WiFi Rogue AP Attack Detection

by
Dan Salmon

A research paper submitted in partial fulfillment for the degree of
Master of Science
in Information Technology
in the College of Graduate Studies & Research

Minnesota State University, Mankato
Fall 2018

Contents

1 Introduction

WiFi is a technology so ubiquitous that most of us take it for granted. We have access to it nearly everywhere and it just sort of works. It connects us to the world around us and allows us to learn, work, and be entertained. Most people don't even think about it when connecting to WiFi networks, but the reality is that attackers and ne'er-do-wells are always looking for an avenue into online accounts. These accounts can be very valuable. One of the ways attackers attempt to gain access to our data is to connect to WiFi networks.

Once on a network it's much easier to pivot and attack devices directly or to create fake web pages to phish credentials from users of that network. The main line of defense against attacks gaining access to wireless networks is the network password itself. To get this password, attackers commonly employ a tactic known as a "Rogue AP (Access Point) Attack". In this strategy, an attacker makes a malicious clone network that looks identical to the victim network. These attacks can be difficult to detect, even for the most discerning of users. In this paper an open-source tool that is free-to-use and helps defend against these attacks is outlined.

2 Project Motivation

Researching this topic, there were very few resources for home users found. The existing systems used for Rogue Access Point (RAP) detection are only available on enterprise wireless hardware which is very expensive. These vendors include Cisco, Ubiquiti, and Aruba. To create something home users can run, tool was created that will run on commodity hardware (read: cheap). The end result is a tool that can be run on any platform that can run Python 3 and can interface with a USB WiFi adapter configured in promiscuous mode. An effective use of this tool would be running it on a handful of inexpensive small form factor computers such as the Raspberry Pi or similar platform.

3 Literature Review of Related Work

Chumcu et al.[1] proposed a new algorithm for detecting spoofed MAC addresses which uses PLCP (Physical Layer Convergence Protocol). This is one of the heads of the IEEE 802.11 data frames. Since the PLCP header is based on the network card driver, it would be much harder to

[1]Chumchu, P., Saelim, T., & Sriklauy, C. (2011, January). A new MAC address spoofing detection algorithm using PLCP header. In Information Networking (ICOIN), 2011 International Conference on (pp. 48-53). IEEE.

spoof. They found in their experiment that they were able to detect MAC spoofing 100% of the time with no false positives.

Alotaibi et al.[2] were able to detect MAC address spoofing using a method called random forests to compare the received signal strength (RSS) of wireless frames. Utilizing some machine learning algorithms, they were able to achieve an averaged success rate of 93.77% at identifying spoofed MAC addresses.

In separate research, Alotaibi et al.[3] looked at all the existing proposals for detecting RAPs and classified them based on different criteria:

- Whether the system is active or passive
- If protocol modification is necessary
- How accurate the system is
- Whether special hardware is needed

They found that the technique type with the fewest weaknesses was the "Hybrid" technique. In it, a system passively monitors traffic both on the wired and wireless network simultaneously. The only weakness identified was that the detection system could be avoided from the wired side. This seems to be an acceptable risk.

4 802.11 Association Process

When a wireless client wants to connect to an 802.11 network, the first step it takes is to send a probe request to all APs in its proximity. In this probe request, the client specifies what data rates and capabilities it can support such as the 802.11 b, g, n, or ac standards. Since this probe request is being sent to the BSSID of ff:ff:ff:ff:ff:ff all APs that can detect the request will respond.

APs that receive this request will check the data rates and capabilities listed as supported and check if they can fulfill this request. If they can fulfill it, the AP will send back a probe response which includes the SSID, data rates, encryption types, and other capabilities it supports. The client then receives one or many of these probe responses and acts accordingly. Modern

[2]Alotaibi, B., & Elleithy, K. (2016). A new mac address spoofing detection technique based on random forests. Sensors, 16(3), 281.

[3]Alotaibi, B., & Elleithy, K. (2016). Rogue access point detection: Taxonomy, challenges, and future directions. Wireless Personal Communications, 90(3), 1261-1290.

wireless devices will search through the list of SSID's in the probe requests and determine which networks they have saved credentials to connect to.

After choosing which networks the client has credentials for, it will find the network with the strongest signal and start the authentication process. This authentication process differs based on the authentication type of the network. In a WPA2 network with a Pre-Shared Key (PSK), the AP requests the client send it the PSK in encrypted form. In a WPA2 network with 802.1x authentication, the AP sends the client a unique challenge to which the client needs to send a unique response.

