes, together with the timestamp of the handshake capture.

Advantages

- Provides context-aware information about where the attack was performed.
- Adds a layer of traceability to ethical hacking operations or research experiments.
- Allows the mapping of vulnerable access points in physical space.

Integrating location data with technical attack steps enhances the comprehensiveness of the case study and demonstrates how multiple data sources can be correlated for advanced analysis.

3.1.5 Password Cracking

Once the WPA/WPA2 handshake has been successfully captured, it is possible to attempt to crack the password using a dictionary-based attack. The tool used for this purpose is `hashcat`, which analyzes the captured handshake file against a wordlist of potential passwords.

The basic command is:

```
sudo hashcat -m 22000 <filehandshake.hc22000> <wordlist.txt>
```

Where:

- `-m` specifies the format file,
- `filehandshake.hc22000` is the file containing the captured handshake.

The tool will iterate through the wordlist and attempt to match each password candidate with the handshake data. If the correct password is found, it will be displayed in the output.

Example:

```
KEY FOUND! [ password123 ]
```

It is important to note that the success of this attack depends entirely on the quality of the wordlist and the strength of the original password. For more robust passwords, a brute-force attack may be infeasible due to time and computational constraints.

`rockyou.txt` is a commonly used wordlist and is often included in Kali Linux by default. It can be found at:

```
/usr/share/wordlists/rockyou.txt.gz
```

Before use, the file must be extracted using:

```
gunzip rockyou.txt.gz
```

3.2 Graphical Interface

The application presents a user-friendly graphical interface structured around a tabbed notebook, enhancing workflow organization for various Wi-Fi auditing tasks. The first tab, **"Scan & Capture,"** is dedicated to network discovery and data acquisition. Within this tab, users can select a wireless network interface from a dropdown list, and subsequently start or stop monitor mode on that interface. Controls are provided to initiate network scans, which populate a table displaying discovered access points. This table details each network's SSID, BSSID, signal strength (Power), Channel, Encryption type, Client status (indicating if connected clients have been checked for on the AP), and Handshake status (showing if a WPA or WPA2 handshake has been captured or if a capture attempt is ongoing). Users can select networks from this table to perform client checks or to initiate handshake captures. All relevant data from scans and handshake capture attempts, along with any available geolocation data obtained via ADB, is automatically saved or updated in the application's SQLite database.

The second tab, **"Crack Handshake,"** focuses on password recovery from the captured handshake files. Users can specify a wordlist file (typically a `.txt` file) which will be used by `hashcat` for the cracking attempts. A list within this tab displays all `.cap` handshake files found in the designated `handshakes` directory. For each file, the list shows its filename, the inferred SSID and BSSID of the target network, and the current cracking status (e.g., "Ready," "Cracking...," "FOUND: [password]," "Failed," or "Wordlist Exhausted"). Users can select one or more handshake files from this list to initiate the cracking process. Controls are also available to stop any ongoing `hashcat` cracking processes for selected handshakes. A notable feature in this tab is the **"Open Site"** button, which is configured to open a predefined external website in the system's default web browser. Below the tabbed notebook, a scrolled text area displays real-time logs of all operations, warnings, errors, and success messages, providing transparent feedback to the user. A status bar at the bottom of the window offers concise updates on the current state of the application.

3.3 Database

All data collected during scans and cracking attempts is stored in a local SQLite database named `wifi_data.db`. The primary table in this database is called `networks`, which is structured to store comprehensive information about each detected WiFi access point.

3.3.1 Database Structure

Its schema is as follows:

- `bssid` (`TEXT PRIMARY KEY`): The MAC address of the access point. Serves as the unique identifier for each network.
- `ssid` (`TEXT`): The network name (SSID). May be `NULL` for hidden or unidentified networks.
- `power` (`INTEGER`): Signal strength, typically in dBm (negative values; closer to 0 indicates stronger signal).
- `channel` (`TEXT`): The operating radio channel (e.g., “1”, “6”, “11”).
- `encryption` (`TEXT`): The encryption type used by the network (e.g., “WPA2 PSK”, “WPA3 SAE”, “OPN”).
- `latitude`, `longitude` (`REAL`): Geographic coordinates where the network was detected, if available.
- `last_seen` (`TEXT`): Timestamp of the last update to this entry in ISO 8601 format.
- `handshake_captured` (`INTEGER DEFAULT 0`): A flag indicating whether a handshake was captured (1 = yes, 0 = no).
- `password` (`TEXT DEFAULT NULL`): The cracked password, if available.
- `handshake_filepath` (`TEXT DEFAULT NULL`): The absolute path to the `.cap` file containing the captured handshake.

- `band` (`TEXT DEFAULT NULL`): The frequency band used by the access point, it could be 2.4GHz or 5GHz.

To avoid duplicate entries, the database uses an `ON CONFLICT DO UPDATE` clause on the `bssid` primary key. When a duplicate is encountered, existing values are only updated if new information is more complete (e.g., non-null SSID, stronger signal, new password). The `last_seen` timestamp is always refreshed to ensure temporal accuracy.

