Intrusion Detection in 802.11 Networks: Empirical Evaluation of Threats and a Public Dataset

Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Stefanos Gritzalis

***Abstract*—WiFi has become the de facto wireless technology for achieving short to medium-range device connectivity. While early attempts to secure this technology have been proved inadequate in several respects, the current, more robust, security amendments will inevitably get outperformed in the future too. In any case, several security vulnerabilities have been spotted in virtually any version of the protocol rendering the integration of external protection mechanisms a necessity. In this context, the contribution of this paper is multi-fold. First, it gathers, categorizes, thoroughly evaluates the most popular attacks on 802.11, and analyzes their signatures. Second, it offers a publicly available dataset containing a rich blend of normal and attack traffic against 802.11 networks. A quite extensive first-hand evaluation of this dataset using several machine learning algorithms and data features is also provided. Given that to the best of our knowledge the literature lacks such a rich and well-tailored dataset, it is anticipated that the results of the work at hand will offer a solid basis for intrusion detection in the current as well as next generation wireless networks.**

***Index Terms*—WiFi, 802.11, Security, Attacks, Intrusion Detection, Dataset.**

I. Introduction

WIRELESS networks have prevailed in the last few years, managing to unsettle the dominance of the wired ones [1]. The 802.11 family of networks commonly known as WiFi are today's most popular choice for local area connectivity, as they provide low cost, and effortless wireless connectedness. Such networks can be found in small office and home (SOHO) settings, enterprise environments or even serve in ad-hoc situations where users simply wish to establish fast and reliable connectivity to exchange data. With the mushrooming of these networks and the proliferation of handheld devices, the vision of "always on, always connected" has become a reality. Anticipated 5G deployments are expected to knit their air interfaces and spectrum together with LTE and WiFi to offer a harmonious user experience and global high-rate coverage. However, the flexibility and mobility that WiFi networks offer, comes with the price of questionable security.

Since the first version of the 802.11 standard [2], dedicated security mechanisms have been incorporated to guarantee safe communication of all the peers in the Wireless Local Area Network (WLAN). Wired Equivalent Protection (WEP) was quickly found to be vulnerable not only to a great number of availability attacks but more importantly to attacks that threat the secrecy of its key, jeopardising the confidentiality of the entire communication. Posterior efforts such as WiFi Protected Access (WPA) and WPA2 proved to be more robust as far as confidentiality is concerned. However, with the increasing computational power and the instalment of low-cost cluster computing this will be soon inaccurate. Naturally, these mechanisms are anticipated to render themselves vulnerable even to brute force attacks [3]. On the other hand, cloud-based systems like CloudCracker [4] can test 300 million possible WPA passwords in just 20 minutes.

In any case, WPA/WPA2 share almost the same vulnerabilities as the early WEP versions as far as availability is concerned. Even the newest amendment, 802.11w [5], which concentrates in patching availability related shortcomings (leading to Deauthentication, Disassociation and Authentication Request attacks for example) has been proved impotent to tackle the entirety of documented DoS attacks [6].

Furthermore, easy to use penetration testing tools, which are able to automate attacks against 802.11 networks, have been developed, and are easily accessible [7]. Such tools are convenient enough to enable even low-skilled opportunists to cause disruption to the normal service of a wireless network in several ways.

A considerable mass of works provides recommendations on how to remedy existing vulnerabilities in order to enhance the security of 802.11 networks [8], [9]. Most of these enhancements could be applied as a firmware update of the Access Points (AP) of the deployed networks, but such strategy usually acts against backward compatibility and may seem impractical to the eyes of inexperienced users. External mechanisms of protection of a wireless network have emerged and quickly became popular. In this context, Intrusion Detection Systems (IDS) such as [10] provide solid means of identifying and possibly responding to a threat in a timely fashion. Such systems recognize intrusions based on predetermined signatures of known attacks. However, Machine Learning (ML) based wireless IDSs are always within the scope of researchers since they do not require pre-compiled (static) signatures of attacks like the misuse detection based ones [11] rather deduce them automatically through the utilization of some classification or clustering algorithm.

Our contribution: The contribution of the work at hand lies in several axes. First off, it gathers and describes the philosophy of most well-known 802.11 attacks. We argue that most existing surveys in this area are either outdated [12] or fail to provide a holistic view of the problem, since they usually focus on a specific subset of the standard [13].

C. Kolias, G. Kambourakis and S. Gritzalis are with the Laboratory of Information and Communication Systems Security, Department of Information and Communication Systems Engineering, University of the Aegean, Samos, GR-83200, Greece e-mail: (kkolias, gkamb, sgritz)@aegean.gr

A. Stavrou is with Computer Science Department, Center for Secure Information Systems, George Mason University, Fairfax, VA 22030, USA e-mail: astavrou@gmu.edu

A categorization of the attacks based on different criteria is also provided. Secondly, in the context of this work, attacks described so far only in a theoretic level were implemented, and their practicality was measured (alongside with numerous other popular attacks) through experiments to conclude to an estimation of the possible threat they pose. Thirdly, it analyzes traces of both 802.11 normal and attack traffic to highlight possible attack patterns. However, the major pillar of contribution of this work is the Aegean WiFi Intrusion Dataset (AWID), a publicly available collection of sets of data in easily distributed format, which contain real traces of both normal and intrusive 802.11 traffic. Opposed to alternatives like [14] our dataset is oriented towards intrusion detection and more specifically intrusion detection in wireless networks. The traces contained in AWID are not artificial but are extracted from real utilization of a dedicated WEP protected 802.11 network. To the best of our knowledge this is the first publicly available dataset of this kind. We argue that the well-known KDD'99 [15] or similar sets crafted for wired environments will not lead to the creation of optimized algorithms targeting 802.11 environments as the two realms possess vitally diverse characteristics. On the contrary, the AWID dataset may prove a valuable tool for research even on different wireless technologies (e.g. WiMax [16], UMTS [17], LTE [18]) or alternative 802.11 settings (e.g., mesh mode [19], vehicular networks [20]) since some of the respective attacks are based on resembling principles. This work concludes with the presentation of comparative results of numerous classification algorithms applied upon the AWID.

