International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 12 Issue 2 ǁ February 2024 ǁ PP. 55-60

Exposing Vulnerabilities in Wi-Fi Security: A Proof-of-Concept Framework for OpenWRT

Santosh Kumar Kande
Kandesantosh9@gmail.com

Abstract: *Wireless security is a critical component in safeguarding high-confidentiality and high-integrity networks. This paper presents a proof-of-concept project that extends OpenWRT to expose vulnerabilities in Wi-Fi security. The framework actively intercepts client credentials through probe requests, demonstrating the ease with which an at- tacker can infiltrate insecure networks. A hardware deliverable utilizing a compact router explores exploits in widely used wireless security protocols, including WEP, WPA, WPS, and 802.1X. The research highlights the pressing need for users to comprehend the risks associated with insecure wireless security protocols, urging a shift towards more robust alternatives. The paper concludes with recommendations for router manufacturers and operating system developers to enhance wireless security practices.*
Keywords: *Wireless Systems, Network-level Security and protection, unauthorized access.*

Date of Submission: 03-02-2024 Date of acceptance: 14-02-2024

I. Introduction

Security is critical infrastructure in high-confidentiality or high-integrity networks. Eavesdropping and packet alteration attacks are significantly easier on a wireless medium, and thus, traditional security mechanisms may not be enough due to the inherent insecurity of the channel. The motivation for our project was to demonstrate vulnerabilities in Wi-Fi security. For this project, researchers extended OpenWRT, an open Linux-based firmware that allows taking a high-level perspective while not having to be concerned about the drivers and other inconsequential parts of the project. It also gave researchers the ability to customize the device and its components fully. The deployment of the Wi-Fi attack framework will attempt to intercept client credentials from the wireless interface by actively listening for probe requests from clients. In addition, our framework will be able to compute basic cracking of wireless protocols with known vulnerabilities.

The purpose of the experiment is to demonstrate a proof-of-concept exactly how many vulnerabilities exist in real-world wireless networks and how easy it is for an attacker to infiltrate an insecure network using minimal resources and opportunity. The first step to finding the solution is to identify the problem clearly, and it is important that users understand the liability they are incurring when using insecure wireless security protocols.

Some related Work is the Hak5 Wi-Fi Pineapple [13]. The Pineapple responds to client probe requests and lets it connect with that SSID on an open access point. The goal of this new device to not respond to probe requests but instead to create an access point and then wait for the client to initiate contact. Another goal is to allow for mutable and different encryption types, whereas the Pineapple only makes open access points. The Pineapple wants the user to connect so that a man-in-the-middle attack can be performed, but the new device wants first to capture credentials and other information that may be sensitive to the client without the client initiating the connection.

This paper is organized as follows: section 2 gives an overview of various wireless security protocols in use in modern networks, and the vulnerabilities and exploits associated with those protocols; section 3 outlines the design and implementation of the proof-of-concept hardware deliverable demonstrating the aforementioned exploits; and finally, section 4 explains the results of testing the final product and what was learned from those results.

II. Wireless Security Overview

Securing wireless networks is a dubious task, to the point where some companies simply do not allow wireless access to corporate services at all. Wireless networks are generally secured using one of two security standards: either IEEE 802.11– 1999 (WEP) or IEEE 801.11i-2004 (WPA and WPA2), the latter optionally combined with Wi-Fi Protected Security (WPS) or IEEE 802.1X-2004.

2.1 WEP

Wired Equivalent Privacy, often acronymed as WEP, was the first security algorithm developed for Wi-Fi and was initially specified as part of the original IEEE 802.11 standard as released in 2003 [1]. As the name implies, the protocol was designed primarily to provide privacy and confidentiality for wireless communications, i.e., ensure that other clients cannot listen in on traffic being sent to the access point. WEP even has an 'Open' mode, where traffic is encrypted, but clients do not have to authenticate to the access point. In fact, the only method of authentication or access the control provided by WEP is pre-shared key authentication, which relies on using the key in an RC4 cipher when encrypting challenge text from the access point.

WEP is thoroughly broken. As an emphasis of this fact, using 'Open' mode, where no authentication is performed, is actually more secure than using a pre-shared key. (The reason behind this has to do with deriving the keystream by capturing the initial challenge.) Among the many issues with the protocol are the use of RC4, a stream cipher that requires using a new key every time, and weak 23-bit initialization vectors that are open to repetition. These vulnerabilities resulted in the Fluher, Mantin, and Shamir attack [5], which, including improvements over the years, allows a 95% chance recovery of a 104-bit WEP key using only 85,000 captured network packets [11]. This attack can be achieved on consumer hardware in just a few minutes.