This database structure enables efficient tracking of WiFi networks, handshake capture status, cracking results, and geolocation. It serves as the core data source for the application's map visualization and analysis modules.

3.3.2 Database Choice: Why SQLite?

SQLite was selected as the database engine for CrackMate due to its lightweight nature, ease of integration, and zero-configuration design. Unlike traditional client-server database systems, SQLite operates entirely as a local, embedded database. This makes it ideal for portable security tools that are executed directly on field machines, such as penetration testing laptops or Raspberry Pi devices.

Key motivations for choosing SQLite include:

- **Simplicity:** Requires no server setup or maintenance, allowing the project to remain self-contained and easy to deploy.
- **Portability:** The entire database resides in a single file, which can be copied, backed up, or transferred effortlessly.
- **Performance:** For single-user environments and moderate data volumes (such as lists of WiFi networks and handshakes), SQLite offers excellent read/write performance.
- **Reliability:** SQLite is ACID-compliant and has proven stability, making it robust enough for data logging even in interrupted or harsh field conditions.
- **Integration:** Seamlessly integrates with Python (via `sqlite3` module) without the need for external libraries or drivers.

Given the scope and requirements of CrackMate, including local data storage, minimal setup, and offline capability, SQLite represented the most practical and efficient solution.

3.4 Website

To provide a clear and user-friendly visualization of the data collected and processed by CrackMate, a static website was developed using standard web technologies such as HTML, CSS, and JavaScript. The main purpose of the website is to offer an intuitive graphical interface that summarizes the project's outcomes and supports a better understanding of its core functionalities.

The website is organized into four main pages:

- **Home:** Presents an overview of the project, including its goals, scope, and the general functioning of CrackMate.
- **Map:** Displays a dynamic map that shows the geographic locations of detected WiFi networks. The map integrates coordinates, signal strength, encryption types, and, where available, recovered passwords and handshake capture status.
- **Statistics:** Provides visual representations of key metrics, such as the distribution of encryption protocols, signal power ranges, channel usage, and the success rate of handshake captures and password cracking attempts.
- **Info:** Contains a general description of the project's structure, methodology, and ethical considerations.

The interactive map was implemented using a JavaScript mapping library, allowing the integration of geolocation data stored in the local database.

3.5 Docker Usage Guide for CrackMate

3.5.1 What is Docker?

Docker is a lightweight virtualization platform that enables applications to run in isolated environments called *containers*. Each container includes everything needed to run the application: source code, libraries, dependencies, and a base operating system.

3.5.2 The Dockerfile

The `Dockerfile` includes the instructions required to build the Docker image. Its main components are:

- **Base**: `FROM kalilinux/kali-rolling` sets Kali Linux as the base OS.
- **Dependencies installation**: essential packages (Python, `aircrack-ng`, `hashcat`, `Pillow`, etc.) are installed via `apt-get`.
- **File copy**: project files are copied into the container's working directory.
- **Sudo permissions**: passwordless `sudo` access is enabled for the `root` user.
- **Startup command**: the container runs `python3 Script5.py` on startup.

3.5.3 Building the Image

To build the Docker image, navigate to the folder containing the `Dockerfile` and run:

```
docker build -t crackmate .
```

3.5.4 Running the Container

To start the container with GUI support (Tkinter) and access the integrated website:

```
docker run --rm -it \
  -e DISPLAY=YOUR_PC_IP:0.0 \
  -p 8000:8000 \
  --privileged \
  crackmate
```

Where `YOUR_PC_IP` is the IP address of the Windows machine running an X11 server (e.g., VcXsrv).

3.5.5 Accessing the Website

The web interface is served on port `8000`. After starting the container, it can be accessed via a browser at:

```
http://localhost:8000
```

3.5.6 X11 Support for GUI

To properly display the Tkinter graphical interface:

- Install and run an X11 server on Windows (e.g., `VcXsrv`).
- Start the container with the `DISPLAY` environment variable set to the Windows PC IP.

3.5.7 Limitations on Windows

- Direct access to USB devices (e.g., phones for ADB) and WiFi adapters in monitor mode is not natively supported by Docker Desktop on Windows.
- For advanced features, it is recommended to use a Linux virtual machine or run Docker directly on a Linux host.

Chapter 4

CrackMate

CRACKMATE

4.1 Architecture and Components

CrackMate is a Python-based desktop application designed for Wi-Fi network auditing. It follows a modular architecture with a clear separation of concerns to improve maintainability and ensure a responsive user interface.

The application uses `Tkinter` for the graphical user interface (GUI), which serves as the control center. Resource-intensive operations, such as *scanning*, *handshake capture*, and *password cracking*, are executed in **separate threads**. Communication between these worker threads and the main GUI thread is handled via a shared queue (`main_queue`), ensuring thread-safe updates to the interface.

4.2 Technologies Used

CrackMate leverages a combination of Python, both standard and third-party libraries, as well as external system tools to implement its full range of functionalities.

Main Language

- **Python 3.x**: The primary programming language for the entire application logic.