We argue that our contributions will (a) assist researchers on getting accustomed with the major vulnerabilities and existing attacks of 802.11 networks, (b) inform them about the practical impact these attacks are expected to inflict under real-life conditions, and finally, (c) provide a problem-directed tool for ML-based intrusion detection on wireless networks.

The remainder of this paper can be broken down in the following parts: The upcoming section briefly inspects the 802.11 architecture and its security mechanisms. Section 3 enumerates and describes major attacks against 802.11 standard. Next section draws conclusions regarding the feasibility of such attacks extracted from experimentation. Section 5 details attack signatures based on theoretic and practical analysis of intrusive and normal traffic. Section 6 outlines the most important aspects of the AWID dataset. The evaluation of well-known classifiers is conducted in section 7. The final discussion along with conclusions and possible future directions is given in the last section.

II. 802.11 Architecture

In this section a brief description of the entities defined in the standard, their supported organisational modes, their possible ways of communication, along with the available security mechanisms is provided. Note that all terms mentioned here, are defined in the respective standard [2].

A. Network Architecture

The IEEE 802.11 family networks can be organized in either Infrastructure or Ad-Hoc mode. In the first paradigm the basic organizational unit is a special piece of hardware, namely the Access Point (AP) to which the stations (STA) -also referred to as clients (these terms will be used interchangeably)- connect and through which the generated packets are transferred. On the contrary, in Ad-Hoc mode the STAs communicate with each other within their range directly, without the requirement for an AP. In this organizational paradigm the nodes of the network also play the role of the router.

Generally, security and lack of infrastructure are two opposing forces in WLAN. By definition, Ad-Hoc WiFi networks are less secure than the Infrastructure-based ones but in such scenarios security is typically of secondary concern. Admittedly, these two areas of study have diverse vulnerabilities and their traffic behavior is significantly dissimilar even under normal conditions. At this point it should be made clear that all experiments in this work, along with the discussed attacks and the resulting dataset, refer to Infrastructure mode networks.

B. Frame Types

802.11 defines three different types of frames, namely management, control, and data. Each of them has different length and fields and fulfils a different purpose.

1) Management Frames: 802.11 management frames allow STAs to establish communication with an AP and preserve connectivity with it. A management frame's structure varies depending on its purpose. Such frames can have one of the following subtypes: (a) Authentication, (b) Deauthentication, (c) Association Request, (d) Association Response, (e) Reassociation Request, (f) Reassociation Response, (g) Disassociation, (h) Beacon, (j) Probe Request, (k) Probe Response. For example, Deauthentication is the type of frame sent from the AP to a STA when the former decides to terminate all communication with that client. Alternatively, Deauthentication frames can be sent from a client to the AP simply to notify about its intention to drop communication. In both cases, Deauthentication frames are not requests and must always be accepted and acted upon. Another example of management frames are the Beacon ones. These are broadcasted periodically by an AP to announce its presence and advertise its capabilities. On the other hand, frames of the Probe Request type are broadcasted by an unauthenticated client in search for a specific AP. It is possible that such messages do not specify an AP so that the STA can immediately obtain information about all APs within its range.

2) Control Frames: 802.11 control frames coordinate access to the wireless medium and play a role in the delivery of data frames from a STA to the AP and vice-versa. A control frame can have one of the following types: (a) Request to Send, (b) Clear to Send, (c) Acknowledgement, (d) Power Save (PS) Poll. For example, a Request to Send frame (RTS) is the first message of the 802.11 RTS/CTS handshake mechanism. This mechanism is optional but when applied it reduces frame collisions caused by the hidden terminal phenomenon. If that mechanism is active, the STA is required to send RTS frame to request permission to occupy the channel before transmitting an actual data frame.

3) Data Frames: Data frames are used to transmit the actual information produced from other layers. There are different types of data frames based on whether the data is sent on a contention based service, whether they carry additional information and whether they have Quality of Service (QoS) enhancements. For example, a frame of the Data type is the basic kind used for sending and receiving data. These frames are transmitted during the contention-based period. On the contrary, frames of Null Data type carry no payload. They are transmitted exclusively from a STA towards the AP to edify a change in its sleep state. This is accomplished simply by altering the value of the respective power management bit.

C. Frame Structure

All data frames have the same structure which consists of a header, the frame body, and a Frame Check Sequence (FCS). Any data placed on the frame body is usually encrypted. The frame body is the only variable length field and can take up any value from 0 to 2,312 bytes. The FCS has length of 4 bytes. It is based on CRC-32 algorithm and it is applied to bytes of both the header and the body. The header is the most complicated of the fields. It is 30 bytes long and in turn it is comprised of 7 fields.

Management frames have similar structure to Data ones with the exception that their body may only be comprised by fixed or variable length tagged parameters. Control frames do not have a body and their header has smaller size than the rest of the frame types.

The highly dynamic nature of 802.11 frames brings to surface the requirement for their representation as static vectors of attributes within a given dataset. In this respect, section VI-D gives details on the adopted record schema on the AWID dataset.

D. WEP Security

Wired Equivalent Privacy (WEP) was the sole security mechanism in the first version of the 802.11 protocol introduced in 1999. As the name implies its main goal was to provide a confidentiality level comparable to that offered in the wired world. Nonetheless, as proved in practice these goals were not met and this protocol was found susceptible to a number of different attacks, including these that allow the efficient calculation of its secret key. Naturally, with the introduction of 802.11i, WEP became officially deprecated. Still, a non negligible mass of 802.11 networks utilize WEP as their sole protection mechanism.

