Encrypted WiFi packet injection and circumventing wireless intrusion prevention systems

White Paper

Tim de Waal - 2015

Table of contents

1. Introduction

This whitepaper discusses certain types of attacks that can lead to a completely compromised system even if your system is connected to a WPA2 enabled access point.

The vast majority of WiFi networks today are secured by WPA2. According to the WiGLE database, at the time this document was written, over 71% of wireless networks have WPA2 enabled. For the most part everyone is comfortable using these networks. After all, AES/CCMP as an encryption has held up pretty well. The perception is that they are secure, as long as they typed in a passphrase, even if everyone else in the same coffee shop,convention center,bookshop typed in the exact same passphrase.

There is a public perception that WPA2-PSK is secure enough, and since there are no password policy restrictions on most consumer Access Points, many are lazy and pick dictionary word based, preshared keys/passphrases (or simply hand out the PSK). They then rely on the encryption to protect them against injection, this is a perception that must be corrected.

Wireless networks are an everyday part of our lives. WiFi is the main connectivity for our devices, whether that is our laptop, tablet, TV, internet radio, and in the near future, our fridge. Welcome to the Internet of Things.

Accomplishing an attack that will manipulate victims datastream without them realizing it will require a tool that does packet injection and TKIP/CCMP encryption.

From the current survey of the tool landscape, it seems like the main use of packet injection is to aid in cracking a specific wireless algorithm/key. This author has not found any tools that do arbitrary, correctly formatted and encrypted, packet injection (spoofing) on WPA/WPA2 networks.

The next level of packet injection MUST be able to manipulate traffic over encrypted networks for it to be of use as an offensive security tool.

2. Proposed attack

It is assumed that the audience reading this paper has some knowledge of networking, security exploitation, and a basic understanding of wireless communication.

If the audience in not already aware, the packet injection referred to in this document consists of building a packet from the ground up. Everything can be manipulated and changed, this allows the attacker to spoof the source and destination MAC addresses. This data is present in the 802.11 header which is always transmitted in the clear regardless of the encryption mechanism.

2.1. Encrypted Man-on-the-side Attack

The goal (summary):

- Promiscuously listen to traffic on either an Open or Encrypted network
- Decrypting and being able to filter “interesting” traffic
- Parse and interpret that traffic
- Forge a packet as a response as if the response is coming from the Access Point
- Transmit the forged packet back to the client on behalf of server it was sent to before the actual server is able to respond (encrypted with the PTK).
- The client device acts on the response as if it were sent by the legitimate server via the Access Point.

