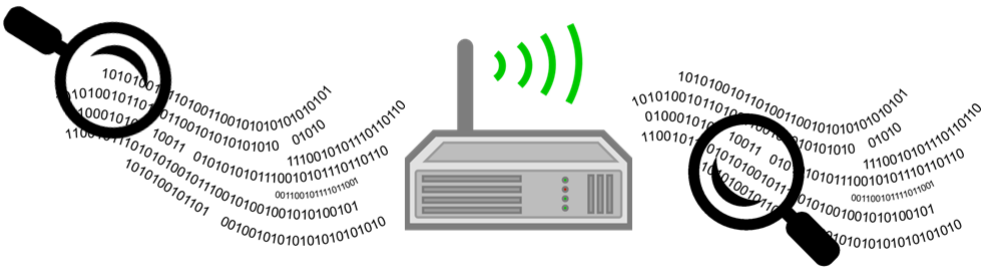
Covert Channel Defense

WiFi Microjamming Manual



Oren Shvartzman

November 26, 2022

Contents

1 Introduction

1.1 WiFi Microjamming - Theory of operation

The Goal of this attack, is to exfiltrate Information from a standalone network (not connected to the internet). This is done by taking advantage of the highly sensitive 802.11 CCA mechanism by using Micro-Jamming to send classified information. The victim's network architecture is two computers, one is standalone and one is connected to the internet via WiFi (as described in Figure 1).

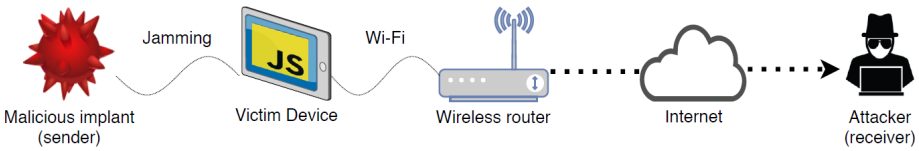


Figure 1: WiFi micro-jamming victim system architecture

First, it is assumed that there is a malicious device implanted in the victim's standalone network that collects classified data. In addition, this malicious device can transmit a CW signal in the 2.4GHz band.

The attack method is as follows. The attacker leads the victim's WiFi-connected device to a malicious website. The malicious website continuously measures DNS response times (without the victim's knowledge) and sends it to the attacker (the receiver of the classified information). Now, to leak the classified information, the malicious implant uses micro-jamming in the same frequency of the router in order to transmit classified information:

- To send '1', transmit a micro-jamming signal for T seconds.
- To send '0', do nothing for T seconds.

The attacker measures the average DNS response time every T seconds (starting in a predetermined time). If the average response time is larger than usual (because of the micro-jamming interference) – '1' was sent. otherwise, '0' was sent.

The micro-jamming is described in Figure 2. The leaked information is modulated using Pulse Amplitude Modulation (PAM). PAM is modulating the information by generating a square wave signal when transmiting '1' and no signal when transmiting '0'.

There are 2 main parameters that should be considered when operating this attack: f_j and D_j.

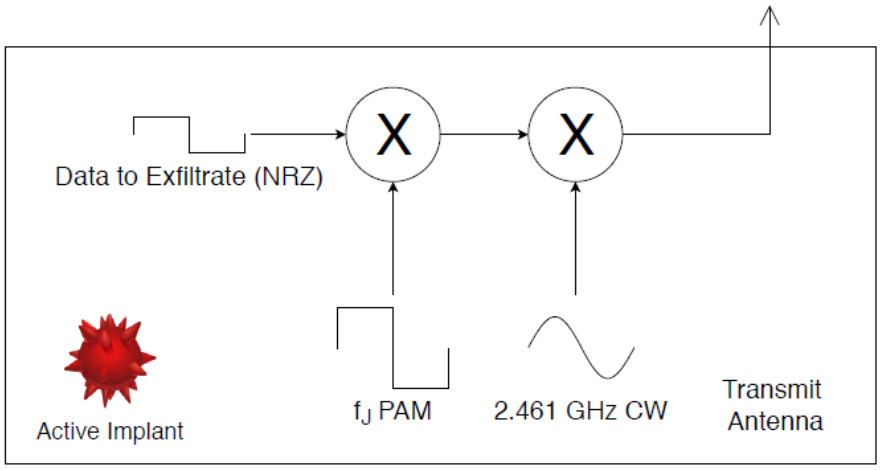


Figure 2: WiFi Micro-jamming malicous implant internals

f_j is the frequency of the PAM modulation. As said before, when '1' is sent, a square wave is generated. This square wave is multiplied by the information signal to get the modulated signal (seen in Figure 3). From the nature of PAM modulation, higher frequency of the square wave means the signal has less "stops". Now, you can't just jam the channel continuously because you only want to interfere the WiFi and delay the packets that are being sent. So, you must choose f_j in such a way that the channel won't be completely jammed but still interfered (and the packets are noticably delayed).