Core Python Libraries

- **Tkinter** (with `tkinter.ttk`): Used to build the desktop Graphical User Interface (GUI);
- **os**, **glob**, **shutil**: For filesystem interaction (e.g., directory creation, file handling, and file searching);
- **subprocess**: To launch and manage external processes (e.g., aircrack-ng tools, `ip`, `iw`);
- **threading**, **queue**: For concurrency and asynchronous communication between the GUI and backend tasks;
- **csv**: For reading and writing Comma-Separated Values files (used by `airodump-ng`);
- **re** (Regular Expressions): For text output parsing and string validation;
- **time**, **datetime**: For timing operations, timestamps, and duration tracking;
- **sqlite3**: For interacting with the local SQLite database.
- **Pillow (PIL)**: Optional; used for image handling and display (icons, logos);
- **io.StringIO**: To handle in-memory strings as file-like objects (useful for CSV parsing);
- **shlex**: For safe shell command parsing (even if not always required, it is a best practice);
- **signal**: To send process signals (e.g., `SIGTERM`, `SIGKILL`);
- **webbrowser**: To open the generated HTML map in the system's default browser;
- **traceback**: To retrieve detailed error information;
- **socket**: used to manage the port for the local server.

External Tools (Used via `subprocess`)

- **aircrack-ng suite**:
  - `airmon-ng`: To enable or disable monitor mode on wireless interfaces.
  - `airodump-ng`: For scanning Wi-Fi networks and capturing packets (including handshakes).
  - `aireplay-ng`: To perform deauthentication attacks and facilitate handshake capture.
  - `aircrack-ng`: To attempt WPA/WPA2 password cracking using dictionary attacks, and to verify handshake presence in `.cap` files.

- **iproute2** (`ip` command): To retrieve network interface information.
- **iw**: To inspect and configure wireless interfaces.
- **adb (Android Debug Bridge)**: Used to attempt GPS coordinate retrieval from a connected Android device.
- **sudo**: Required to execute commands with elevated privileges (nearly all `aircrack-ng` tools and network interface configuration require it).

4.3 Functions

4.3.1 WiFi Scanning and Capture

Wi-Fi network scanning and WPA/WPA2 handshake capture are core features of CrackMate.

Network Scanning:

- **Monitor Mode:** Before scanning begins, the selected wireless interface must be set to monitor mode. This is handled by the `monitor_mode_task` function, which executes:

  ```
  sudo airmon-ng check kill
  sudo airmon-ng start <interface>
  ```

- **Running airodump-ng:** The `scan_wifi_networks_task` function launches `airodump-ng` on the monitor interface:

  ```
  sudo airodump-ng <monitor_interface> --output-format csv -w <temp_scan_prefix>
  --write-interval 1
  ```

  - `-output-format csv`: Saves output in CSV format.
  - `-w <temp_scan_prefix>`: Prefix for output files. `airodump-ng` generates multiple files (e.g., `-01.csv`).
  - `-write-interval 1`: Updates CSV every second (although CrackMate uses a fixed scan duration).

- **Result Parsing:** After scanning, the most recent CSV file is processed by `parse_airodump_csv`, which extracts data from the Access Point section, including BSSID, ESSID, signal strength, channel, and encryption type. Security standards (WPA, WPA2, WPA3, OPN) are correctly interpreted.
- **Display:** The scanned networks are shown in a GUI table.

Handshake Capture:

- **Target Selection:** The user selects a network from the scan results.
- **Capture via airodump-ng:** The `capture_handshake_task` function runs:

  ```
  sudo airodump-ng <monitor_interface> --bssid <target_bssid> -c <target_channel>
  -w <handshake_filepath_prefix> --output-format cap --write-interval 5
  ```

  - `-bssid <target_bssid>` and `-c <target_channel>`: Filter traffic by BSSID and channel.
  - `-w <handshake_filepath_prefix>`: Saves traffic in a `.cap` file under `handshakes/`.
  - `-output-format cap`: Specifies output format.
- **Deauthentication (Optional but Included):** To accelerate handshake capture, `aireplay-ng` is periodically run:

  ```
  sudo aireplay-ng --deauth <num_packets> -a <target_bssid> <monitor_interface>
  ```

- **Handshake Verification:** While capturing, the `check_handshake_in_file` function runs `aircrack-ng` on the capture file:

  ```
  aircrack-ng <file.cap>
  ```

  If one or more WPA handshakes are found, the capture is marked as successful.
- **Saving and Notification:** Successful captures are stored; failed ones are deleted. The GUI is notified of the outcome.
- **Client Check (Optional):** The `check_for_clients_task` function can be run before capturing to verify if clients are connected to the target AP using:

  ```
  airodump-ng --bssid <target_bssid> -c <target_channel>
  ```

  An AP with active clients is more likely to produce a handshake.

4.3.2 Handshake Cracking

After a WPA/WPA2 handshake is captured, CrackMate can attempt to crack the password using a dictionary attack.

Selection: The user selects one or more `.cap` files and a valid wordlist path (e.g., `rockyou.txt`).