2.1.1. The Attack Illustrated

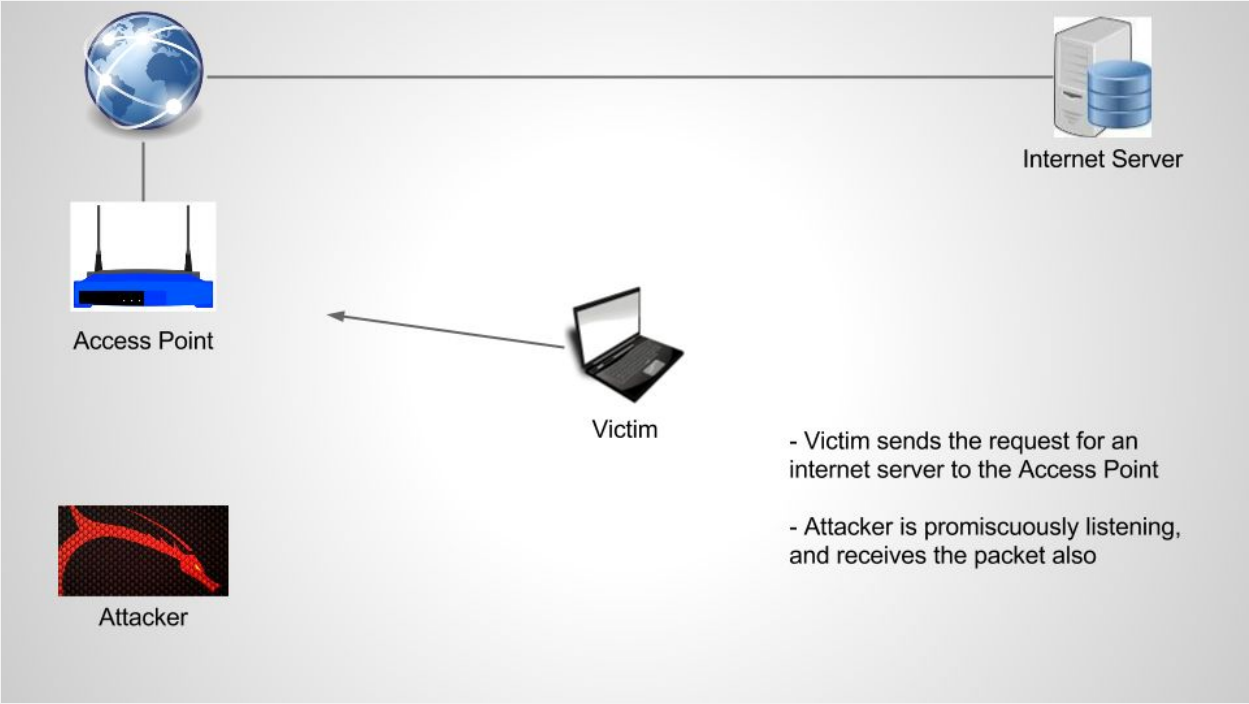

Figure 1: Business as usual, the attacker is passively listening

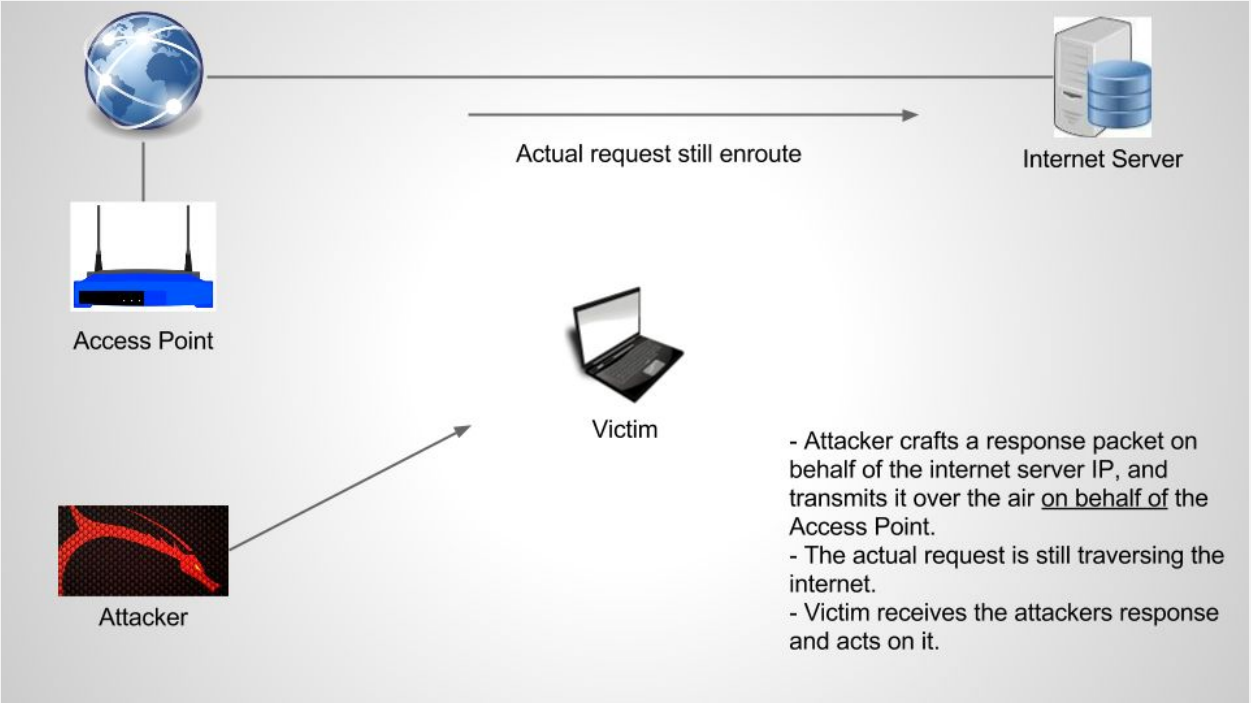

Figure 2: Attacker injects a forged response, Victim accepts it as authentic

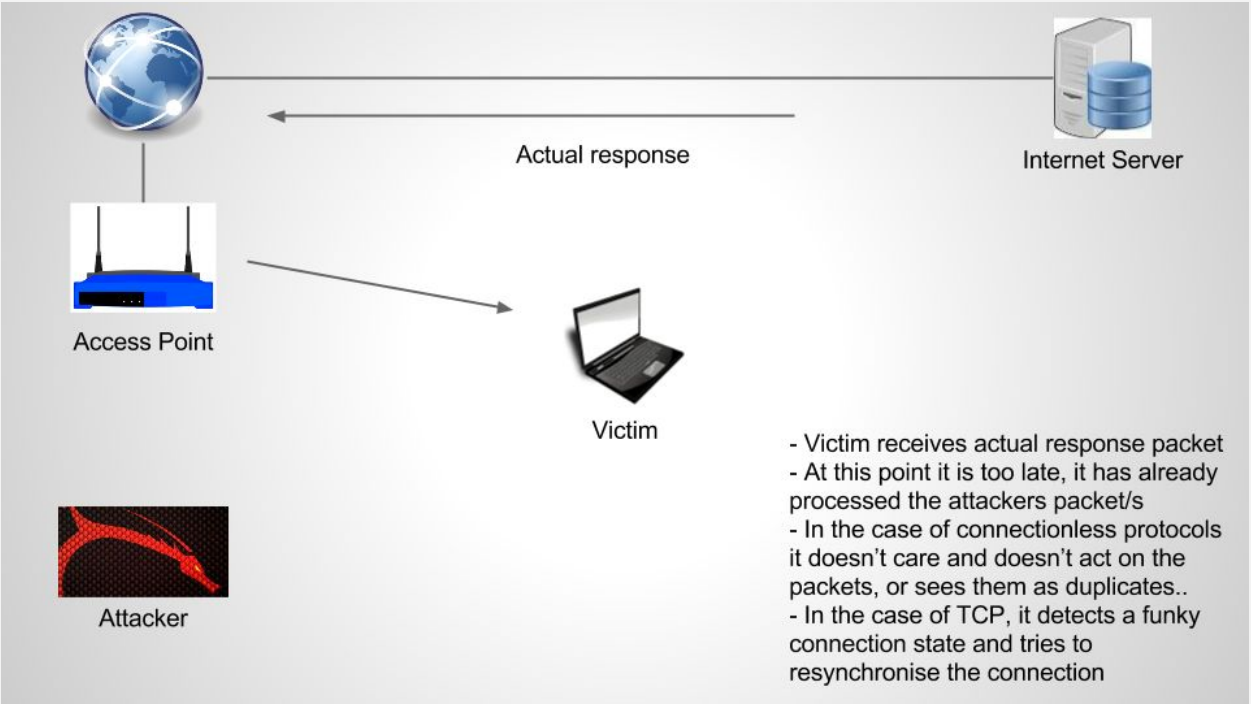

Figure 3: The Victim sees duplicate traffic and drops it

2.2. Usefulness of this attack

The current use of NSA systems such as QUANTUM, prove the usefulness of this type of man-on-the-side attacks, if and only if, it responds faster than the server the client is attempting to contact.

