**Wi-Fi deauth implementation & Prevention**

**A PROJECT REPORT**

***Submitted by***

Siddharth Singh

Harsh Kashyap

Jyotiraditya Singh

Varun Menon

**Semester – VI**

**Academic Year: 2024–25**



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

# College of Engineering and Technology

# SRM Institute of Science and Technology

SRM Nagar, Kattankulathur – 603203, Chengalpattu District, Tamil Nadu.

**ABSTRACT**

This report presents the development and demonstration of a Wi-Fi deauthentication attack module designed to simulate real-world wireless security threats. Deauthentication attacks exploit vulnerabilities in the IEEE 802.11 protocol by sending forged management frames to forcibly disconnect devices from a wireless access point. The project implements this attack using [ESP8266], allowing for controlled testing in a secure environment. The objective is to understand the mechanics of wireless network disruption, evaluate the ease of execution, and emphasize the importance of modern security protocols like WPA3 and Protected Management Frames (PMF). The report also discusses mitigation strategies to defend against such attacks, highlighting the need for secure Wi-Fi configurations in both personal and enterprise networks.

**OBJECTIVES**

* To demonstrate the vulnerability of Wi-Fi networks: Develop a deauthentication attack module to exploit weaknesses in the IEEE 802.11 protocol, showcasing how unauthenticated management frames can disrupt wireless connections.
* To understand and simulate denial-of-service attacks: Use tools such as [insert tool, e.g., scapy or ESP8266] to send forged deauthentication packets, simulating a real-world DoS scenario on wireless networks.
* To analyze the impact on connected devices: Observe and document how devices behave when subjected to deauth attacks, including disconnection patterns and recovery times.
* To test in a controlled and ethical environment: Ensure that all experiments are conducted within a secure, isolated network environment without violating any network security policies.
* To explore detection techniques: Investigate how deauth attacks can be identified using network monitoring tools such as Wireshark or Kismet.
* To propose and discuss prevention strategies: Highlight the importance of Protected Management Frames (PMF), WPA3, and other modern protocols in mitigating such attacks.

**CHAPTER 1**

**INTRODUCTION**

Wireless networks have become a fundamental part of modern communication, enabling seamless connectivity for devices ranging from smartphones to IoT systems. However, with the convenience of wireless communication comes significant security risks. One such vulnerability lies in the IEEE 802.11 Wi-Fi protocol, which is susceptible to management frame exploitation — particularly deauthentication attacks.

A deauthentication attack is a type of denial-of-service (DoS) attack where an attacker sends forged deauthentication packets to a client or access point, effectively forcing a disconnection. This is possible because management frames in older Wi-Fi standards are unencrypted, allowing attackers to spoof packets without needing authentication. Such attacks can be used to disrupt connectivity, intercept handshakes for cracking passwords, or test network resilience.

This project aims to implement a deauthentication attack module to simulate this vulnerability in a controlled environment for educational and research purposes. By understanding the inner workings of the attack, its execution, and its impact on wireless networks, we can better appreciate the importance of modern Wi-Fi security enhancements such as WPA3 and Protected Management Frames (PMF).

The report covers the technical implementation of the attack, its effects on targeted devices, potential detection mechanisms, and countermeasures to defend against such exploits.

**CHAPTER 2**

**LITERATURE SURVEY**

[1] Detection of De-Authentication DoS Attacks in Wi-Fi Networks: A Machine Learning Approach

Published in: 2015 IEEE International Conference on Systems, Man, and Cybernetics

Authors: Agarwal, M., Pasumarthi, D., Biswas, S., & Nandi, S.

This study introduces a machine learning-based Intrusion Detection System (IDS) to detect deauthentication denial-of-service (DoS) attacks in Wi-Fi networks. By employing various classifiers, the proposed system achieves over 96% accuracy in identifying such attacks, offering a cost-effective alternative to hardware or protocol upgrades.

[2] Preventing Wireless Deauthentication Attacks over 802.11 Networks

Published on: arXiv (2018)

Author: Arora, A.

This study proposes a session management system to prevent deauthentication attacks in 802.11 networks. The approach involves verifying deauthentication frames through session tokens, ensuring that only legitimate clients can initiate disconnection, thereby mitigating unauthorized deauthentication attempts.

[3] Wi-Fi Deauthentication Attack Against 802.11 Protocol

Published on: GeeksforGeeks (2024)

This article provides a comprehensive overview of deauthentication attacks exploiting the 802.11 protocol's vulnerabilities. It outlines the attack methodology, tools like aircrack-ng, and emphasizes the importance of Protected Management Frames (PMF) and WPA3 in mitigating such threats.

