Embedded Systems
OpenWrt WiFi Multi-Channel Sniffer

Giovanni Baccichet
10851745

Politecnico di Milano — June 21, 2023

1 Introduction

External Resources Everything disclosed in this report (except for the network dumps, due to privacy reasons) can be found in this repository: github.com/GiovanniBaccichet/openwrt-multi-sniffer.

The ever-increasing integration of wireless technology into our everyday lives brings with it a spectrum of challenges and opportunities. One such opportunity resides in the analysis of WiFi probe request frames which are inherently broadcasted by WiFi-enabled devices as they search for available networks. By performing a detailed analysis of these frames, we can gain insightful knowledge about the device behavior, network congestion, security aspects, and much more.

This project aims to harness this potential by utilizing off-the-shelf hardware in combination with OpenWrt, a leading open-source router Operating System (OS). The primary goal is to develop a cost-effective and efficient system for **WiFi Probe Request frame sniffing over multiple channels**, specifically focusing on the 2.4 GHz and 5 GHz spectrum bands, traditionally used in WiFi communications.

Leveraging the inherent power and flexibility of OpenWrt, the system will be capable of capturing and analyzing WiFi Probe Request frames, thereby providing an accessible, in-depth perspective on the wireless activity in a given environment. This data can be crucial in a variety of scenarios, from troubleshooting network issues, cybersecurity investigations, to academic research, and more.

The system's functionality will be presented through OpenWrt's LuCi web graphical user interface (GUI), where users can view real-time data, and manage the overall operation. The user-friendly interface ensures that even those with little to no technical background can easily use the system. The collected data can be stored locally, offering an opportunity for subsequent detailed analysis or long-term trend observation.

This innovative approach combines the power of open-source software with the availability and affordability of consumer hardware, opening doors for a new kind of WiFi network analysis tool. This system has vast potential for a wide array of applications, such as network performance optimization, device tracking, studying human behavior through device movement, and contributing towards the broader fields of IoT and Smart Cities.

2 Background on IEEE 802.11

Active Scanning Active scanning is a proactive method employed by wireless devices, or stations, to identify Wi-Fi networks within their proximity.

When a device is set to actively scan, it releases a frame called a "Probe Request". This is the station's way of declaring its presence and asking nearby networks to respond. There are two types of probe requests: a broadcast probe, which is a general call-out to any network within range, and a directed probe, which is specifically aimed at a particular network identified by its Service Set Identifier (SSID).

Upon receiving a probe request, any access point (AP) within range responds with a "Probe Response" frame. This frame isn't simply an acknowledgement; it also serves as an information packet that carries specifics about the network. These specifics include the SSID, data rates that the network can support, the specific 802.11 physical layer (PHY) standard in use (like 802.11ac, 802.11n), channel usage, and much more.

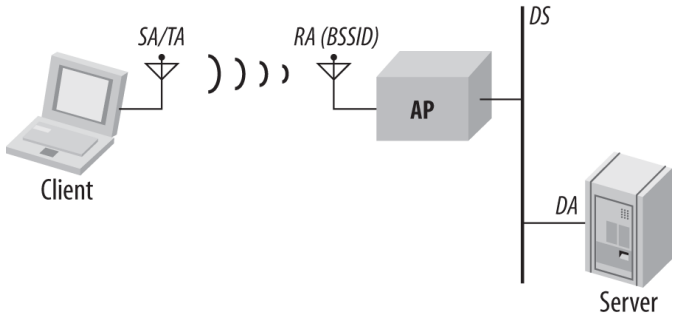

Figure 1: Address Field Usage in WiFi Networks

Having collected the probe responses from all APs within range, the station then has the task of selecting the most suitable network to join. This decision isn't just influenced by signal strength, but also factors in aspects like network congestion, supported security protocols, and others.

Zooming in on the probe request frame, one realizes that it's a container for a wealth of information. Most notably, it carries the source MAC address - a unique identifier attached to the device's network interface card (NIC). Traditionally, each NIC has a globally unique MAC address.

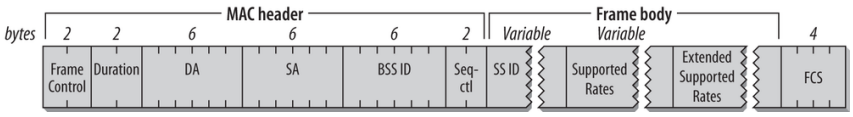

Figure 2: Probe Request Frame

This MAC address uniqueness has a flip side: it creates a potential privacy risk because the consistent use of a static MAC address allows for tracking of the device. To mitigate this risk, many modern devices employ a technique known as MAC address randomization. With this approach, the MAC address that the device uses to communicate on the network isn't static but changes periodically. Thus, even if a probe request is intercepted and the MAC address logged for tracking, the address would have changed by the next cycle, adding a layer of privacy protection.

However, while MAC address randomization helps, it is not a foolproof solution. Certain patterns in the randomization process or additional identifiable information in the probe requests may still enable tracking or identification of the device. Therefore, the evolution and robustness of privacy-protection mechanisms like MAC address randomization remain a topic of ongoing research and development.