Now that the client has negotiated the authentication process properly, the AP places the client in an authenticated state. This does not mean that the client can start receiving data from the AP, though, since a client can be authenticated to multiple APs on the same network simultaneously. This method of pre-authenticating to multiple APs aids in the process of roaming between them since a client can be authenticated to many APs but can only be associated and transfer data with a single AP. After the client has been authenticated, it can make a request to the AP to be associated with it. As long as this client is not associated with another AP already, the AP will send an association response, putting the client into the authenticated and associated state.

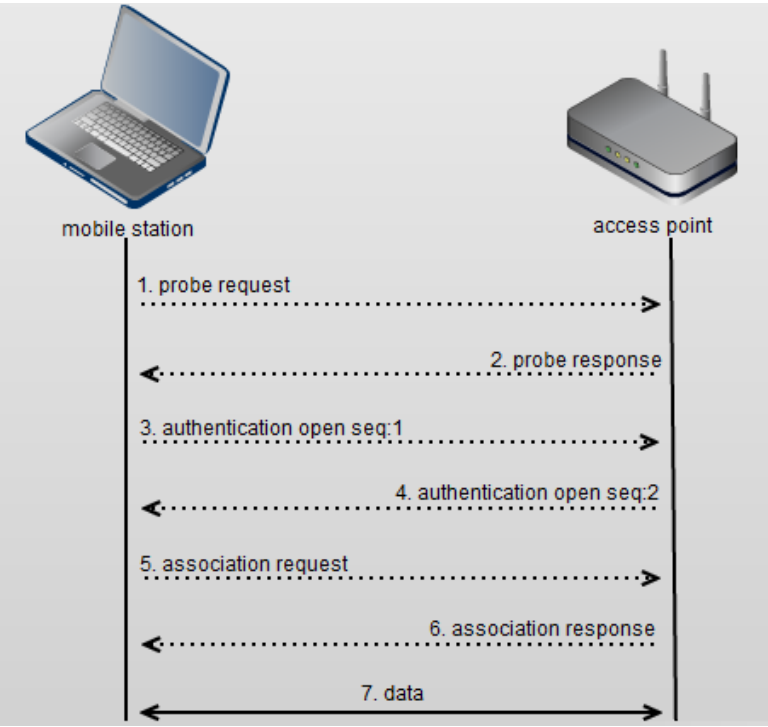

Figure 1: The 802.11 association process.
Image credits: Cisco - https://goo.gl/QZR64d

At this point, the client and AP can transfer data freely and from this time forward either of the two devices can disassociate or deauthenticate the client from the AP by sending the respective request. This can happen if the client leaves the proximity of the associated AP and wants to associate with another one that it has already authenticated to. This entire process is visualized in Figure 1.

5 Anatomy of a Rogue AP Attack

The crux of this attack is the fact that the 802.11b/g/n standards don't state that client devices have to maintain a whitelist of "known-good" MAC addresses when looking for known networking nearby. This means that a client device, when searching for a network to connect to, will blindly trust any router that is broadcasting an SSID that the client has connected to before. Thus, in a Rogue AP Attack, the attacker creates a network that is identical to the victim's known-good network in order to get the victim to connect to the evil network.

The end goal of the attack could be many things. The evil network could open a Captive Portal which looks identical to the victim's corporate or school network login page and get the user to enter their credentials, saving them in plaintext. Another common goal of a Rogue AP attack is to capture the handshake of the legitimate network.

After capturing this handshake, the attacker can then attempt to crack the pre-shared key of the network and subsequently gain access to said network. The only two features of the real network the attacker needs to copy exactly are the SSID and the encryption type. For example, the client device will not try to connect to a WEP network that has the same name as a WPA2 network that the client has connected to previously.

Another tactic an attacker might employ, in addition to creating the fake network, is to launch a "deauthentication attack" against the victim. To help speed up the process and ensure that the victim connects to the RAP, the attacker will send a flood of deauthentication packets to the victim. These packets are normally sent legitimately from an AP when it decides the client device needs to disconnect from it. This usually occurs in a multi-AP network (like a large college campus) when a client roams away from an AP and the network detects that another AP can provide a stronger signal to the client. The 802.11 b,g, and n protocol specifications do not require any additional checks that the deauthentication packets are coming from the real AP besides checking the source MAC address. Such an address can easily be faked resulting in

attackers flooding a victim with illegitimate requests. Tools like aireplay-ng[4], besside-ng[5], and wifite[6] can all automate this process.