2.2.1. Attack Vectors

The following list is by no means extensive and is not meant to cover every conceivable attack vector. It merely serves to outline some simple attacks that would be possible. It will also not go into in depth details of each scenario since it is assumed that the audience of this paper are well versed in these sorts of attacks.

i. DNS

DNS requests can be intercepted and responded to well before the local DNS server can lookup the record and respond. This could be hit or miss, given some longer TTL records.

ii. ARP

ARP requests can also be easily intercepted and responded to in order to effectively poison the client's arp-cache.

iii. TCP/HTTP

HTTP GET/POST requests are easily intercepted. However, unlike the DNS/ARP responses, TCP sequence numbers need to be incremented and TCP frames correctly manipulated to successfully send a response. However this is fairly trivial with a basic knowledge of TCP. A HTTP 302 Redirect response can easily be sent to the client, making the client go and fetch the redirected asset as opposed to the real asset. Alternatively, rather than a redirect, a fully formed HTTP response can be transmitted. However, this would require more frames to be transmitted.

With just these three simple attacks, a lot can be accomplished. A successful ARP/DNS response can upgrade the man-on-the-side attack to full fledged man-in-the-middle attack. Malicious javascript code can be injected into just about any HTTP page the client decides to load.

2.3. Advantages over Rogue AP, or wireless MITM

When setting up a rogue AP on the network, it requires several things:

1. The rogue AP has to have a unique BSSID but the same SSID as other APs on the network.
2. This means that the AP needs to look like a normal AP to clients
    a. it needs to transmit beacons every 100ms,
    b. it needs to respond to probe requests,
    c. it needs to be able to accept new connections from clients
    d. it requires a separate connection back onto the network (either wired or wireless)
3. It requires the attacker to repeatedly and actively deauthenticate the clients it wishes to attack in order to try and effect a roaming decision.

All of this traffic makes it is much easier to detect with a Wireless Intrusion Protection System. And with the use of triangulation tools, the location of the attacker can be determined fairly quickly.

The setup of the proposed attack is much simpler. It does not require the clients to roam to a different Access point at all. A single reassociation on the part of the client is imperceivable to the user, which is not true for the deauthentication flood needed to force a station to roam. The aforementioned attack also does not require a separate connection back to the network.

2.4. Stealth, circumventing WIPS

Undoubtedly, stealth is always a concern when it comes to any type of attack. The simplicity of this attack allows for the attacker to only transmit a very small amount of wireless traffic. The transmission and destination addresses, the 802.11 sequence numbers, the frame sequence numbers, the checksums are all valid. There is no real way to detect the forged packets until the real response is received by the client as a duplicate frame, at which point it is already too late. The attack happens with a single injected frame, so triangulation becomes very difficult. This completely circumvents any wireless intrusion protection system.

3. Current tools

This section is aimed at providing an overview of several tools currently available and the functionality that they provide.

The following tools already make use of sending packets on behalf of either the AP or client:

3.1. Airpwn

This does the proposed attack beautifully. And it did it back in 2005... For whatever reason it seems that not a lot of people use it. It is this authors perception that it seems to have gotten forgotten amongst the mainstream crowd. The Latest release: 1.4 on 5/27/2009… It therefore does not support TKIP/CCMP decryption.

3.2. LORCON (Loss Of Radio CONnectivity)

Don't let the name fool you, this library is the original wireless packet injection framework. It also seems to have gotten forgotten with few people taking note of its capabilities. The library has bindings for python and ruby support, and has cross platform support to compile under windows via cygwin. Written by Mike Kershaw (Dragorn) it has some awesome functionality. However, it also (currently) does not support TKIP/CCMP decryption.

3.3. airtun-ng

airtun-ng is interesting in that a BSSID and client source MAC address can be specified. A tun interface is created and frames can be received and injected by any additional existing tool. This is great, however it works on a 1-1 mapping of access point to client and only supports open and WEP networks. written by Thomas d'Otreppe.

3.4. wifiarp, wifidns, wifiping

These tools are python based, and will do the proposed attack on open networks. They listen for certain frames and send responses on behalf of the server to the client. This is exactly the functionality that could be very useful to compromise wireless systems, if it were fast enough. This brings the main disadvantage of these tools, they are not always fast enough to beat responses from local servers. They are written in python and use scapy, which is awesome as a proof of concept, but are just too slow to respond within the time we need for a consistent attack. Additionally they do not support WPA/WPA2.