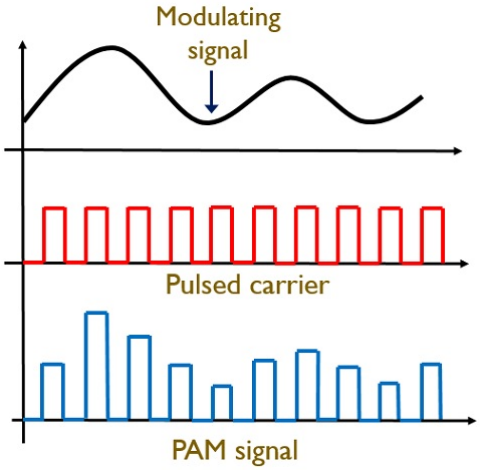


Figure 3: PAM modulation

D_j is the duty cycle of the PAM modulation square signal. In a single cycle of

the jamming signal, this is the percentage of time that you actually transmit and block the channel (as seen in Figure 4). A larger D_j increases the interference to the network traffic.

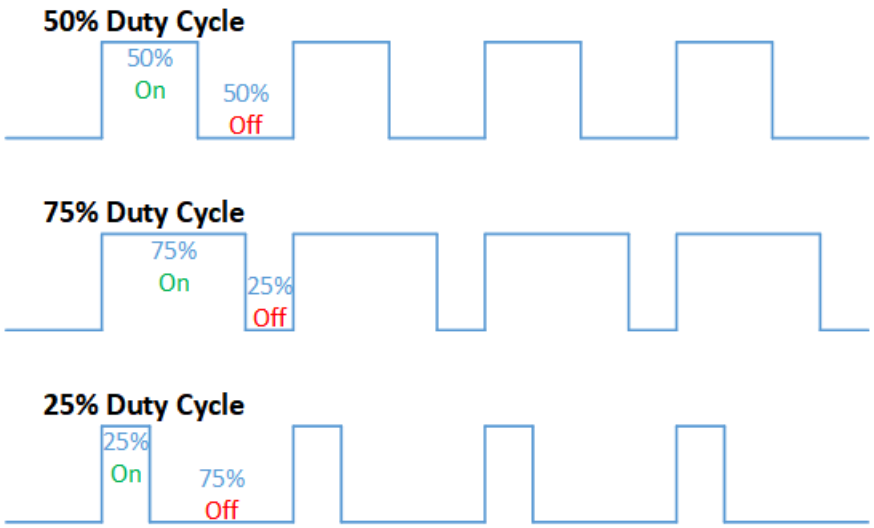


Figure 4: Duty Cycle

2 Setup

2.1 General setup idea

The following Figure 5 describe the general setup idea:

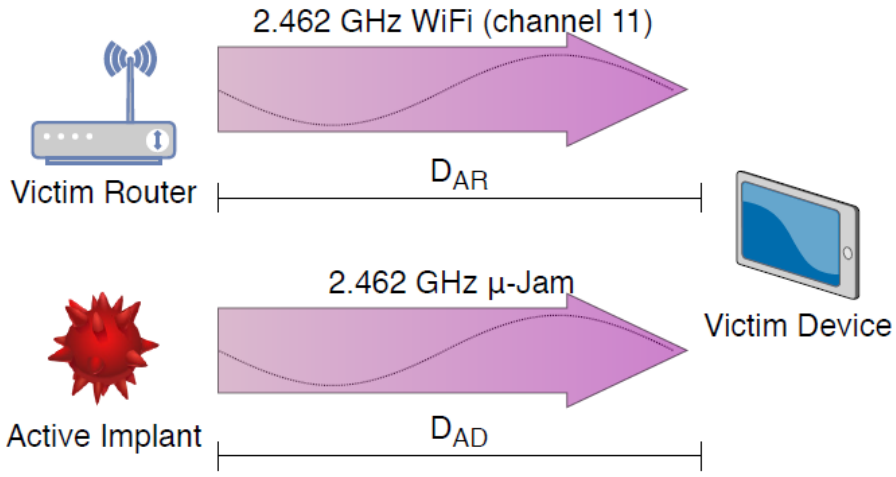


Figure 5: WiFi Micro-jamming general setup idea

As shown in Figure 5, there are a few components to the setup. First, the malicious implant which can transmit raw unmodulated CW signals in the 2.4GHz frequency band. Second, the router that'll process the DNS requests and answer them. Finally, the victim device which will access the malicious site and measure DNS response times.