6 Tool: WatchTower

6.1 Technical Details

WatchTower is written in Python and makes heavy use of the Scapy library. The only other necessary pip module is the requests library for sending Slack notifications, if so desired. Because it is written in Python, the tool can run on virtually any hardware that supports a GNU/Linux operating system running Python 3. The only requirement of the platform is that it can support a wireless card configured in "promiscuous mode" to allow full inspection of wireless packets. During development, both the TP-Link TL-WN722N and the Alfa AWUSA036NHA were used. Software license details are provided in Appendex II.

6.2 Implemented Features

WatchTower v0.1 has two main attack detection features. The first will detect rogue networks with identical SSID but having at least one of the following settings not properly cloned:

- Encryption type
- Authentication type
- Channel
- MAC address
- Cipher suite

The second detection feature will watch for abnormal amounts of deauthentication packets destined for a client of the known-good network. When the tool is run, it immediately starts listening for traffic between the known-good AP and any client who is authenticated and associated. Any time such traffic is heard, the client's MAC address is added to a list of known clients. The tool simultaneously is listening for any deauthentication packets destined for any

[4]https://www.aircrack-ng.org
[5]Ibid.
[6]https://tools.kali.org/wireless-attacks/wifite

of the known clients. Since deauthentication packets are part of the normal 802.11 connection lifecycle, the tool will watch for multiple packets being sent in a two second window.

To avoid an abundance of attack notifications, WatchTower has a five second rolling window to categorize new attack. If deauthentication packets are seen constantly for more than this five second window, another notification is sent out. This window length is a setting in the config file, so users will need to tune this to match their environment. In addition to detection, the tool will also send a notification of attack detection via Slack. This Slack notification can be delivered to either a private message to a single user or to a channel. Figure 2 shows a deauthentication attack alert while Figure 3 shows the RAP detection alert.

Figure 2: Slack PM with a WatchTower deauth alert

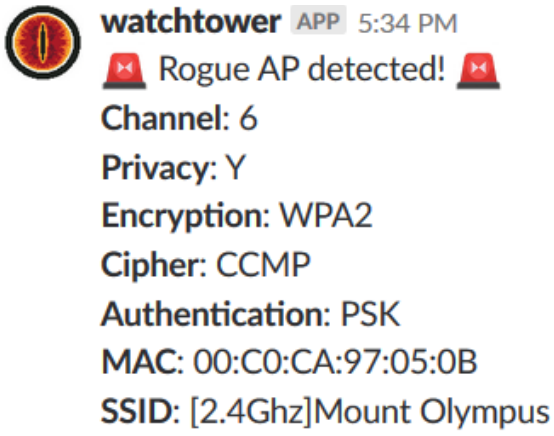

Figure 3: Slack PM with a WatchTower rogue AP alert

6.3 Limitations of the tool

Due to the open nature of WiFi traffic, this tool does not detect if an attacker is listening to the victim's channel and attempting to capture a handshake. This technically would not be possible, so the password strength of the pre-shared key is imperative. This tool will, however, detect if an attacker is actively creating a cloned network of the victim's network. It will also detect if an attacker is attempting to send a flood of deauthentication packets to the victim client or to the victim network AP. This tool will also not detect if an attacker has cloned his MAC address to match the legitimate AP MAC. This feature is currently on the v0.2 roadmap.

6.4 Effectiveness of Attack Detection

To test the effectiveness of WatchTower's attack detection, Kali Linux running in a VMWare virtual machine was used on a MacBook Pro. Two USB WiFi adapters were connected, one to act as the attacker and the other to act as the detector. Kali was chosen as it has many wireless drivers and WiFi auditing tools built-in. Four different offensive tools were tested while simultaneously running WatchTower, the results of which are listed in Figure 4 with the raw commands listed in Appendix III.

Tool	WatchTower Detected?	Attack Result
wifite	Yes	WPA2 handshake captured after deauthenticating client
besside-ng	No	WPA2 handshake captured after deauthenticating client
aireplay-ng	Yes	Deauthentication successful – client reconnected to a different network
airodump-ng	No	WPA2 handshake captured after waiting 15 minutes for an association

Figure 4: Deauthentication experimental results

Overall, WatchTower was able to detect most of the active attacks launched. Though it is unknown why the besside-ng attack was not detected, the fact that airodump-ng was also not detected is to be expected. Airodump-ng was added as an example of a passive attack that tools like WatchTower will not be able to detect.

To test the rogue AP detection functionality, the same hardware setup was used. With WatchTower running on one wireless interface, the other created a network which was nearly-identical to our monitored network using hostapd[7]. In each test case, the configuration was changed by one network setting to simulate an attacker who didn't clone the target network exactly right. The results of this test can be seen in Figure 5.