2.2 WPA

WEP has since been deprecated and superseded by Wi-Fi Protected Access. The Wi-Fi Alliance released the first version of WPA in the IEEE 802.11i draft standard as a stop-gap measure for replacing WEP, at least until the IEEE 802.11 working group could flesh out the final version of the protocol. That final version was released as IEEE 802.11i-2004 and is commonly referred to as WPA2 [2]. The first version of WPA used the Temporal Key Integrity Protocol (TKIP). Although TKIP uses a 128-bit key and discourages attacks with a new key mixing function and message integrity checks, it uses some of the same techniques as WEP encryption, e.g., using CRC32 for checksums, and thus is vulnerable in many of the same ways. Halvorsen, et. al [6] Demonstrated being able to recover up to 596 bytes of the TKIP keystream in under twenty minutes. Eventually, this was fixed in WPA2 with the advent of the Counter Mode CBC- MAC Protocol (CCMP or AES-CCMP), which is considered secure as of the writing of this paper. CCMP works by using CBC-MAC to encrypt the data and compute the integrity tag, and then encrypting that tag in CTR mode, thus achieving authenticated encryption. Any attack on CCMP would imply an attack on AES itself.

WPA also, unlike WEP, has more support dedicated to access control. In In addition to using a pre-shared key, two other methods of client authentication may be used: Wi-Fi Protected Setup (WPS) and IEEE 802.1X.

2.3 WPS

WPS was created by the Wi-Fi Alliance in 2006. It is not an IEEE standard, but nonetheless, it is supported on many consumer routers. It was created to allow non-technology-savvy users to authenticate their wireless devices more easily. The protocol works over EAP and involves having the user either push a button on the router or enter a preset PIN on the device. The enrollee (client) and the registrar (access point) negotiate configuration data and then reconnect using the new configuration. The WPS protocol has since been broken due to a timing attack on the PIN authentication, which allows a brute force attack to be completed in under four hours [12]. To make things worse, some routers are misleading, and even if a user disables WPS in the user interface, it remains enabled, and the router will still negotiate with clients. Fortunately, some router manufacturers have released firmware updates that fix this issue, while others have implemented rate-limiting on the WPS mechanism to stop brute-force attacks.

2.4 802.1X

Contrary to WPS, 802.1X is not an authentication protocol but a wrapping of an existing authentication framework. IEEE 802.1X-2007 is a method of encapsulating the Extensible Authentication Protocol (or EAP, as defined in RFC 3748 [3]) in IEEE 802 packets [1]. The protocol is sometimes referred to as EAPOL, or EAP over LAN, and was originally designed for IEEE 802.3 Ethernet, but was extended to wireless in 2004. The 802.1X protocol itself is not of much interest for security research, but the underlying EAP is. As mentioned, EAP is an authentication framework that can be used with many different authentication methods, some more secure than others.

Common authentication methods available on EAP are EAP-MD5, EAP- PWD, EAP-MSCHAP, EAP-TLS, EAP-TTLS, and PEAP. There are many others, but only these are both widely supported natively by clients and are commonly used by network administrators.

Starting at the bottom, EAP-MD5 involves a pre-shared key that, as the name implies, is combined in MD5 to authenticate the client [3]. Other than the various vulnerabilities inherited from using MD5, this protocol does not have any server authentication and thus is blatantly vulnerable to man-in-the- middle attacks using a fake access point. EAP-MD5 was officially deprecated in Windows Vista. EAP-PWD improves upon EAP-MD5 by using a secure hash function, specifically HMAC-SHA-256, and authenticating both the server and the client [7]. While the key exchange protocol is resistant to attack, this protocol may still suffer from users setting low-entropy passwords or passwords that can be guessed using a dictionary attack.

A more commonly used protocol is EAP-MSCHAP (either version 1 as defined in RFC 2433 [15] or version 2 as defined in RFC 2759 [16]). It involves using Microsoft's customized version of the Challenge-Handshake Authentication Protocol, and uses a three-way handshake using SHA1, MD4, and DES (together) to authenticate both the client and the server. As the reader may be able to infer the protocol is needlessly complicated and, as demonstrated by Marlinspike in 2012 [10], can be broken by cracking a single DES key and then cracking the resulting MD4 hash. Since both tasks are trivial on modern hardware, MS-CHAP is considered broken.