2.2 Components

Figure 6 presents the actual setup of this attack:

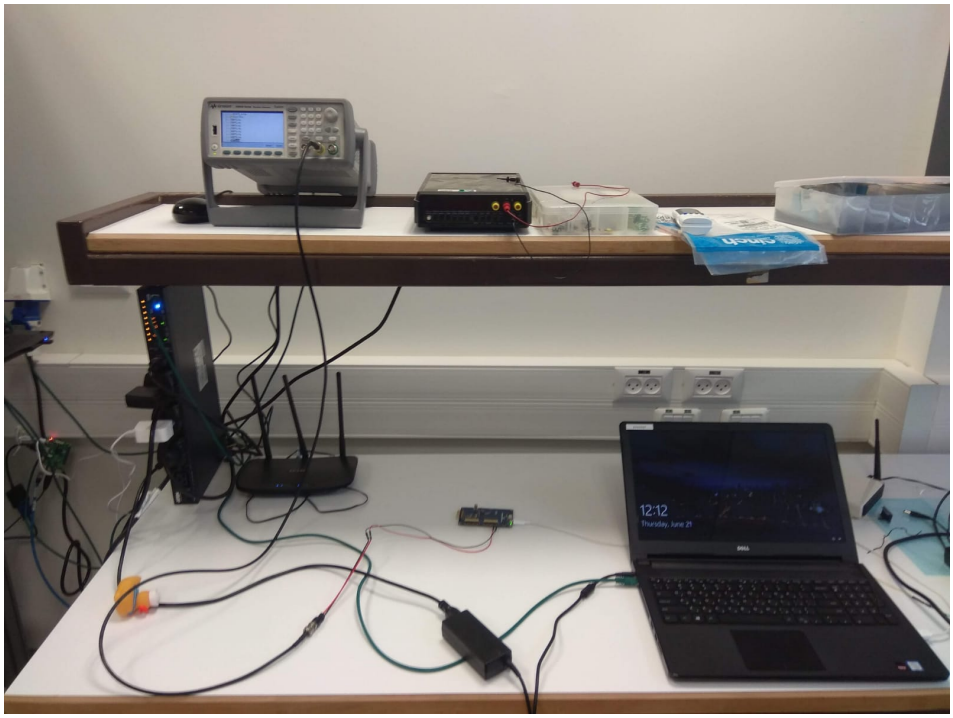


Figure 6: WiFi Micro-jamming setup in the lab

The components seen in the image:

- The malicious implant: This component is the Atmel ATMEGA256RFR2 Xplained Pro board. The code that controls its TX is called "mtk.c".
  To control the PAM modulation frequency (f_j), we'll connect the Atmel's external I\O to a wave generator: Keysight 33622A 120MHz Waveform Generator. We configured the generator with $f_j = 200Hz$, $D_j = 80\%$ (as seen in Figure 7). According to the original paper, these values are optimal for this attack. The data sent by the waveform generator is a psuedo random sequence called PN7 (see 3.1 for details)
- The router: This component is the TP-Link WR940N. This router will be connected via Wi-Fi to the laptop and will answer the DNS requests from the laptop.
- The laptop: This component is the laptop Dell Inspiron 5559. It'll run the script wifi_microjam.py which opens the malicious HTML (that makes DNS