Cracking via aircrack-ng: The `crack_handshake_task` function launches:

```
sudo aircrack-ng -a2 -b <bssid> -w <wordlist_path> -l <output_cracked_file>
-q <handshake_cap_file>
```

- `-a2`: WPA (PSK) attack mode.
- `-b <bssid>`: Specifies the target AP BSSID.
- `-w <wordlist_path>`: Dictionary file path.
- `-l <output_cracked_file>`: Output file for found passwords (in `cracked_passwords/`).
- `-q`: Quiet mode.

Process Monitoring: The process runs in a separate thread and can be manually stopped by the user.

Result Analysis:

- **Password Found:** If `aircrack-ng` finds the password, it is saved in the output file and parsed from either file or stdout (`KEY FOUND! [ password ]`). It is then saved to the SQLite database and shown in the GUI.
- **Password Not Found:** If no match is found, the GUI is updated accordingly.
- **Error/Timeout:** Failures due to timeouts or errors are logged.

Storage: Cracked passwords are stored in the SQLite database, linked to their BSSID. CrackMate focuses solely on dictionary attacks, relying on aircrack-ng's capabilities. More complex rule-based or bruteforce attacks are not implemented.

4.3.3 Database Management

CrackMate uses SQLite to persist scanned networks, captured handshakes, and cracked passwords.

Table Structure: The `networks` table is created (if not exists) as:

```
CREATE TABLE IF NOT EXISTS networks (
    bssid TEXT PRIMARY KEY,
    ssid TEXT,
    power INTEGER,
    channel TEXT,
    encryption TEXT,
    latitude REAL,
    longitude REAL,
    last_seen TEXT,
    handshake_captured INTEGER DEFAULT 0,
    password TEXT DEFAULT NULL,
    handshake_filepath TEXT DEFAULT NULL
);
```

ON CONFLICT Update Logic: Data insertions use `ON CONFLICT(bssid) DO UPDATE SET ...` to ensure data quality:

- `ssid`: Updated if new SSID is non-null or better than `<Hidden>`.
- `power`: Updated if the new signal is stronger or missing previously.
- `channel, encryption`: Updated if new values are provided.
- `latitude, longitude`: Updated if coordinates are newly available.
- `last_seen`: Always updated.
- `handshake_captured`: Set to 1 if capture succeeded (never reset to 0).
- `password`: Updated if newly discovered and differs from previous.
- `handshake_filepath`: Updated if a new path is given.

Usage: The `save_scan_data_task` function manages database writes, invoked after scan, capture, or cracking phases.

4.3.4 Local Server

In order to start the website via the CrackMate application, an internal and local server has been developed. This little server uses the port 8080 as default in order to start the website locally in the default browser. This is possible via two functions, the start_http_serve function which is responsible for initiating the server. It first performs prerequisite checks, such as port availability, the presence of the python3 command, and the existence of the web files directory. If these conditions are met, it launches Python's built-in HTTP server as a separate process, monitoring its startup and handling any errors with user notifications and logs. Conversely, the stop_http_server function handles the termination of the previously started server. It implements a robust shutdown strategy, initially attempting a graceful close and, if that fails or takes too long, resorting to more forceful methods, while also accounting for operating system differences.

Chapter 5

Experiments

5.1 Objective

The primary objective of this experiment was to assess the security posture of Wi-Fi networks within specific areas of the city of Bari. This involved actively scanning for available wireless networks, attempting to capture WPA/WPA2 handshakes from susceptible networks, and subsequently endeavoring to crack the captured handshakes to recover network passwords using dictionary-based attacks. The experiment aimed to evaluate the feasibility of these common attack vectors in a real-world urban environment.

5.2 Experimental Setup

The experiment was conducted to simulate realistic conditions for Wi-Fi security auditing. The study was carried out in the city of Bari, Italy. Data collection, including network scanning, handshake capture, and password cracking attempts, was focused on the following specific zones (as identified from the provided map):

- Via Giovanni Amendola;
- Uniba Campus;
- Via Adolfo Omodeo;
- Via Edoardo Orabona;
- Via Gaetano Salvemini;
- Viale Giuseppe di Vittorio;
- Viale della Repubblica;
- Via Gioacchino Toma;
- Via Giuseppe Zanardelli;
- Via Guido De Ruggiero;
- Via Giuseppe Re David;

- Via Domenico Cirillo;
- Via Maria Cristina di Savoia;

These areas were chosen to represent a mix of residential, commercial, public, and potentially student-populated zones. The total duration for network sampling, including active scanning and handshake capture attempts across the designated zones, was approximately 8 hours. Password cracking attempts were performed subsequently. As utilized hardware tools there are:

- Two laptop computers running a Kali linux operating system via virtual machine (VMWare);
- Two wireless network interface card (NIC) that posses monitor mode for network scanning. The two models used are AC650 by BrosTrend and AC-1200. ;
- An Android phone to get the geolocation position via ADB with an usb cable.