Finally, there are EAP-TLS, EAP-TTLS, and PEAP, which, with respect to the protocol, are considered secure. The first uses TLS to authenticate both the server and the client. X.509 certificates are issued to clients and servers, all signed by a common certificate authority (CA). The latter two are very similar and do not require the client to have its own X.509 certificate. Instead, they establish the TLS connection as a tunnel and then let the client authenticate using further protocols that are tunneled inside the TLS stream.

PEAP specifically, as developed my Microsoft, uses EAP once again inside the encrypted TLS tunnel. In other words, it is a second iteration of EAP inside a tunnel established via the first iteration of EAP. The catch is that each version of PEAP (there are two) only supports a single authentication mechanism for the inner iteration of EAP. PEAPv0 uses EAP-MSCHAPv2, and PEAPv1 uses EAP-GTC, an alternative to MSCHAPv2 developed by Cisco. With PEAP, since the inner an authentication protocol is run inside a TLS tunnel, it remains secure even despite the vulnerabilities of MS-CHAP [4].

However, while all of these protocols are indeed the most secure out of all the EAP options, they still inherit vulnerabilities from the X.509 trust model. The server's certificate must be signed by a trusted CA and users must be instructed not to trust unsafe certificates. Unfortunately, this can be nearly impossible since: certificates for use network-wide in 802.1X systems are extraordinarily expensive and are not affordable to smaller entities; and even modern operating systems, like Windows 8 and OS X, provide trivial and almost entirely ignorable warnings when a network provides an untrusted server certificate. The only method of protection is to go multiple user interface levels deep into advanced network settings and toggle a flag disallowing unsafe certificates. As of the writing of this paper, the wireless network at the Stevens Institute of Technology does not use a trusted certificate and the Information The technology department explicitly instructs users to ignore warnings and connect anyway.

III. Design and Execution

3.1 Threat Model

Overall, there are numerous vulnerabilities in all of the available security protocols for Wi-Fi. The ones more readily exploitable, as listed above, are:

- Sniff traffic for WEP-secured network and brute-force the keystream.
- Perform a man-in-the-middle attack on a WPA-TKIP-secured network.
- Brute-force the PIN for a WPS-secured network.
- Use dictionaries or brute-force to guess a weak WPA-PSK or EAP-PWD password.
- Impersonate an Access Point of an EAP-MD5-secured network and crack the client credentials.
- Crack the DES key and the resulting MD4 hash of a key exchange on an EAP-MSCHAP-secured network.
- Provide a fake server TLS certificate on an EAP-TLS- or EAP-TTLS-secured network and social engineer the user into accepting it.
- Provide a fake server TLS certificate on a PEAP-secured network and then crack the inner MS-CHAPv2 exchange as described before.

Many of these attacks allow complete security breaks in the entire wireless network. Looking at the system from the perspective of the attacker, there are a number of assets that might be of interest. Specifically, they are:

a. If the attacker can perform a man-in-the-middle attack between the access point and the RADIUS server used to authenticate, then a new attack on PEAP becomes possible, but this paper does not consider that attack since the primary threat model involves an outside attacker attempting to gain access to the network.

3.2 Network access

The network may contain resources that only authenticated clients can be allowed to access. If the wireless network provides access to the entire LAN, including business-critical resources; this can pose a high-level risk to the confidentiality and integrity of those resources, and a medium-level risk to availability depending on the services running. On the other end of the spectrum, even if the wireless network is only provided for personal use, it could still pose a low-level risk for integrity, specifically impersonation.

3.3 Client credentials

Some wireless attacks allow cracking of the client's original credentials. This gives the attacker persistent access to the network under the client's identity, thus exaggerating any risks aforementioned concerning network access.

3.4 Network traffic

In limited cases where the attacker gains the client credentials, and in cases where the attacker performs a man-in-the-middle attack on the client, the attacker can sniff and later data sent to and from the actual network. This poses a high-level risk to confidentiality and integrity of any mission-critical information is communicated over that network.

As a further note, some assumptions are made about the system when making these attacks. Specifically, the attacker must be in an opportune location to launch these attacks. A WEP key cannot be cracked if you are not in range of the network in order to capture traffic. As a result, it is a requirement that this device be portable and unnoticeable, thus facilitating easy, persistent planting of the device in a suitable ground zero. In addition, if EAP-TLS, EAP-TTLS, or PEAP protect the network, the assumption needs to be made that either a trusted authority does not sign the server certificate or that users are either not knowledgeable enough or not caring enough to properly validate the certificate every time they connect to the network. In most situations, these two assumptions can be made trivially.