[4] Use 802.11w or WPA3 to Prevent De-Authentication Attacks in Your Wi-Fi Network

Published on: Medium (2023)

Author: Tay, K.

Discusses the implementation of 802.11w and WPA3 standards to secure management frames in Wi-Fi networks. By introducing mechanisms like Message Integrity Check (MIC) and Integrity

**CHAPTER 3**

**SYSTEM DESCRIPTION**

**HARDWARE SPECIFICATIONS**

1. **ESP[8266]**



Figure ESP 8266

The ESP8266 is a low-cost Wi-Fi microchip developed by Espressif Systems, designed to provide wireless connectivity for embedded systems. It features a built-in TCP/IP stack and can function as both a client and an access point, making it ideal for IoT applications. With its integrated 32-bit processor and multiple GPIO pins, the ESP8266 can run simple programs, connecting sensors, and handling communication tasks. Its compatibility with the Arduino IDE simplifies development, making it a popular choice for projects that require Wi-Fi connectivity, such as the deauthentication attack module in Wi-Fi networks.

1. **OLED**

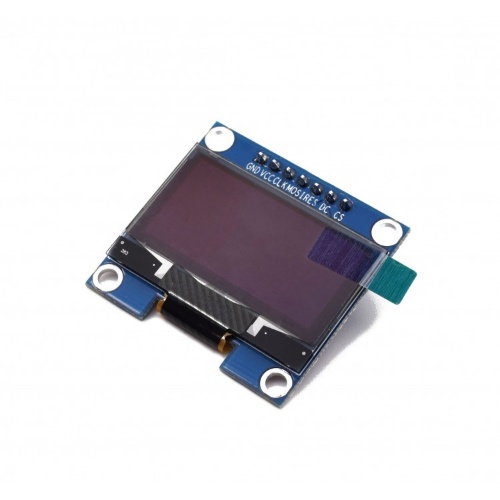


Figure SH1106

The SH1106 is a popular OLED display driver used for controlling monochrome OLED screens. It’s commonly paired with small, low-power OLED displays in embedded systems and IoT projects. The SH1106 can drive displays with resolutions such as 128x128 or 128x64 pixels, offering a clear and crisp output for text, graphics, and other visual information.

One of its main advantages is its low power consumption, making it ideal for battery-operated devices. The SH1106 communicates with microcontrollers like the ESP8266 or ESP32 through I2C or SPI protocols, allowing easy integration into projects. It’s typically used in devices like handheld gadgets, wearables, and sensor-based systems, where space is limited, and visual feedback is needed. The SH1106 supports various font and graphic display functions, which makes it highly versatile for showing system information or real-time status in your project.

**SOFTWARE SPECIFICATIONS**

**Arduino ide**

Arduino is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board. The Arduino platform has become quite popular with people just starting

, and anyone interested in creating interactive objects or environments. Arduino can interact with buttons, LEDs, motors, speakers, GPS units, cameras, the internet, and even your smart-phone or your TV! This flexibility combined with the fact that the Arduino software is free, the hardware boards are pretty cheap, and both the software and hardware are easy to learn has led to a large community of users who have contributed code and released instructions for a huge variety of Arduino-based projects

There are many varieties of Arduino boards (explained on the next page) that can be used for different purposes. Some boards look a bit different from the one below, but most Arduinos have the majority of these components in common.

A blue circle with white text and a plus and minus symbol

AI-generated content may be incorrect.

Figure ide

**PyCharm**

The Python program for the jammer and prevention simulation is designed to model the effects of jamming techniques on wireless communication and explore various methods for preventing these attacks. The program uses Tkinter for the graphical user interface (GUI) to allow users to interact with the system visually, selecting different jamming techniques and prevention strategies. The Matplotlib library is employed to plot and visualize critical data, such as the success rate of communication in the presence of jamming and the effectiveness of different prevention methods. The program simulates common jamming techniques like noise jamming and repeater jamming, showing how they impact signal quality. Additionally, the simulation includes options for prevention techniques, such as channel hopping, allowing users to toggle different methods and observe their effect on the communication system. The program aims to provide a comprehensive understanding of jamming and prevention in wireless networks, making it useful for both educational and research purposes.