Chapter 6

Results

6.1 Table of Detected Wi-Fi Networks (Revised)

TABLE 6.1: List of Detected Wi-Fi Networks with Technical Details (condensed version)

SSID	BSSID	Power (dBm)	Channel	Encryption	Band
eduroam	78:F1:C6:67:B9:AE	-79	100	WPA2 CCMP MGT	5 GHz
FASTWEB-RLKXG4	C0:9F:51:7F:EC:B7	-75	100	WPA2 CCMP PSK	5 GHz
TP-Link_AFFC_5G	50:D4:F7:7D:AF:FB	-68	36	WPA2 CCMP PSK	5 GHz
Mothprú	98:25:4A:F4:09:D5	-77	40	WPA2 CCMP PSK	5 GHz
TIM-27189876	E6:00:6A:A5:53:FA	-71	11	WPA2 CCMP PSK	2.4 GHz
FASTWEB-CASINA - 5 ghz	22:B0:01:F8:22:8F	-83	100	WPA2 CCMP PSK	5 GHz
Linkem_5038D3	80:02:9C:50:38:D5	-78	40	WPA2 CCMP PSK	5 GHz
TIM-42829962	3C:98:72:71:4D:B5	-53	64	WPA2 CCMP PSK	5 GHz
Linkem_63160C	80:02:9C:63:16:0D	-67	8	WPA2 CCMP PSK	2.4 GHz
Home&Life SuperWiFi-8C05	54:83:3A:BA:8C:06	-63	100	WPA2 CCMP PSK	5 GHz
SKYWIFI_WAJD7	00:A3:88:7C:F7:85	-80	13	WPA2 CCMP PSK	2.4 GHz
D-Link-3A7121	A8:63:7D:3A:71:23	-62	56	WPA2 CCMP PSK	5 GHz
FASTWEB-HTPCAL	98:3B:67:F2:68:07	-62	116	WPA2 CCMP PSK	5 GHz
FASTWEB-E4REUY	9C:63:5B:F3:9F:7D	-60	116	WPA2 CCMP PSK	5 GHz
TP-Link_9B98	30:DE:4B:1F:9B:97	-65	40	WPA2 CCMP PSK	5 GHz
WINDTRE-E8FAF8	DC:62:79:E8:FA:FA	-65	100	WPA3/WPA2 (SAE/PSK)	5 GHz
SKYWIFI_FVQNX	3C:9E:C7:C8:DE:35	-78	44	WPA2 CCMP PSK	5 GHz
ADB-DC166A	20:83:F8:DC:16:6C	-69	112	WPA2 CCMP PSK	5 GHz
Turnover Gaming Pub	EA:AA:3F:82:4D:31	-63	120	WPA2 CCMP PSK	5 GHz
Wind3 HUB - 0FD42B	78:90:A2:0F:D4:2C	-63	140	WPA2 CCMP PSK	5 GHz
TIM-11233976	82:59:43:64:F7:A1	-70	112	WPA2 CCMP PSK	5 GHz
Wind3 HUB-8329C2	B8:D5:26:83:29:C3	-56	52	WPA2 CCMP PSK	5 GHz
mestieri wifi_5G	48:22:54:CF:95:32	-68	36	WPA2 CCMP PSK	5 GHz
D-Link-72859C	58:D5:6E:72:85:A4	-53	100	WPA2 CCMP PSK	5 GHz
TIM-35235459	BE:49:43:79:81:70	-81	56	WPA2 CCMP PSK	5 GHz
Vodafone-BELL'ESSERE	BC:15:AC:78:A5:8D	-75	36	WPA/WPA2 CCMP TKIP PSK	5 GHz
FASTWEB-BF0663	A6:91:B1:BF:06:6B	-73	104	WPA2 CCMP PSK	5 GHz
FASTWEB-B864E1-5GHZ	A6:91:B1:B8:64:E9	-75	104	WPA2 CCMP PSK	5 GHz
Vodafone-34410052	64:59:F8:F3:5D:CC	-55	36	WPA2 CCMP PSK	5 GHz
Home&Life SuperWiFi-FE49	C8:54:4B:D0:FE:4A	-75	52	WPA2 CCMP PSK	5 GHz
TIM-48529551	F0:1B:24:20:68:7D	-66	52	WPA2 CCMP PSK	5 GHz
Linkem (5.0)	80:02:9C:05:F5:4E	-81	40	WPA2 CCMP PSK	5 GHz
Wind3 HUB - 4548BE	E0:19:54:45:48:BF	-81	13	WPA2 CCMP PSK	2.4 GHz
Wind3 HUB-9FE8E9	10:71:B3:9F:E8:EA	-88	52	WPA3/WPA2 (SAE/PSK)	5 GHz
DIRECT-MBUX 37495	B2:00:73:50:7B:98	-74	157	WPA2 CCMP PSK	5 GHz
FASTWEB-H3YFZ3	F4:F6:47:35:19:BD	-73	36	WPA2 CCMP PSK	5 GHz
TIM-70633676	7A:2D:1A:56:01:1F	-63	112	WPA2 CCMP PSK	5 GHz
TIM-65774794	DE:FD:73:BD:9D:78	-76	7	WPA2 CCMP PSK	5 GHz
FASTWEB-DFJCSU	D0:B6:6F:1A:EF:97	-63	36	WPA2 CCMP PSK	5 GHz
FASTWEB-DNYSG4	A4:F3:3B:B9:C1:ED	-77	56	WPA2 CCMP PSK	5 GHz
casa_nostra_5GEXT	D8:47:32:BB:F8:3B	-81	36	WPA2 CCMP TKIP PSK	5 GHz
Wind3 HUB-7FF5E5	10:71:B3:7F:F5:E6	-79	36	WPA3/WPA2 (SAE/PSK)	5 GHz
TIM-99361314	C0:49:43:FE:92:C5	-81	60	WPA2 CCMP PSK	5 GHz
TISCALI-6124	00:AA:BB:01:23:42	-77	161	WPA2 CCMP PSK	5 GHz
FRITZ!Box 5530 DE5	B0:F2:08:2B:EA:0B	-63	120	WPA2 CCMP PSK	5 GHz
FASTWEB-GRX4UG	48:3E:5E:55:9E:27	-61	124	WPA2 CCMP PSK	5 GHz
FASTWEB-K2CFY2	04:71:53:57:3B:87	-71	36	WPA2 CCMP PSK	5 GHz
FASTWEB-d742DK	DC:21:E2:F2:74:F8	-74	36	WPA2 CCMP PSK	5 GHz
Linkem_404C4F	58:13:D3:40:4C:51	-77	36	WPA2 CCMP PSK	5 GHz
TIM-22171461	12:13:31:52:4F:4D	-75	64	WPA2 CCMP PSK	5 GHz