3.5 Hardware Design

In order to facilitate easy placement, a tiny form factor router was chosen for this experiment: the TP-Link TL-WR703N [9]. The device measures 5.7 cm by 5.7 cm by 1.8 cm (slightly larger with a low-profile USB storage device plugged in) and has an Atheros AR7240 CPU at 400 MHz and an Atheros AR9331 chipset, with integrated 802.11b/g/n wireless.

One of the issues with the device was power. Power is supplied via Micro USB, but a power source is needed for the router to remain operational. For an attacker, plugging into an outlet may not be an option if the router is to be hidden, and thus a portable battery must be connected to the router when it is planted. With Wi-Fi at 18 dB mW, average current is 100 mA at approximately 5 V, giving a power consumption of 0.5 W. Under normal conditions, a small 3200 mA h phone battery could power the router for 32 h, which is more than enough to exploit most network vulnerabilities or capture at least one set of client credentials. With an even more powerful battery (on the current market, going as high as 25.000 A h), the device becomes larger but can last more than ten days.

OpenWrt 12 'Attitude Adjustment' was used as the base system, with various additional packages installed, such as hostpad for launching multiple wireless interfaces simultaneously, scapy for scanning wireless networks [8], and freeradius2 for simulating access to a RADIUS server in 802.1X-secured networks.

3.6 Wireless Exploits

Some of the main purposes of the framework are to provide wireless reconnaissance, to locate a wireless network, and to collect information about its configuration and associated clients. The 'scapy' python library, an interactive packet manipulation program, was used for the reconnaissance framework since it provides the ability to forge or decode packets of many protocols. In order to use Scapy, the wireless network interface, was set to monitor mode to capture and listen to all traffic transmissions from the wireless access point. Unlike promiscuous mode, which is also used for packet sniffing, monitor mode allows packets to be captured without having to associate with an access point or ad hoc network first. Using 'scary', probe requests and response frames were captured to determine whether clients were actively attempting to seek any, or a particular, access point.

For discovered 802.11 PEAP/WPA networks, the device then employs a RADIUS impersonation attack. The attack, as described earlier, exploits the lack of certificate validation by setting up a fake access point with a forged certificate. In addition, the device runs a patched RADIUS server that allows all username and password combinations to be authenticated while simultaneously logging the credential exchange. Combined, the client will attempt to connect and subsequently expose the client's credentials in the resulting MS-CHAP exchange.

IV. Results and Testing

The device was programmed to receive client probe requests and set up to six access points on demand. Access points are configured to match the expected encryption settings of the actual access point. Afterward, the device pools for clients attached to the access point and removes duplicate access points that have the same SSID but are operating on that interface. Furthermore, old interfaces are pruned based on the time they were created due to the upper limit of the number of interfaces supported by the wireless driver.

There are some limitations to this setup. One is that it is hard to delete interfaces, add interfaces, and modify them without restarting all of the interfaces. This causes issues because any existing client traffic is interrupted upon adding, modifying, or deleting an interface. Currently, the research team is unaware of any method of hot-swapping wireless interfaces with the currently available drivers.

The research team also took some of the results from Spoofing Enterprise Routers, specifically FreeRadius-we patch files that allow for logging RADIUS requests and responses in order to decrypt passwords and log them as passed in plaintext. The patches were not made for OpenWRT thus, patches had to be applied to the version of FreeRadius that works with OpenWRT. To my knowledge, this is the first instance of a self-contained FreeRadius-we router in OpenWRT.
Otherwise, the device was able to successfully capture 802.1X client credentials sent over PEAP-MSCHAPv2, which is the most popular enterprise authentication protocol.

V. Conclusion

As demonstrated by the device, wireless security is in a broken state. Most networks, both personal and enterprise, are open to exploitation using various methods depending on the authentication protocol in use.

The moral of the story is that there are only two security protocols for Wi-Fi that is effective at mitigating threats: using WPA-CCMP-PSK with a strong, non-guessable key and using EAP-TLS, EAP-TTLS, or PEAP (over 802.1X) with a trusted certificate authority and clients that are configured always to reject untrusted certificates. Literally any other wireless environment can be broken by consumer-grade hardware and is entirely insecure.