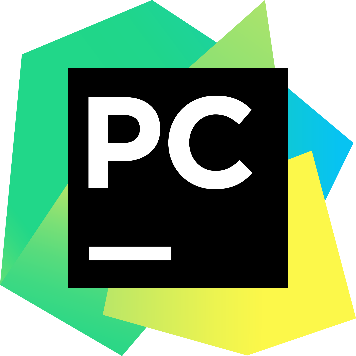


Figure PyCharm

**Block diagram of WIFI Deauth**

A diagram of a network

AI-generated content may be incorrect.

Figure Deauth ARCH

**Flow Chart**

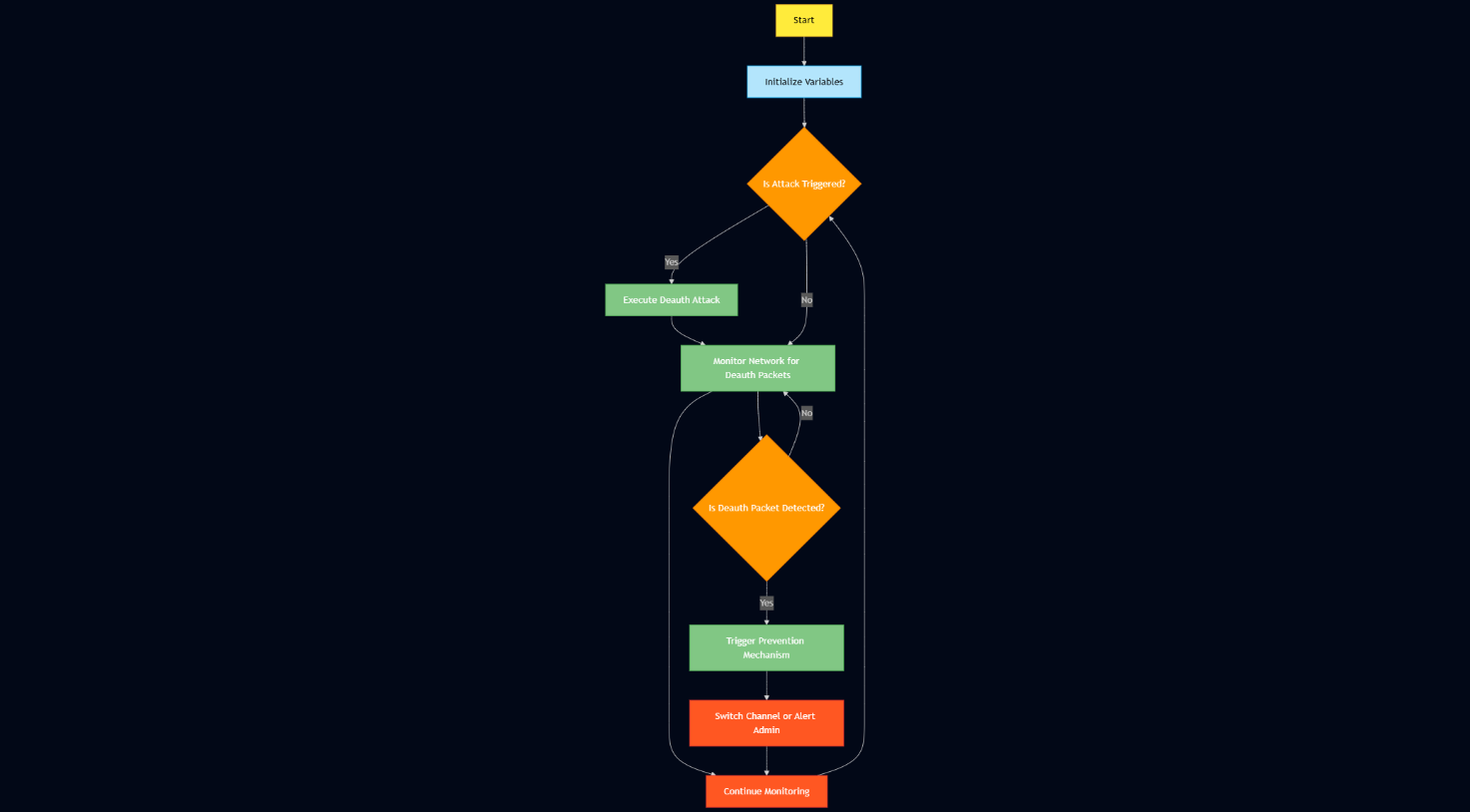
****

Figure Flow Chart

**CHAPTER 4**

**METHODOLOGY**

**Step 1: Setup and Network Discovery**

* **Action**: Set up the **ESP8266** module to scan and identify nearby Wi-Fi networks. Gather information about available access points (APs) and devices connected to them.