1) Authorization: WEP supports two methods of authentication namely, open system and shared key. In the first case, the client does not need to provide any credentials for connecting to the AP. The authentication is completed after the exchange of only two messages. Frequently, in such scenarios the network is protected through means of whitelisting specific MAC addresses.

On the other hand, in the shared key authentication, a process that completes with the exchange of four messages takes place before a client can enter the network. More specifically, (a) the client sends an Authentication Request message which contains the MAC address of the client and the MAC address of the AP, (b) the AP responds with a challenge message which contains a 128 bits random number, (c) the client sends a response message which contains the random number encrypted with the WEP shared key. The AP then decrypts the previous message using its shared key. If the number contained in the decrypted message matches the random number previously send, then the AP considers that the client is in possession of the shared key. As a final step the AP responds with (d) an Authentication Response message containing the outcome of the authentication process.

It is clear that the authentication process described above is strictly unidirectional meaning the AP can authenticate the client but not vice-versa.

2) Traffic Encryption: WEP depends in the RC4 algorithm for confidentiality, while the CRC-32 mechanism is employed for message integrity. Confidentiality in WEP relies on a static key also known as root key. WEP supports two different key sizes and as a result two versions exist, namely WEP-40 and WEP-104.

WEP-40 supports key sizes of 40 bits. This key is never used for direct packet encryption, but it is the basis (seed) for the generation of a session key. Only data frames are protected while management and control ones remain unguarded. Every time a packet is to be encrypted the following sequence of actions takes place: A 24-bit long Initialization Vector (IV) is generated usually in a sequential way, although, a detailed methodology is not specified by the standard. Next, the root key is concatenated with the IV forming the "per packet key". Note that even thought the root key remains the same, the IV varies in each encryption attempt. For this reason, the resulting "per packet key" is also different for each packet. This key (which itself is merely a 64 bit sequence) seeds the RC4 algorithm producing a key sequence which is known as keystream. As a final step, the keystream is XORed with the concatenation of the plaintext of the packet and its CRC-32 value resulting to the ciphertext of the specific packet.

The encryption process on WEP-104 is analogous except for key size which in this case is 104 bits.

E. WPA

WiFi Protected Access (WPA) is a security technology that was introduced in 802.11x amendment in order to alleviate the weaknesses of the original security mechanism. Since WEP was found vulnerable to attacks that could be launched by attackers even with moderate level of skills many network administrators started deploying third-party security solutions including 802.1X and Virtual Private Networks (VPN) in order to increase the level of security. The lack of native wireless security triggered the development of 802.11i by IEEE and the WiFi alliance. WPA was treated as a transitional step since the more robust 802.11i (frequently referred as WPA2) security sub-protocol was still under development. Actually, WPA is a subset of 802.11i but it maintains forward compatibility with it.

The cornerstone of WPA is the provision of stronger encryption mechanisms, such as Temporal Key Integrity Protocol

(TKIP) or the Advanced Encryption Standard (AES) which is employed as an alternative. At the same time, WPA effectively addresses critical security issues, including mutual authentication via the utilization of 802.1X framework and the Extensible Authentication Protocol (EAP), more appropriate IV lengths, a stronger integrity check mechanism (namely Michael), a secure rekeying function, and others.

WPA depends on central authentication servers such as RADIUS for user authentication, access control and management. While this practice is typically adopted in enterprise environments for home users, a variation of WPA, namely WPA Pre-Shared Key (WPA-PSK) has been developed. In essence, WPA-PSK is a simplified version of WPA which is based on the use of a passphrase as a pre-shared secret key among the users much like in the case of WEP.

F. WPA2

IEEE 802.11i, commonly known as WPA2, was an amendment to the original IEEE 802.11 standard aiming to increase the security of the protocol. The final draft was ratified on June 24, 2004 and it was finally incorporated into the IEEE 802.11-2007 standard. Note that the currently latest version of the standard, namely 802.11ac [21] (which was finalized and approved in January 2014) also adopts 802.11i as its primary security protocol. Although the beamforming feature will possibly generate the need for redisign of the penetration testing equipment, it is safe to assume that most of the vulnerabilities discovered for 802.11i will apply also in the newest version of the protocol. Below, we will provide details on how key generation process is conducted in 802.11i and what processes are provisioned for traffic confidentiality and integrity.

1) Key Construction: In WPA2 all keys are derived from a single key which is placed in the highest level of the hierarchy. There are two types of that key which depend on the utilized method of authentication. If the authentication method is based on a pre-shared key, the top key is simply the pre-shared key itself and it is referred as Pre Shared Key (PSK). If the latter is based on the 802.1X framework, the top key is called Master Session Key (MSK).

These top level keys are used for generating the primary keying material in WPA2, which is the Pairwise Master Key (PMK). In the case of a pre-shared key based network the PMK is equal to the PSK, while in the 802.1X based network scenario the PMK is produced from a portion of MSK. The PMK is never used for encryption or integrity checks directly; rather it contributes to the generation of shorter-life keys.

In the next level of the keying hierarchy, the Pairwise Transient Key (PTK) and the Group Transient Key (GTK) exist. These keys are specific to the client-AP pair as they are produced during the authentication process from the PMK or the GMK respectively, as well as other random numbers negotiated with the client.

The PTK key is then split into five sub keys, i.e., temporal encryption key, two temporal Message Integrity Code (MIC) keys, EAPOL-Key Key Confirmation Key (KCK), EAPOL-Key Key Encryption Key (KEK). These are the bottom level keys in the WPA2 hierarchy. The KCK and KEK are used to protect EAPOL-Key frames while the temporal key is used to encrypt/decrypt unicast network traffic.