The current set of tools definitely have their purpose and usefulness but none completely meet the requirements needed to execute the attack described in section 2 over an encrypted network.

4. Current challenges and proposed solutions

4.1. Speed

As mentioned previously, the attacker must be able to inject a response packet before the actual server is able to. So the tool must be able to intercept, process and send the response in a timely fashion. To achieve this, the aircrack suite provides a good framework on which to build. Since it is written in C it is well suited for meeting the speed requirements. It also has much of the functionality already implemented in various different pieces of existing code. Especially the decryption provided in airdecap-ng.

4.2. Ease of use

For an attack tool to be used effectively, it needs to be easy to use. It is not desirable to have a powerful tool that is complex and a pain to use. The command line options must be intuitive and simple to understand. It must also be simple to compile and not have too many external dependencies.

4.3. TKIP/CCMP encryption

Although open networks are still fairly prevalent in hot spots at hotels, coffee shops etc,more and more people are encrypting their wireless networks. But usually the PSK is given out freely to paying customers. To truly take wireless packet injection to the next level, we must be able to decrypt and inject encrypted wireless traffic on these networks.

The fact that these networks are encrypted gives a false sense of protection. The tool must be able to keep track of clients that are encrypting data. It must be able to handle multiple clients traffic and inject responses for any number of clients associated to the same access point. It should maintain an internal table of all the 4-Way handshake data necessary to encrypt traffic for any number of stations. Since the 4-Way handshake data is easily captured if the client deauthenticates and reassociates (within range of the attacker), the tool should have an option to deauthenticate existing encrypted stations automatically when it starts up. Going back to the ease of use, the need for a separate tool to do the deauthentication is not desirable.

TKIP and CCMP decryption/encryption should both be supported. Even though it will all be done in software, the available processing power now-a-days should be sufficient to do the decryption, inspection, response forging, and encryption within the time it takes for a server to respond to the captured request.

4.4. 802.11 Sequence collisions

Each 802.11 frame has a sequence number, and predicting the next sequence number could pose a problem given that there can be multiple data streams going to and from the client/access point.

The fairly clever technique to avoid this is to transmit the forged packet on a different QOS level so as not to collide with any other sequence numbers. This was not something this author came up with,

rather this technique was described in the presentation “Advanced WiFi Attacks Using Commodity Hardware” [1]. This however makes the attack considerably easier to detect.

5. Introducing a new tool: Airventriloquist

The functional attack described in this white paper has been implemented and will be released as a patch to the popular aircrack suite.

The details of the use of the tool is beyond the scope of this document.

6. Mitigation Actions

All of the standardly accepted mitigations that apply to open networks also apply here.

- **Switch to WPA2 Enterprise**
- **If you must use wireless, VPN into a trusted network**

However, it should be noted that we are entering an era where the “Internet Of Things” is poised to take over. Many of these new devices on the market are wireless. Given the complexities of setting up a WPA2 enterprise level setup for the average home user, paired with fact that such a level of security is not a requirement outside of the corporate environment, we will be seeing a lot of interesting opportunities for exploitation in this field in the years to come. Given the prevalence of PSK in the current wireless networks, the following option may be the most feasible for most home users.

- **Pick a secure PSK, and keep it a secret.**

7. Conclusion

There is a need for a 802.11 tool that supports packet injection using TKIP/CCMP encryption.

The attacks described in this document should provide another valuable weapon in the already impressive arsenal of available wireless tools.

With the existing framework provided by the aircrack suite, the coding of the attacks outlined in this whitepaper were fairly easy to implement and hopefully also easy to use.

With the ability of the current Airventriloquist tool you too can have a micro QUANTUM attack within the wireless network you are engaging.

8. Citations

[1] “Advanced WiFi Attacks Using Commodity Hardware” by Mathy Vanhoef and Frank Piessens: http://www.slideshare.net/vanhoefm/acsac2014-presentation
[2] “Descending Into Depression and Drink” by Mike Kershaw / Dragorn