**Step 2: Target Selection**

* **Action**: From the scan results, select the target access point (AP) and the device (client) connected to it. This will be the target of the deauthentication attack.

**Step 3: Initiate Deauthentication Attack**

* **Action**: Using the **ESP8266**, send deauthentication packets to the target device. The packets will be sent with the intention of disconnecting the target device from the access point.

**Step 4: Monitor Attack**

* **Action**: Observe the effectiveness of the attack by monitoring the target device. Check if the device gets disconnected from the AP and whether it reconnects.

**Step 5: Prevention Mechanism**

* **Action**: Implement prevention techniques to protect the AP from deauthentication attacks:
  + Enable **802.11w** (Management Frame Protection) on the access point to protect management frames from being spoofed.
  + Use WPA2/WPA3 security protocols for encrypting the data and preventing unauthorized deauth frames.

**Step 6: Attack Detection**

* **Action**: Monitor the network for deauthentication frames. Detecting these frames indicates that a deauthentication attack is occurring. This can be done by checking for unusual patterns in network traffic.

**Step 7: Response to Attack**

* **Action**: Upon detection of a deauth attack, take immediate actions to mitigate its impact:
  + **Channel hopping**: Change the AP's operating channel to make it harder for the attacker to continue sending deauth frames.
  + **Temporarily disable the AP**: Disconnect the access point to stop the attacker’s access, and reconnect once the attack subsides.

**Step 8: Continuous Monitoring and Adaptation**

* **Action**: Continuously monitor the network for signs of deauthentication attacks. Use automated tools to adapt and prevent such attacks in real-time, ensuring that network traffic remains secure and stable.

**Software model development**

**A screenshot of a graph

AI-generated content may be incorrect.**

Figure Jammer Sim

The jammer simulation shown in the image demonstrates a Noise Jamming attack on a network, where the goal is to interfere with the normal communication by injecting noise into the frequency domain.

The Packet Success Rate graph (on the left) shows how the success rate of transmitted packets decreases over time, especially as noise is injected into the system. The red dashed line represents the threshold, marking the boundary at which communication is considered unreliable or blocked. The green line indicates the rate at which packets are successfully transmitted, and you can observe a sharp drop as the jamming starts affecting the network.

On the right, the Frequency Domain (FFT) graph shows the magnitude of the signal across different frequencies. The jamming signals cause significant disturbances, visible in the form of spikes at certain frequencies, which disrupt the regular communication frequencies.

This simulation provides insights into the effectiveness of Noise Jamming and how it impacts the network's ability to transmit data. It also highlights how the success rate of communication is directly affected by the presence of jamming signals, making it essential to implement mitigation strategies to counteract these types of attacks.