The GTK on the other hand, is split into two keys the Group Encryption Key (GEK) which is used for encrypting/decrypting multicast traffic, and the Group Integrity Key (GIK) which is used for verifying the MIC of multicast/broadcast traffic.

2) Traffic Confidentiality & Integrity: WPA2 supports three alternative protocols for protecting network traffic: Temporal Key Integrity Protocol (TKIP), Counter-Mode/Cipher Block Chaining Message Authentication Code Protocol (CCMP), and Wireless Robust Authenticated Protocol (WRAP).

TKIP is based on RC4 and is regarded as a transitional step from WEP which simply provides buffed up security and backward compatibility. On the downside WRAP, is based on the Offset Codebook (OCB) mode of AES which is considered much more secure but may be subject to licensing issues.

CCMP is based on the Advanced Encryption Standard (AES) algorithm in its CCM mode. It breaks the plaintext in chunks of 128 bits and encrypts them with a key of the same size. On the other hand, MIC computation is conducted with the Cipher Block Chaining (CBC-MAC) which initially encrypts a nonce block, the source address and the packet number and then XORs the result with each succeeding block. The MIC is attached at the end of the plaintext and it is encrypted along with it.

G. 802.11w

While the 802.11i focuses on the confidentiality and integrity aspects of the wireless communication it has been proven rather thrifty on the availability ones. In this way, DoS attacks discovered even since the WEP ages, fully apply on WPA/WPA2 settings too. The common denominator of most of these vulnerabilities is the fact that management frames are unprotected, thus easily issued even by unauthorized entities. For this reason the 802.11w amendment, which was approved in 2009, focused on these issues and addressed them by introducing the Robust Management Frames (RMF) mechanism which is merely the cryptographically protected version of some of the management frames (Deauthentication, Disassociation, and Action management ones).

In 802.11w the Robust Security Network Information Elements (RSN IE) field is extended by two bits (bits six and seven) to advertise the new capabilities which indicate that 802.11w is supported. More specifically, the sixth and seventh bit correspond to Management Frame Protection Required and Management Frame Protection Capable flags.

Unicast management frames are protected by the PTK, while for broadcast management frames, a new encryption key had to be introduced, namely the Integrity Group Transient Key (IGTK). The latter is used in a MIC information element. In further detail, the MIC is comprised of a packet ID, IGTK key ID, a serial number (IPN), and a cryptographic hash derived from the packet's MAC header and payload. IPN protects against replayed frames which are dropped if the same IPN has been used in the past.

To tackle Association Request attacks, the Security Association Query (SA Query) mechanism has been introduced.

This mechanism makes use of two new management frames, namely SA Query Request and Response which are exchanged between STA and AP as a follow up of every Association Request issued by the STA. The association procedure carries on only if the SA Query Response message is verified by the AP. The Timeout Information Element (TIE) is introduced for the cases where a STA is in associated state but somehow the serving AP receives a new Association Request message. The AP replies with a rejection notice and remains blindfolded to every Association Request stemming from the same client for a time interval equal to the one specified in the TIE field.

The traces contained in AWID as well as all the relative experimentation are conducted on a WEP protected network. We argue that such setting provides a rather attack-rich platform for experimentation. Since the attacks that can be executed in that platform are a subset of the existing attacks of versions up to 802.11w, the efficiency of the tools tested with AWID is expected to deviate minimally under such situations.

III. Attacks Against 802.11

This section is devoted to attacks against several versions of the 802.11 security mechanism (i.e. WEP, WPA, WPA2). Although the AWID dataset was gathered from a WEP protected network and contains only WEP related attacks, here WPA/WPA2 ones are also included for reasons of completeness. Note that physical layer attacks or higher layer ones are considered out of scope (the reader should refer to [22] for such) and therefore are not examined in the context of this work. Moreover, the emphasis is put on attacks that have practical value and have been implemented (or can be implemented relatively easy) by penetration testing tools. In this section we organize the attacks in groups of similar goals. Nevertheless, the reader must keep in mind that in the AWID dataset an alternative classification which is based on the implementation methodology is followed. This section concludes with an evaluation of the severity (impact) of these attacks drawn from empirical, experimental observations and theoretic assumptions.

A. Key Retrieving Attacks

This subsection concentrates on attacks that attempt to reveal the Secret Key. In all cases, the attacker simply needs to monitor for specific packets and then proceed with the key cracking process offline. While this passive practice is totally untraceable even by the most sophisticated, state-of-art IDS, the attacker will often choose to execute the active counterpart of these attacks which rely on the injection of a (large in most cases) number of packets in the network, possibly revealing herself.

1) FMS Attack: The FMS attack [23] was the first documented successful attempt to derive the WEP Shared Key, by taking advantage of a vulnerability on the key scheduling algorithm of the RC4 stream cipher. This attack is based on the theory of weak IVs [24, 25]. When such an IV has been used to encrypt a packet, then the attacker can make safe assumptions about the value of byte n+1 of the encryption key simply by having knowledge of the first byte of the keystream and the first n bytes of that key. In this case, the input conditions are easy to derive since the first byte of the plaintext is predictable (it can take one of the very limited number of values of the corresponding SNAP header field). After completing this process the attacker will have a possible value of byte n+1 but not definitely the actual one. So, she may choose to repeat this process for multiple messages that satisfy the weak IV condition. The real value will be encountered with significantly higher frequency than the rest. From that point on, the same cycle may be repeated for the rest of the bytes of the key.