With wireless communication becoming even more widespread, the interplay between technology, data analytics, privacy, and cybersecurity becomes ever more complex and intriguing. The continuous study of these facets within the realm of Wi-Fi probe request tracking is, thus, a pressing and compelling necessity.

3 Infrastructure Overview

The infrastructure for this project was designed with careful consideration to ensure optimal performance, robustness, and ease of use. We leveraged the capabilities of the `Linksys MR8300 v1.1` as the primary hardware component for our setup. This device is an advanced MU-MIMO Tri-Band WiFi router known for its superior performance and flexible functionality.

Running on `OpenWrt 22.03.2`, an open-source router operating system renowned for its stability and community support, the Linksys MR8300 allows us to implement and execute our WiFi probe request frame sniffing system smoothly. The OS's open nature provides us with the flexibility to modify and tweak the system as per our requirements.

Figure 3: Linksys MR8300 Router

The Linksys MR8300 comes with one 2.4GHz WiFi interface and two 5GHz WiFi interfaces, enabling us to cover the entire range of commonly used WiFi spectrum bands. This wide spectrum coverage is vital for our system to monitor and analyze WiFi probe request frames across multiple channels effectively.

One of the primary reasons we chose the Linksys MR8300 was its generous hardware specifications, making it highly suitable for data-intensive operations. It is equipped with 512 MB of RAM and 256 MB of NAND ROM, providing ample space for storing and processing data. Additionally, the powerful `Qualcomm IPQ4019` CPU ensures efficient processing and seamless operation of the system.

Moreover, the Linksys MR8300 boasts four 1 Gbps LAN ports and one 1 Gbps WAN port, allowing for high-speed, wired connections when needed. The USB 3.0 interface opens up possibilities for additional storage or other peripherals, and the four external antennas ensure excellent wireless coverage. Additionally, the device includes a Bluetooth chip, which could be utilized for other projects or applications.

The support and community around this device are robust, ensuring that any issues can be addressed swiftly and effectively. Its advanced features and capabilities make it a suitable choice for similar projects and applications, promoting flexible and powerful network solutions. Overall, the Linksys MR8300 running OpenWrt provides a solid foundation for our system's setup.

3.1 Remote File Storage

In addition to the already mentioned setup, there is potential to employ tunneling solutions such as Tailscale to transfer data to a remote network filesystem. This technology allows secure connections to be established over the internet, effectively creating a virtual network infrastructure that can span across multiple geographical locations.

Tailscale is a secure network connectivity tool that leverages WireGuard, a modern, high-performance VPN protocol. It is renowned for its simplicity, speed, and security. With Tailscale, you can create a virtual private network (VPN) that connects your device to a remote network, even across NAT, firewalls, or other restrictive network environments.

In the context of our project, Tailscale can be used to securely send the collected WiFi probe request frame data from our OpenWrt-based device to a remote network filesystem. This could be especially useful when the system is deployed in a remote or difficult-to-access location, or if you want to aggregate data from multiple devices into a central storage location for further analysis.

This remote network filesystem could be a dedicated Network Attached Storage (NAS) system, a server, or even a cloud-based storage service. The data can then be accessed, analyzed, and managed remotely, providing an extra layer of convenience and flexibility.

Using Tailscale or similar solutions, the data captured by our system can be transmitted securely and reliably over the internet, allowing for remote data analysis and storage. This extends the use cases for our project, making it even more versatile and effective for a wide range of applications.

4 Methodology

For the core functionality of sniffing WiFi probe requests, we leveraged two important tools: `tcpdump` and `iw`. `tcpdump` is a robust command-line packet analyzer, while `iw` is a flexible tool used for configuring Linux wireless devices. By integrating these tools, we ensured efficient and accurate data capture from the WiFi environment, using stat-of-the-art tools.

We then developed a shell script that uses these tools to put all wireless interfaces into monitor mode, which is a mode that allows WiFi devices to listen to all frames on the network. Following this, we used `tcpdump` to capture the management frames, specifically probe request frames. These frames are sent out by devices seeking to connect to WiFi networks, and they contain valuable information such as MAC address, RSSI (Received Signal Strength Indicator), and timestamp.

To make the data more consumable, we parsed the output from `tcpdump` to extract these data fields and formatted them in a JSON structure. The JSON format was chosen because of its wide acceptance and ease of use in data handling, particularly for web-based user interfaces.

Finally, we built a module in LuCI to display this information. A new page was created to present the data in a tabular form, with columns for each data field including MAC address, vendor, channel, RSSI, and timestamp. By using Lua scripting, we were able to manipulate the JSON data and render it effectively in the LuCI interface.

Throughout this process, we consistently prioritized the selection and use of open-source and widely-used tools, enabling future adaptability and extensibility of the project. We strived to ensure efficient data capture, processing, and presentation, all contributing to the final goal of providing a robust and user-friendly WiFi probe request sniffer.

5 Results

Here is the results, include screenshots and code snippets.