(Continued on next page)

TABLE 6.1: (Continued)

SSID	BSSID	Power (dBm)	Channel	Encryption	Band
Wind3 HUB-E91B85	D4:3D:F3:E9:1B:86	-68	132	WPA3/WPA2 (SAE/PSK)	5 GHz
enel-WiFi_0EBB5A81	14:36:0E:BB:5A:82	-71	108	WPA3/WPA2 (SAE/PSK)	5 GHz
Hidden Network	E4:FA:C4:2E:7D:77	-54	2	WPA2 CCMP PSK	2.4 GHz
iliadbox-1C1AE8	3A:07:16:17:58:30	-66	112	WPA2 CCMP PSK	5 GHz
FASTWEB-CCZ3T7	A4:F3:3B:BA:27:51	-68	1	WPA2 CCMP PSK	2.4 GHz
FASTWEB-E3005F	A6:91:B1:E3:00:67	-72	1	WPA2 CCMP PSK	2.4 GHz
Vodafone-C02130076	80:16:05:DB:21:7C	-71	8	WPA2 CCMP PSK	2.4 GHz
Vodafone-A81588633	E4:8F:34:B8:80:56	-69	100	WPA2 CCMP PSK	5 GHz
Wind3 HUB-05D1B1	88:AC:C0:05:D1:B2	-78	36	WPA2 CCMP PSK	5 GHz
TIM-49259727	CE:6F:88:CE:EE:74	-81	40	WPA2 CCMP PSK	5 GHz
SKYGU7BV	04:81:9B:77:16:1D	-79	44	WPA2 CCMP PSK	5 GHz
Vodafone-C00370172	80:16:05:C8:9A:9C	-55	44	WPA2 CCMP PSK	5 GHz
TIM-14368161	A2:B5:3C:C6:EF:A9	-46	36	WPA2 CCMP PSK	5 GHz
Home&Life SuperWiFi-B09A	54:83:3A:D1:B0:9B	-72	52	WPA2 CCMP PSK	5 GHz
FASTWEB-TCDRFH	F4:F6:47:35:6D:11	-53	56	WPA2 CCMP PSK	5 GHz
FASTWEB-PJPJYE	F4:23:9C:98:C9:C7	-49	100	WPA2 CCMP PSK	5 GHz
FARMACIA CAPEZZUTO	D8:EC:E5:5D:90:35	-58	36	WPA2 CCMP PSK	5 GHz
PosteMobile_C829CF_5G	60:14:66:CA:29:CF	-83	48	WPA2 CCMP TKIP PSK	5 GHz
Rete da craccare -02	0A:3A:3A:F6:07:1E	-38	5	WPA2 CCMP PSK	2.4 GHz

6.2 Discussion of Results

The provided CSV dataset lists a total of 69 Wi-Fi access points (considering unique entries by BSSID, although some have partial data). The analysis of this data reveals several trends and peculiarities in the surveyed wireless environment. The table above presents a selection of the most relevant fields for an overview, omitting for brevity the column indicating handshake capture, information which is nevertheless considered in this discussion based on the original data.

6.2.1 Frequency Band Distribution

From a total of **69 access points scanned**, the distribution by frequency band was observed. Most networks operate on the **5 GHz band**. The analysis showed the following:

- **5 GHz**: 60 networks (accounting for **86.95%**)
- **2.4 GHz**: 9 networks (accounting for **13.05%**)

The predominance of the 5 GHz band is typical of modern environments, given its greater bandwidth and lower interference compared to the congested 2.4 GHz band.