The only solution to this problem is for router manufacturers to find new ways to encourage users to utilize secure wireless technology. For example, completely remove WEP, WPA-TKIP, and WPS from routers. By only allowing WPA2-CCMP, using either a PSK or the secure 802.1X, the protocol is secure, and any vulnerabilities can only be introduced by user bad practices. For WPA2 with a PSK or with PEAPv0-MSCHAPv2, this paper does not address methods of encouraging users to create secure passwords and follow proper password management policies, as that is a much larger scope problem that needs to be addressed by a dedicated research. For EAP-TLS, however, operating systems developers should make rejection of unsafe certificates the default settings and create more prominent warnings when a user is prompted to inspect and accept an unsafe certificate. A similar example of this problem and solution is phishing warning toolbars in browsers, which were proven largely ineffective and replaced with more serious and prominent, full-page warnings [14].

The onus is now on router and operating systems developers to disable insecure protocols in their firmware and software, thus enforcing user best practices and improving the general state of wireless security.

References

[1]. "IEEE Standard for Local and metropolitan area networks - Port-Based Network Access Control," in IEEE Std 802.1X-2004 (Revision of IEEE Std 802.1X-2001) , vol., no., pp.1-175, 13 Dec. 2004, doi: 10.1109/IEEESTD.2004.96095. keywords: {IEEE Standards;Patents;Local area networks;Access control;Licenses;authentication;authorization;controlled port;local area networks;metropolitan area networks;port access control;uncontrolled port},

[2]. "ISO/IEC International Standard - Information Technology Telecommunications and Information Exchange Between Systems Local and Metropolitan Area Networks Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Medium Access Control (MAC) Security Enhancements," in ISO/IEC 8802-11, Second edition: 2005/Amendment 6 2006: IEEE STD 802.11i-2004 (Amendment to IEEE Std 802.11-1999) , vol., no., pp.c1-178, 23 July 2004, doi: 10.1109/IEEESTD.2004.311922.

[3]. Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., Levkowetz, H., et al.: Extensible authentication protocol (eap). Tech. rep., RFC 3748, June (2004)

[4]. Dunstan, P.: Attacking and securing peap, http://www.defenceindepth.net/2010/05/attacking-and-securing-peap.html

[5]. Fluhrer, S.R., Mantin, I., Shamir, A.: Weaknesses in the key scheduling algorithm of rc4. In: Revised Papers from the 8th Annual International Workshop on Selected Areas in Cryptography. pp. 1–24. SAC '01, Springer-Verlag, London, UK, UK (2001), http://dl.acm.org/citation.cfm?id=646557.694759

[6]. Halvorsen, F., Haugen, O., Eian, M., Mjølsnes, S.: An improved attack on tkip. In: Jøsang, A., Maseng, T., Knapskog, S. (eds.) Identity and Privacy in the Internet Age, Lecture Notes in Computer Science, vol. 5838, pp. 120–132. Springer Berlin Heidelberg (2009). https://doi.org/10.1007/978-3-642-04766- 4_9, http://dx.doi.org/10.1007/978-3-642-04766-4_9

[7]. Harkins, D., Zorn, G.: Extensible authentication protocol (eap) authentication us- ing only a password. Tech. rep., RFC 5931, August (2010)

[8]. Keeble, E.: Passive wifi tracking, http://edwardkeeble.com/2014/02/passive-wifi-tracking/

[9]. mandrawes: Tp-link tl-wr703n, http://wiki.openwrt.org/toh/tp-link/tl- wr703n?rev=1417238011

[10]. Marlinspike, M.: Divide and conquer: Cracking ms-chapv2 with a 100% success rate, https://www.cloudcracker.com/blog/2012/07/29/cracking-ms-chap-v2/

[11]. Tews, E., Weinmann, R.P., Pyshkin, A.: Breaking 104 bit wep in less than 60 sec- onds. In: Proceedings of the 8th International Conference on Information Security Applications. pp. 188–202. WISA'07, Springer-Verlag, Berlin, Heidelberg (2007), http://dl.acm.org/citation.cfm?id=1784964.1784983

[12]. Viehböck, S.: Brute forcing wi-fi protected setup. Wi-Fi Protected Setup (2011)

[13]. Wood, R.: Wifi pineapple / jasager, https://forums.hak5.org/index.php?/forum/64-wifi-pineapple-jasager/

[14]. Wu, M., Miller, R.C., Garfinkel, S.L.: Do security toolbars actually prevent phishing attacks? In: Proceedings of the SIGCHI Conference on Human Fac- tors in Computing Systems. pp. 601–610. CHI '06, ACM, New York, NY, USA (2006). https://doi.org/10.1145/1124772.1124863

[15]. Zorn, G., Cobb, S.: Microsoft ppp chap extensions. Tech. rep., RFC 2433, October (1998)

[16]. Zorn, G.: Microsoft ppp chap extensions, version 2 (2000)