2) KoreK Family of Attacks: A cryptanalyst with the pseudonym KoreK published (in the Netstumpler forum) implementations of seventeen attacks that aim at retrieving the WEP key. Each one of these attacks is based on similar mathematical principles as the FMS one, but makes use of different correlations. Once more, these approaches use statistical methods to vote for probable keys. A detailed analysis of this family of attacks is included in [26].

In any case, a significant amount of IVs must be collected in the hands of an attacker before she is capable of retrieving the WEP key. Although, typically, the KoreK incursions are more efficient than the FMS one, injection of packets to the network is still advised so that this process proves efficient in matters of time too. Actually, both the FMS and KoreK attacks have been used in conjunction to create an initial space of a limited number of possible keys and then carry on with a brute force attack to effectively reveal the correct one.

3) PTW Attack: PTW attack [27] was described by researches Pyshkin, Twes and Weinmann and was based on Klein's attack which targets the generic version of RC4 [28]. The PTW attack attempts to break the WEP key in a much more efficient way (i.e., with much less IVs/data frames) than the statistical methods.

In practice, this attack is constrained to ARP packets, thus making techniques such as ARP injection necessary for someone who wishes fast WEP key cracking. Nowadays, many WEP cracking tools consider this attack as their default cracking method mainly due to its efficiency.

4) ARP Injection: ARP injection is not actually an attack itself but it may be used as a first (frequently necessary) step for any of the Key cracking attacks [27] (especially from the IV greedy ones). The purpose of this attack is to manipulate the network in such a way, so that new IVs are produced steadily in large numbers even if no real traffic is moved in the network. These forcefully generated IVs will then be captured by the attacker and be fed to the respective Key cracking algorithms in a subsequent offline step.

Assuming that the attacker is already in possession of a Pseudo-Random Generation Algorithm (PRGA) she will construct an ARP Request packet with broadcast IPs, encrypt and finally transmit it. Upon reception, the AP will broadcast it to the network and a new IV will be produced for each ARP request. There are methods to achieve an ARP amplification effect meaning for each frame injected two or even three messages (with different IVs) will be produced. Such methods rely on the knowledge of the network topology and valid client IPs. In the later scenario the attacker constructs an ARP Request with an IP of a valid STA and transmits it to the

network. The AP will receive the message and re-transmit it towards the STA producing the first IV. The interested STA will receive the message and construct the appropriate ARP Response and then transmit it to the AP, producing a second IV. Finally, the AP will transmit the ARP Response back to the client producing a third IV.

5) Dictionary Attack: The Dictionary Attack is a form of brute force attack that has been widely used to retrieve weak WPA/WPA2 [29] and less frequently WEP keys (since more efficient methods exist). By today's standards this is considered the most reliable method for WPA/WPA2 cracking.

In the first phase of the attack the aggressor sniffs a target network hoping to catch a live handshake. Alternatively, she can target a victim client and actively issue deauthentication frames (usually a single or a very small number of frames) forcing the client to perform a 4-way-handshake immediately. In the first case, the attacker can be totally untraceable, while in the second the volume of the injected packets is so low that she also has a good change to pass undetected. In the second phase, the attacker goes through a process of generating the third message of a 4-way-handshake based on potential keys contained in a large database, usually referred to as dictionary. For each key, the attacker evaluates the result against the captured sequence and if the two match she can be certain that the currently evaluated key is the PSK of the network. The aforementioned process is done in an offline fashion. This methodology is described in Figure 1 analytically.

This attack is limited towards networks protected with the PSK method. It is considered effective only if the dictionary utilized contains the passphrase. So, the dictionary must be large enough, that is, to accommodate many possible passphrases. Also, its efficiency heavily depends on the computational power the attacker possesses. Although usually such dictionaries have size of multiple GBs the words contained on them are just a small fraction of the total combinations that can be used as a passphrase. For this reason, the attack will fail if the passphrase is not contained in the dictionary. The authors in [30] describe the various techniques that have been employed for retrieving the WPA/WPA2 key.

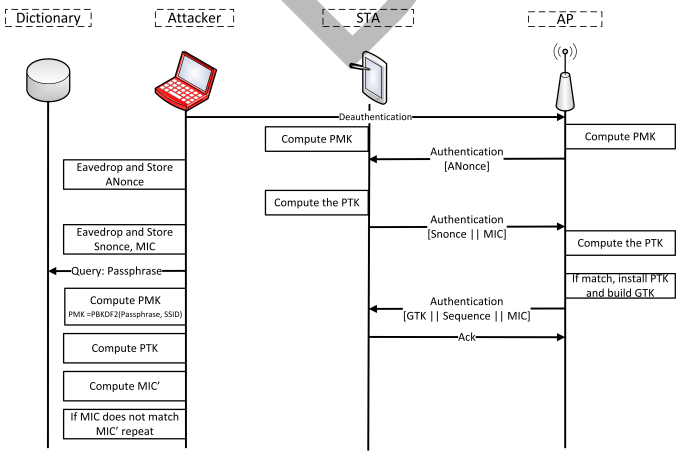


Fig. 1: Dictionary Attack

B. Keystream Retrieving Attacks

The profits of a cracked Shared Key are obvious. Yet, in WEP protected networks it is possible for an attacker to benefit even from the knowledge of the keystream alone. For example, one possibility for the attacker is to use the keystream to forge and inject packets to the network as a stepping stone for more serious attacks. This is possible as the standard allows the sender of a message to choose its IVs and at the same time it does not apply any technical means to forbid the reuse of IVs. Another (less popular) option is to decrypt portions of packets. That is by decrypting critical segments of packets the aggressor can learn the topology of a network or indirectly render herself able to decrypt all traffic by building a comprehensive database of keystream/IV pairs.