6.2.2 Encryption and Security

The analysis of encryption types used by the **69 access points** revealed that the vast majority use **WPA2 with PSK (Pre-Shared Key)** and CCMP encryption, which is considered a robust security standard for home and small business use. The distribution of encryption types was as follows:

- **WPA2 CCMP PSK**: This was the most common type, found in **60 access points (86.95%)**.

- **WPA2 CCMP MGT**: Present for the "eduroam" network, indicating an enterprise-type network using WPA2-Enterprise with centralized credential management (typically RADIUS), not a PSK. This type, along with the subsequent ones, constitutes the remaining access points.
- **WPA3/WPA2 (SAE/PSK)**: Some networks (e.g., WINDTRE-E8FAF8, Wind3 HUB-9FE8E9) show a mixed WPA3/WPA2 mode, indicating support for the newer WPA3 standard (with SAE - Simultaneous Authentication of Equals) while maintaining compatibility with WPA2 devices. This is a positive sign towards enhanced security.
- **WPA/WPA2 CCMP TKIP PSK**: One network ("Vodafone-BELL'ESSERE") uses a mixed WPA/WPA2 mode that includes TKIP. TKIP is an older and less secure protocol than CCMP (AES), and its presence, even in mixed mode, could represent a potential vulnerability if clients connect using WPA/TKIP.
- **WPA2 CCMP TKIP PSK**: Two networks ("casa_nostra_5GEXT", "PosteMobile_C829CF_5G") specify both CCMP and TKIP with WPA2. Here too, the presence of TKIP is discouraged.

No open networks (without encryption) or networks using obsolete and insecure protocols like WEP were detected.

6.2.3 Captured Handshakes

During the experiment conducted on the 69 scanned access points, handshakes were successfully captured for **27 of them (39.14%)**. For the remaining **42 access points (60.86%)**, handshakes were not captured. This outcome reflects the success rate of handshake capture during the data collection.

6.2.4 Detected Passwords

A noteworthy aspect is the presence of an explicitly recorded password in the CSV for the **FASTWEB-TCDRFH** network (BSSID: 'F4:F6:47:35:6D:11') and also the **Rete da craccare -02** (BSSID: '0A:3A:3A:F6:07:1E'). The recorded password is "password" for the first one and "627574746572666c793839" for the second, which decrypted is "butterfly89". This represents a **severe security vulnerability** for that specific network, as such a weak password is easily guessable and compromises the network's integrity.

6.2.5 Signal Strength (Power)

The signal strength of Wi-Fi networks varies based on factors such as the distance from the access points and eventual interferences (like walls). The collected data on signal strength (dBm) for the scanned networks showed several trends. Key findings from the signal strength data are:

- There were few networks with a strong signal, defined as stronger than -50 dBm (e.g., -45 dBm is stronger than -50 dBm). The data showed 3 networks in this category (2 networks in the -50 to -46 dBm range and 1 network in the -45 to -41 dBm range).
- The majority of networks had signal strengths between **-85 dBm and -61 dBm**. Specifically, 56 out of the 69 total scanned networks fell into this range, distributed across signal strength bins from -85 to -81 dBm up to -65 to -61 dBm.
- The most common signal strength range observed was **-75 dBm to -71 dBm**, which contained 15 networks.

6.2.6 SSIDs and Channels

The SSIDs observed were varied, with many following typical provider patterns such as TIM-xxxx, FASTWEB-xxxx, Vodafone-xxxx, and Wind3 HUB-xxxx. Additionally, a "Hidden Network" SSID was present, indicating an attempt to obscure the network, although its effectiveness as a security measure is limited. The channels used were diverse across both frequency bands. For the 9 access points operating on the **2.4 GHz band**, the analysis of channel usage showed a specific distribution. The majority of these APs, **66.66%** (corresponding to 6 APs), utilized channels **1, 8, or 13**, with each of these channels being used by 2 APs. The remaining **33.33%** of 2.4 GHz APs (3 APs) used channels **2, 5, or 11**, with each of these channels being used by 1 AP. While channels 1, 6, and 11 are traditionally recommended as non-overlapping in the 2.4 GHz band, the observed usage included channels 8 and 13, which can cause interference depending on regional regulations and actual spectral separation. Regarding the 60 access points on the **5 GHz band**, the analysis of channel usage revealed that the most prevalent listening channel was **channel 36**, employed by **14 access points (23.33%)**. This was followed by **channel 100**, used by **8 access points (13.33%)**. Numerous other less-used channels were also observed, including, but not limited to, channels **161, 157, 124 and others**. The overall distribution indicated a wide utilization of various channels within the 5 GHz spectrum, encompassing those that may be DFS (Dynamic Frequency Selection) channels.