**Hardware model development**

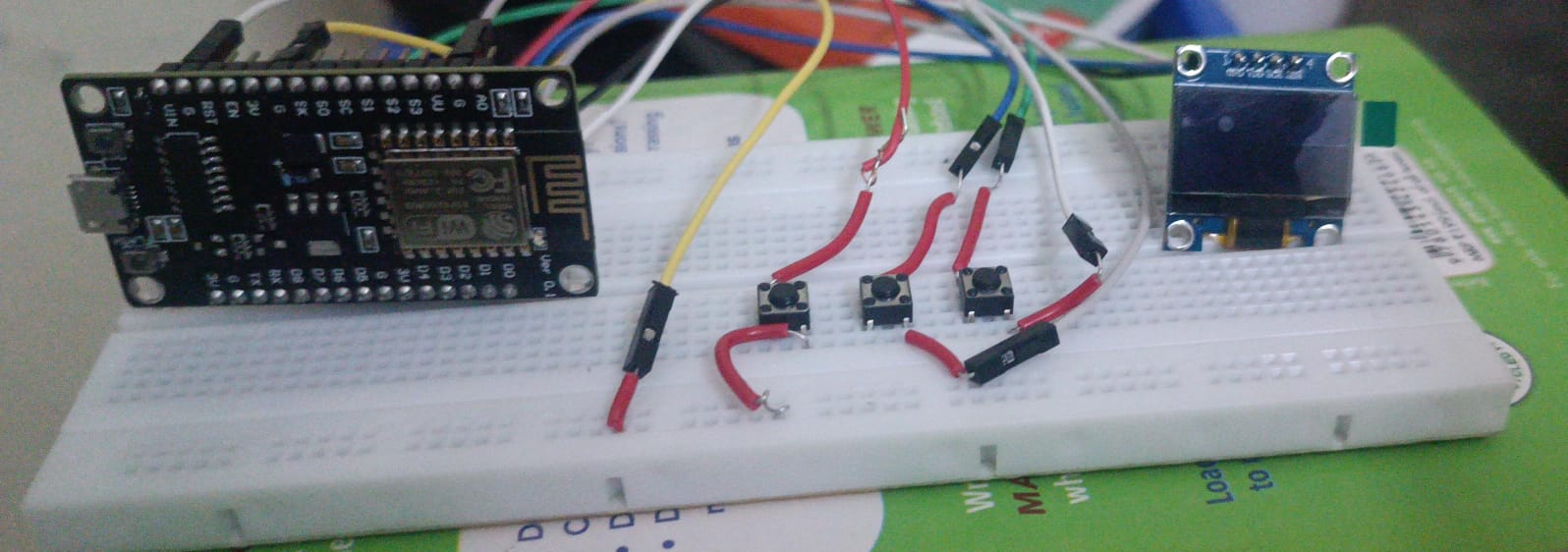


Figure Deauth device

The hardware setup consists of a NodeMCU (ESP8266) microcontroller, four tactile push buttons, and a 0.96-inch OLED display, all neatly arranged on a breadboard. The NodeMCU serves as the core of the system, handling input from the buttons and driving the OLED display via I2C communication. Each button is connected to a GPIO pin, allowing users to interact with the system—potentially to navigate menus, trigger functions, or simulate commands. The OLED display provides a clear visual interface for outputting text or status messages, making the setup suitable for interactive projects like IoT controllers, testing environments, or simple user interfaces. The breadboard and jumper wires enable quick prototyping and easy modifications, making this setup both practical and versatile.

**CHAPTER 5**

**CONCLUSION**

In conclusion, this hardware setup brings together key components—NodeMCU ESP8266, push buttons, and an OLED display—into a versatile and compact system capable of handling a range of interactive tasks. The NodeMCU provides robust processing and Wi-Fi connectivity, making it ideal for IoT applications, while the push buttons serve as reliable input controls for navigating menus or triggering actions. The OLED screen enhances user interaction by offering clear visual output in real time. This setup is not only cost-effective and beginner-friendly but also scalable, allowing future upgrades like sensor integration or cloud connectivity. Overall, it serves as a solid foundation for building innovative and practical embedded systems.

Alongside the physical hardware, a simulation of the system was developed to test the functionality in a controlled environment without needing constant hardware interaction. This simulation aids in rapid prototyping, debugging, and demonstrating the system’s behavior before deployment. Furthermore, to ensure the security and reliability of the communication—especially in wireless environments—techniques like channel hopping were considered. Channel hopping mitigates risks like interference and jamming by dynamically switching the communication channel at regular intervals, thereby enhancing the robustness of the system. These preventive strategies make the project more suitable for real-world IoT applications, where consistent performance and security are critical.

**REFERENCES**

[1] Singh, Rishabh, et al. "WiFi Deauth and Cloning using ESP8266." *2022 5th International Conference on Advances in Science and Technology (ICAST)*. IEEE, 2022.

[2] Gillan, Charles J., and Denis Guilhot. "Overcoming Wifi Jamming and other security challenges at the Edge." *Computing at the EDGE: New Challenges for Service Provision*. Cham: Springer International Publishing, 2022. 183-211.

[3] Gil-Martínez, Alejandro, et al. "Wi-Fi direction finding with frequency-scanned antenna and channel-hopping scheme." *IEEE Sensors Journal* 22.6 (2021): 5210-5222.

[4] Pirayesh, Hossein, and Huacheng Zeng. "Jamming attacks and anti-jamming strategies in wireless networks: A comprehensive survey." *IEEE communications surveys & tutorials* 24.2 (2022): 767-809.

[5] Sharma, Himanshu, Neeraj Kumar, and Rajkumar Tekchandani. "Mitigating jamming attack in 5G heterogeneous networks: A federated deep reinforcement learning approach." *IEEE Transactions on Vehicular Technology* 72.2 (2022): 2439-2452.

[6] Singh, Rishabh, et al. "WiFi Deauth and Cloning using ESP8266." *2022 5th International Conference on Advances in Science and Technology (ICAST)*. IEEE, 2022.

[7] Tamilselvi, S., R. Akash, and M. Hariharan. "IoT based WiFi Penetration Testing Toolkit." *Grenze International Journal of Engineering & Technology (GIJET)* 10 (2024).

[8] Hussain, Ahmed, et al. "Jamming detection in iot wireless networks: An edge-ai based approach." *Proceedings of the 12th International Conference on the Internet of Things*. 2022.