1) ChopChop Attack: ChopChop attack was also proposed by KoreK [31]. It allows an attacker to retrieve the m last bytes of both the keystream and the plaintext of a packet without having knowledge of the Key. The sources of this vulnerability are (a) the fact that CRC-32 is wrongly utilized in WEP for message integrity, and (b) WEP offers no protection mechanism against replaying previously sent packets.

The attack is based on 'chopping' the last byte of the encrypted portion of a packet and attempting to deduce the actual ciphertext value for this byte. Due to the missing byte, in this truncated form, the frame will have invalid ICV. Therefore, firstly the attacker XORs the truncated packet with a chosen value hoping that this value will lead to a sequence which will be valid for the specific ICV. Since the attacker has no means of knowing if the ICV is the valid one, she injects the modified packet in the network. Theoretically, the AP must reply with a message stating that the ICV is not valid, therefore revealing if her guess was fruitful or not. Ultimately, the attacker is using the AP as an oracle. If the ICV for an attempt is not valid then she simply repeats the process for all possible values of that byte (256 values). In the end, the attacker will know the plaintext of the truncated byte, and the keystream as well. Statistically, only $128m$ guesses are required on average and $256m$ guesses maximum to retrieve the last m bytes of a packet.

In practice, the ChopChop attack is usually executed with the purpose of deriving large portions of the keystream. This keystream will be used to forge and inject frames in a network on a subsequent step. Less frequent scenarios want this attack to be used for partially decrypting packets, especially when the attacker does not have knowledge of the WEP Key.

2) Fragmentation Attack: The fragmentation attack [32] aims at revealing a significant portion of the keystream by sending notably less messages than the ChopChop one. The keystream can later be used to generate and inject packets into the network as part of other assaults. Due to its efficiency it is sometimes embraced by attackers when they aim to create a full dictionary of keystreams for different values of IVs.

The fragmentation attack takes advantage of the fragmentation mechanism of 802.11. This mechanism allows any packet that exceeds the maximum frame length to be broken into several smaller fragments which can be sent independently. Also, this attack capitalizes on the observation that the first 8 bytes of any encrypted data frame are predictable. These bytes

correspond to the LLC header which is comprised by constant and known fields. That is, all but the last of these fields have fixed values. Even so, the last byte which corresponds to field EtherType (this indicates the protocol of the encapsulated packet) can take either the values for an ARP or IP packet. Nevertheless, the actual size of the packet can indicate whether one is dealing with an ARP or IP packet.

This attack assumes that the attacker has first falsely authenticated herself to the network. As a first step, the fragmentation attack requires the attacker to capture at least one data packet from the network. Since the first 8 bytes of plaintext are known, the attacker can deduce exactly 8 bytes of keystream with high probability. Of course, 8 bytes of keystream leave room for only 4 bytes of data (since the ICV itself requires 4 bytes), which is not sufficient for constructing any meaningful packet. At this point, the attacker takes advantage of the 802.11 fragmentation mechanism. She constructs a sufficient number of 8 byte packets with specific content and marks them as fragments. The protocol provisions that a message can be broken down to 16 fragments at maximum. Finally, she sends these packets through the AP to the broadcast address. Typically, the AP will reassemble the fragments and transmit them back, to all clients in a single packet. The contents of this packet are known beforehand. Hence, the attacker (by simply XORing) is able to retrieve keystream of size equal to the packet's length. The entire process along with the messages involved is described in Figure 2.

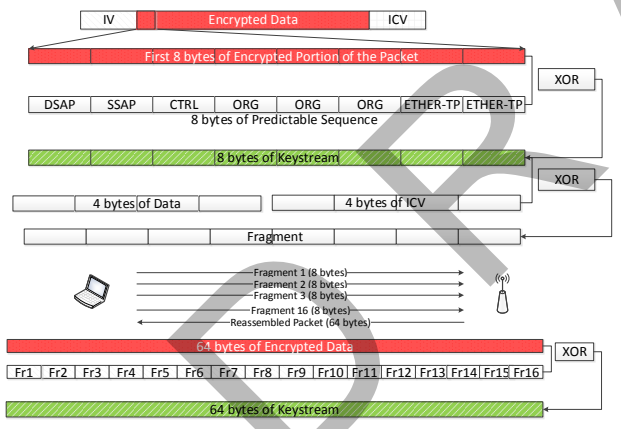


Fig. 2: Fragmentation Attack

3) Caffe Latte Attack: This attack was the first one used for retrieving the WEP key without requiring from an attacker to be within the range of the target network. The authors in [33] demonstrated that it is possible to retrieve the WEP key by attacking an isolated client who was once authenticated to the target network even if he is no longer near it (e.g., the client seats in a Cafe enjoying his latte).

The attack capitalizes on the fact that (a) clients usually maintain a list of known ESSIDs along with their corresponding keys (e.g., in Windows OS such keys are cached in the Preferred Network List (PNL) of the configuration manager), (b) most clients actively probe for such networks when they are in an unauthenticated state, revealing this way a list of networks with which they have been associated in the past, and (c) the client automatically attempts to connect to a network if that network has the same ESSID as any of the probed ones.

For this attack to be fruitful, the victim STA must be probing for known networks. As a first step, the attacker sniffs these probes and then using the appropriate equipment she poses as a valid AP (i.e., becomes a Honeypot) with the chosen ESSID. Next, the victim client authenticates and associates with the attacker's fake AP (this is possible since WEP does not incorporate a mechanism for authenticating the AP). At this point the client will typically request an IP address from a DHCP server (by sending several DHCP requests), but if not one is found, it will self-assign a private address and then send encrypted gratuitous ARP packets. Right then, the attacker will have to capture an encrypted ARP packet and modify certain bits in order to transform it into an ARP request packet. This is done in an attempt to learn the client's IP. The attacker will continue flipping specific bits of the ARP packet that correspond to the IP address in a brute force manner until she receives a matching ARP Response by the client. In this way, she can deduce the IP address of that client and transmit a series of encrypted ARP Requests messages. As expected, the STA will respond with a new ARP Response packet to each one of those messages, producing multiple new IVs, a fact which can be fully capitalized with the one of the WEP cracking attacks.