Chapter 7

Countermeasures

7.1 How to Protect Against These Attacks

While tools like `aircrack-ng` and `airodump-ng` are widely used for penetration testing and educational purposes, they also demonstrate real-world vulnerabilities in wireless networks. To mitigate the risks associated with sniffing and password cracking attacks, the following countermeasures should be considered:

- **Use Strong Passwords:** Avoid short or common passwords. Use long, complex passphrases that include a combination of uppercase and lowercase letters, numbers, and symbols.
- **Enable WPA3 (if available):** WPA3 offers stronger encryption and protection against offline dictionary attacks. If both your router and devices support it, enable WPA3 in the router settings.
- **Disable WPS:** Wi-Fi Protected Setup (WPS) is vulnerable to brute-force attacks and should be disabled to prevent easy network access.
- **Hide the SSID (optional):** Although it does not provide real security, hiding the network name may reduce exposure to casual attackers.
- **Monitor Connected Devices:** Regularly check for unknown devices on your network. Most routers provide an interface to view connected clients.
- **Update Router Firmware:** Ensure that your router is updated with the latest firmware to patch known security vulnerabilities.
- **Use MAC Address Filtering (limited use):** While MAC addresses can be spoofed, filtering may still add a small layer of difficulty for attackers.
- **Educate Users:** Awareness is essential. Users should be trained to recognize phishing attempts and avoid connecting to rogue access points.

By implementing these best practices, users can significantly reduce their exposure to Wi-Fi-based attacks and enhance the overall security of their wireless environments.

7.2 Security Tips for Home and Enterprise Wi-Fi Networks

Securing your wireless network is crucial for preventing unauthorized access and protecting sensitive data. The following security tips are tailored for both home and enterprise environments:

Home Networks

- **Change Default Router Credentials:** Always change the default username and password for accessing the router's administrative interface. Default credentials are often well-known and easy to exploit.
- **Use WPA2 or WPA3 Encryption:** Ensure your router is configured with WPA2 or WPA3 encryption. WPA3 provides enhanced security and protection against modern attacks.
- **Set a Strong, Unique Wi-Fi Password:** Create a unique and complex Wi-Fi password to prevent unauthorized access. Avoid using easy-to-guess or default passwords.
- **Disable Remote Management:** Turn off remote management features unless absolutely necessary, as they can be exploited by attackers to gain control over the router.
- **Keep Router Firmware Updated:** Regularly check for and install firmware updates to ensure your router is protected against the latest vulnerabilities.
- **Use a Guest Network:** If possible, create a separate guest network for visitors or IoT devices. This keeps the primary network more secure.

Enterprise Networks

- **Use WPA2-Enterprise or WPA3-Enterprise:** Implement WPA2-Enterprise or WPA3-Enterprise, which uses RADIUS authentication to ensure secure access and network segmentation for users and devices.
- **Network Segmentation with VLANs:** Separate different network traffic using VLANs (Virtual Local Area Networks). This limits access to sensitive resources and reduces the impact of any potential security breach.
- **Continuous Network Monitoring:** Deploy intrusion detection and prevention systems (IDS/IPS) to monitor network traffic for suspicious activity and unauthorized access attempts.
- **Access Control Policies:** Enforce strict access control policies, including device authentication and MAC address filtering, to restrict network access to authorized users and devices only.

- **Disable SSID Broadcast (where appropriate):** Hiding the network SSID (Service Set Identifier) may prevent casual attackers from easily detecting the network, although it is not a full-proof security measure.
- **Employee Training on Security:** Conduct regular cybersecurity training for employees to ensure they are aware of phishing attempts, social engineering, and safe Wi-Fi usage practices.

By following these guidelines, both home users and enterprise network administrators can enhance the security of their wireless networks, reducing the likelihood of sniffing, cracking, and other forms of attacks.

Chapter 8

Conclusions

In this case study, we have explored the vulnerabilities of Wi-Fi networks by focusing on the sniffing and cracking of WPA2 networks using popular tools like `aircrack-ng` and `airodump-ng`. These tools provide valuable insights into the weaknesses of wireless protocols, highlighting the importance of robust security measures to safeguard network integrity.

8.1 Key Learnings

Throughout this case study, we learned about the process of capturing Wi-Fi traffic, extracting WPA2 handshakes, and using dictionary attacks to crack passwords. These practical exercises have demonstrated the real-world risks associated with weak passwords, improper encryption, and poorly configured wireless networks. Moreover, we have discussed various attack vectors, such as the vulnerability of the WPS protocol and the potential threats posed by unsecured public Wi-Fi networks.

8.2 Future Developments

While WPA2 encryption is still widely used, future testing could focus on assessing the security of WPA3, which provides improved protections against dictionary and offline attacks. Additionally, the study could be extended to include more advanced attack techniques, such as Evil Twin attacks, which involve setting up rogue access points to capture sensitive data from unsuspecting users.

Furthermore, research into other wireless security protocols, like WPA3's enhanced handshake mechanisms, and the development of new attack detection and prevention systems would be valuable to improve network security and protect against emerging threats.

In conclusion, this case study has not only illustrated the technical aspects of Wi-Fi attacks but has also emphasized the critical importance of strong encryption, network segmentation, and user education in defending against these threats.