[7]https://wireless.wiki.kernel.org/en/users/documentation/hostapd

Network Setting	Good network value	Rogue AP network value	WatchTower detected?
Encryption type	WPA2	WPA	Yes
Channel	6	1	Yes
MAC address	B0:39:56:0E:E7:12	00:C0:CA:97:05:0B	Yes
Cipher suite	CCMP	TKIP	Yes

Figure 5: hostapd experimental results

7 Future Work

After researching the related work included in the above literature review, there were some ideas of features to add to this tool. Here are some that were not able to be added to v0.1 due to time constrains:

- Detect a RAP with all the same settings but significantly different received signal strength (RSS)
- Detect a RAP with identical settings by sending out an authentication request for our monitored SSID and checking if we get more than one response from APs with the same MAC
- Find why the beside-ng attack wasn't detected
- Perform MAC cross-references on non-cloned RAPs to help try to track down the attacker

8 Wireless Security Recommendations

As was demonstrated in this paper, wireless attacks are very easy to perform, but not always simple to detect. The integrity of a wireless network in many cases comes down to the authentication mode and the strength of the network credentials. A wireless network should not be using WEP, or WPA but WPA2. Additionally, if using a Pre-Shared Key (PSK) for authentication is desired, make sure the router is configured to use WPA2 with AES in CCMP mode – not TKIP. If compatibility with older devices needs to be preserved, TKIP may be used as a fallback but it is significantly less secure.

If it all possible, replace PSK with 802.1x authentication. With 802.1x EAP, each wireless user has their own set of credentials to the network. This way, the network admin has much

more fine-grain control over who is allowed access to the network. EAP-PEAP is a generally easy authentication method to roll out even in smaller WLANs when combined with a RADIUS server like FreeRADIUS. If an organization has the resources to implement PKI infrastructure, EAP-TLS is the most secure authentication mechanism. This will require each user to present a signed certificate to authenticate to the network.

9 Conclusion

In this paper is detailed a very common WiFi attack that malicious actors launch to gain access to a network or to trick users into connecting to a malicious network. A new tool was presented that uses known methods to detect these common attacks and alerts users to attacks happening in real-time. The effectiveness of this tool at detecting these attacks was tested by utilizing a few commonly used offensive tools. Alternative network configurations were also recommended.

Overall, attack detection should always take second place to using better security in a network because if an alert that there has been at attack is sent, the network is likely already compromised.

10 Bibliography

802.11 association process explained. (2015, Feb). Cisco Meraki. Retrieved from https://documentation.meraki.com/MR/WiFi_Basics_and_Best_Practices/802.11_Association_process_explained

Koopmann, L. (2017, Sep). Common wifi attacks and how to detect them. Retrieved from https://wtf.horse/2017/09/19/common-wifi-attacks-explained/

Stefanick, G. (2014, Sep). A closer look at wifi security ie (information elements). Hewlett Packard Enterprise Development LP. Retrieved from https://blogs.arubanetworks.com/industries/a-closer-look-at-wifi-security-ie-information-elements/

Uppunda, V. (2017, Sep). Wpa information element. Retrieved from http://www.hitchhikersguidetolearning.com/2017/09/17/wpa-information-element/

11 Appendix I - Glossary

- **AP** - Access Point. The device broadcasting a wireless signal.
- **BSSID** - Basic Service Sets ID. The MAC address of an AP.
- **MAC Address** - Media Access Control. A unique hexadecimal address associated with every wireless devices in the 802.11 space.
- **PSK** - Pre-shared Key. A shared cryptographic secret which has been previously shared between at least two parties over secure channels.
- **RAP** - Rogue Access Point.
- **SSID** - Service Set Identifier. The name of a WiFi network.
- **WPA2** - Wi-Fi Protected Access II. Security protocol commonly used for WiFi, the successor to WPA.

12 Appendix II - Applicable Licenses

Because some of its code is based on the Airoscapy project by Peter Kacherginsky, WatchTower is licensed under the Creative Commons “Attribution-NonCommercial-ShareAlike 4.0

International" (CC BY-NC-SA 4.0) license . Airoscapy is released under this license which requires material built upon it to be released under the same license as the original.

13 Appendix III - Experiment Commands

- `wifite`
- `besside-ng -b B0:39:56:0E:E7:12 -c 6 wlan0mon`
- `aireplay-ng -deauth 1 -a B0:39:56:0E:E7:12 -c 5C:F7:E6:EC:E7:1B wlan0mon`
- `airodump-ng -bssid 04:A1:51:9F:98:BB -write test -channel 6 wlan0mon`