4) Hirte Attack: Hirte Attack [34] is another "AP-less" method for retrieving the WEP key using solely a client and not needing an AP of the network at all. It works in a similar fashion to the Caffe Latte one but it incorporates methods found in the fragmentation attack.

In a typical attack scenario, the attacker acquires an encrypted packet (a gratuitous ARP or IP packet) after setting up a Honeypot similarly to the Caffe Latte attack. Then, it relocates the IP address field by breaking that packet into fragments and changing their order. The concatenated packet will finally become an ARP Request one and from that point on a flooding of these messages can take place to harvest IVs for offline cracking attacks.

C. Availability Attacks

In this subsection attacks that may lead to loss of availability of service, commonly referred to as Denial of Service (DoS) attacks, are presented. Attacks of this category usually target either specific clients or try to stress the resources of the network (e.g. the AP) leading to DoS for all the clients in it. Most of the attacks discussed here, rely on the broadcast of forged 802.11 management messages. Such attacks are considered trivial to mount in versions of the standard up to 802.11n [35], since management messages are transmitted unprotected. Note that in all cases the negative results (i.e., the DoS effect) apply as long as the attacker mounts her attack (or at least they have a linear dependency with the duration of the attack). This practically means that (a) the effect of DoS attacks are not permanent, and (b) the attacker is required to be physically present within the range of the network during the course of the attack.

For a holistic survey of DoS attacks in 802.11 and possible countermeasures the reader should consult [36].

1) Deauthentication Attack: This is considered the most potent DoS attack in 802.11 networks due to its simplicity and efficiency. It is based on the fact that deauthentication packets are transmitted unprotected and they can easily be spoofed by an ill-motivated entity. Moreover, upon receiving such packets the client must abandon the network immediately without any additional actions.

The attacker monitors the traffic on the network to deduce the MAC addresses associated with a specific client and the one of the AP. Then, she forges a Deauthentication management frame and sends it to that client on behalf of the AP. Alternatively, she can send it to the AP on behalf of the client, so that the network stops considering that client authenticated. The client will immediately loose connectivity with the network, but typically it will re-initiate the authentication procedure automatically. This cycle is usually very brief but of course, the attack may be mounted repeatedly depriving the client of the service for a longer period of time.

2) Disassociation Attack: The disassociation attack is very similar to the Deauthentication one in both methodology, ease of use and effects. In this case, the attacker will send a Disassociation message instead. Theoretically, such attacks are less efficient because there is a smaller amount of procedures involved for the client to return from non-associated back to associated state. Thus, the duration of loss of service is expected to be shorter.

3) Deauthentication Broadcast Attack: The Deauthentication Broadcast Attack works in the same way as the simple Deauthentication one but instead of a client address in the corresponding field the aggressor inserts the broadcast address. This will cause all clients to receive this message and deauthenticate. The system might be stressed mildly since possibly all the connected clients in the network will attempt to initiate the authentication process roughly at the same time.

4) Disassociation Broadcast Attack: The Disassociation Broadcast Attack works similarly to the Deauthentication one but it utilizes the Disassociation message instead. The effects of this attack are analogous but it is expected to be less severe as the reassociation process is briefer and less computationally intensive. The authors in [37] focused their study on Deauthentication/Disassociation attacks and proposed a modification of the protocol based on one-way functions to counteract the effects of this attack.

5) Block ACK flood: This attack may cause an AP to voluntarily drop all packets originating from a specific client. It is effective against 802.11n networks and it achieves its results by taking advantage of the Add Block Acknowledgement (ADDBA) mechanism introduced in that version of the standard. More specifically, this mechanism allows a client to transmit a single large block of frames at once instead of several smaller segments [38]. An ADDBA message must be send on behalf of the client to notify the AP for its intention to conduct such a transaction. This message contains information such as the size of the block and the corresponding sequence numbers. After receiving such a message, the AP will only accept frames that fall within the indicated sequence and drop the rest.

To mount this attack the aggressor will simply have to falsify an ADDBA frame having the client's MAC address and large sequence numbers. All traffic transmitted from the client will be ignored until the sequence numbers indicated in the invalid ADDBA frame have been reached. This attack is hard to be detected as it is effective even by injecting extremely low volume of traffic in the network. This also means that the attacker needs not to be present during the entire course of the attack [39].

6) Authentication Request Flooding Attack: In this case the aggressor attempts to exhaust the AP's resources by causing overflow to its client association table. It is based on the fact that the maximum number of clients which can be maintained in the client AP's association table is limited and depends either on a hard-coded value on the AP or on its physical memory constraints. An entry on the AP's client association table is inserted upon the receipt of an Authentication Request message even if the client does not complete its authentication (i.e., is still in the unauthenticated/unassociated state).

Typically, an attacker will have to emulate a large number of phony clients and simply send an authentication frame on behalf of each one. After the AP's client association table overflows with fake entries, the AP will not be able to associate legitimate STAs any longer. This attack has been described and studied more extensively in [40].

7) Fake Power Saving Attack: This is the only DoS attack that does not rely on management frames but is rather implemented through null data frames (see section II-B1). The fake power saving attack was originally described in [41] and in theory it has the advantage that it requires a smaller number of frames to achieve its goal. By abusing the Power Saving mechanism this attack basically tricks the AP into thinking that a specific STA has fallen into doze mode. Note that the Power Management mechanism in 802.11 helps to reduce the power consumption of a STA by setting their network adapters into power saving mode. This state is also referred to as doze mode (more commonly known as sleep mode). The transition to doze mode is typically done when the client spends an amount of time without communication. For switching to sleep mode, the client first has to notify the AP about its intention through a null data frame with the Power Save bit set to 1. When in doze mode the client is not able to receive or transmit frames and the AP temporarily stores all frames destined to it.