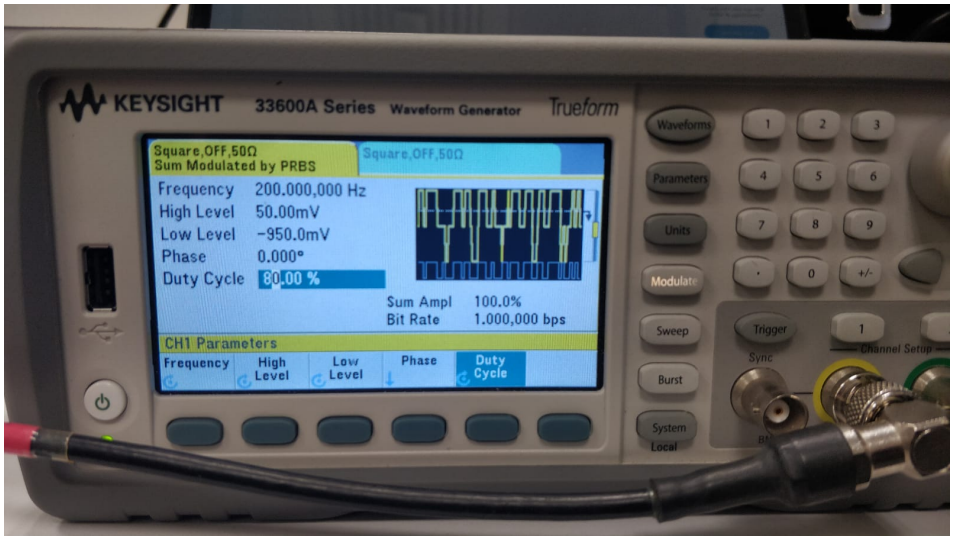


Figure 7: Signal generator configuration

requests) and records the response times from the router. It also control the wave generator which controls the Atmel. It is the heart of the attack setup.

2.3 Setup assembly

We'll now describe what connections exactly are needed for this setup:

- The laptop: connects to the router via WiFi.
- The waveform generator: connects to the laptop via Ethernet.
- The malicious implant (Atmel): connects to the laptop via USB. Connects to the waveform generator via jumper-BNC cable (the BNC is connected to the generator, the jumper wires are as seen in Fig. 8). It uses its internal WiFi antenna to transmit (no need for an external one)

2.4 How to use the setup

1. Put the micro-jam code in the Atmel
   (a) connect the Atmel to the laptop
   (b) open Atmel studio
   (c) open the project located in: `at05558_software/mtk_code_r1.3/LwMesh_1_1_1/apps/Template/astudio/atmel_mtk.atsln`

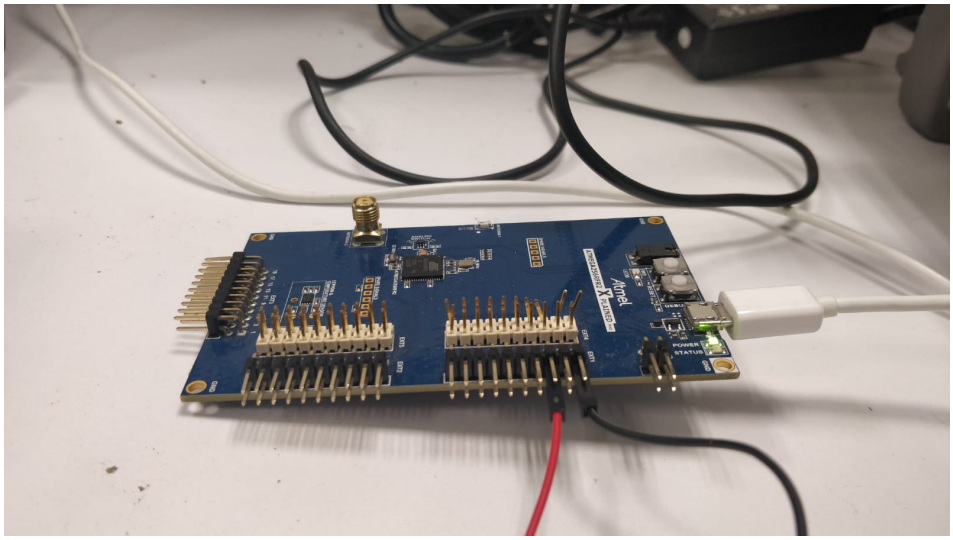


Figure 8: Atmel connection to the waveform generator (black = GND, red = Vcc)

(d) load the project to the Atmel (to change its functionaliy, edit the mtk.c file)

2. run wifi_microjam.py (the main script that controls the waveform generator and sends the DNS requests and measures the response times).

3 Useful details

3.1 PN7 sequence

The data sent from the waveform generator is the psuedo-random sequence PN7 which is 127 bit long. The PN7 sequence is:

{1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0}

3.2 Useful commands

- `cat /var/lib/misc/dnsmasq.leases` - who is currently connected to my DNS server (shows DNS name, IP and MAC) - used to check if the waveform generator is connected.
- `cat /etc/dnsmasq.conf` - all the configured connections on my DNS server (even the ones who aren't connected now) - used to check the wavform generator's IP (in case its not connected).
- `nc waveform 5025` - connect to the waveform generator. Now you can send SCPI commands (like *IDN? for its version)