So in essence, this attack takes place by sending a null data frame with the Power Save bit field set to 1. The AP will accept this message and immediately start buffering all data frames destined to that STA. The upcoming Beacon frame will contain a TIM field (with the client's MAC address), but since that client is not actually on power saving mode, it will be ignored. If this procedure is repeated for many cycles, sufficient time will elapse and the AP will be forced to discard all buffered frames. However, the exact time is depended on the adopted "Ageing Function" and is vendor specific. Actually, the role of the null data frames in 802.11 has been severely critisized [42] and eventually has been tackled by 802.11w.

8) CTS Flooding Attack: As explained in section III-C7 the Request to Send (RTS), Clear to Send (CTS) message

pair is an optional mechanism to control the access to the RF medium. When this mechanism is enabled and a STA has data for transmission in its queue, it sends an RTS frame to gain access to the RF medium for a pre-specified amount of time. This privilege is actually granted upon receiving the CTS frame.

In CTS Flooding Attack the attacker may constantly transmit CTS frames to itself or another STA, thus forcing the rest of the STAs in the network to continuously postpone their transmission.

9) RTS Flooding Attack: An RTS Flooding Attack also takes advantage of the RTS/CTS mechanism but works in an opposite way than the CTS Flooding one. It transmits a big number of spoofed RTS frames with possibly a large transmission duration window, hoping to monopolize the wireless medium in such a way that will eventually force the rest STAs to back-off from transmitting. For the interested reader, the works [43, 44] provide a detailed analysis and empirical evaluation of the numerous flavors of both CTS and RTS attacks.

10) Beacon Flooding Attack: The Beacon Flooding attack is a form of DoS attack that an aggressor may use in two different ways to achieve annoyance or complete denial of entry of new clients to the network [45].

In the first case, the attacker will transmit a constant stream of fake beacons that advertise non-existing ESSIDs. This will cause an overflow to the list of available networks, making it troublesome for the end-user to locate his preferred one. In the second case, the attacker will transmit a flood of spoofed beacon frames with a specific ESSID which correspond to different (non-existing) BSSIDs. Depending on the implementation, most probably the client(s) will go into a loop of checking if each of the synonymous ESSIDs corresponds to an existing network.

11) Probe Request Flooding Attack: A Probe Request Flooding Attack [46] aims at stressing the resources of an AP and eventually drive it to paralysis. This attack is based on the fact that according to the 802.11 standard an AP is obligated to reply to every Probe Request message with a Probe Response one.

Such messages contain details about the network and the capabilities of the AP. An attacker may send a constant stream of fake Probe Request packets. If this is done in high volume and for prolonged periods of time the AP will not be able to afford serving its legitimate clients too, as it will be probably struggling to reply to the probes of the non-existing ones.

12) Probe Response Flooding Attack: This attack also takes advantage of the Probe mechanism although it works in reverse by targeting the client rather than the AP.

This time the attacker monitors for probe request messages coming from valid clients and by acting like an AP, she transmits a flood of fake and inaccurate probe responses to the STAs. These messages contain bogus information about the network, thus misleading the STA from receiving the response from the valid AP and further preventing it from connecting to any AP.

D. Man-in-the-Middle Attacks

1) Honeypot: In the context of this work Honeypots [47] are networks created and controlled by malicious administrators to attract naive users and then perform different attacks to them. Usually, such networks are open and advertise luring ESSIDs (e.g., Free Internet, Free WiFi, Open Hotspot etc.) in order to maximize the number of clients connected to them.

When users connect and since no encryption is applied (or if so, the key is already in possession of the malicious user) all (unencrypted by the higher layers) traffic is visible to the attacker. Furthermore, the attacker may use sophisticated penetration testing tools to discover security holes to that client and then launch higher-layer attacks (e.g. Session Hijacking) to bypass even higher-layer security mechanisms.

A Honeypot is not considered an attack per se and there are no means to detect if a given network is actually a Honeypot or not (at least not in the MAC level). It is the responsibility of a user to connect to reliable networks only. Nevertheless, IDS working on higher layers will be able to detect intrusion attempts normally.

2) Evil Twin: An Evil Twin is a special case of Honeypots that advertise an existing ESSID to fool naive users into connecting to it instead of the valid network [48]. Evil Twin APs are possible due to the fact that (a) multiple APs with the same ESSID is allowed to exist in the same area, and (b) in such situations the client will prefer to connect to the one with the strongest signal disregarding the BSSID of the legitimate AP.

Initially, the attacker brings up a fake AP (usually a software one) that advertises the same ESSID with a valid one in the vicinity. Preferably, the impersonated networks must be open (e.g., networks of coffee shops, airports etc.) or at least their credentials should have been acquired by the attacker first (e.g., the case of a hotel wireless connection). Naturally, if the attacker's Network Interface Card (NIC) transmits with a stronger signal then the client will prefer to connect to that fake network. As in the case of a normal Honeypot, from that point on the attacker is able to launch higher level attacks or simply monitor the traffic.

3) Rogue Access Point: Rogue APs are unauthorized access points (i.e., either hardware or software AP) enabled within the corporate, home or office premises by an insider of that network. Such settings may be spawned by undisciplined users without the permission of the network administrator in order to render a security policy more convenient for them or by traitors with an ulterior purpose to leave a backdoor open for outsiders.

Rogue APs are usually connected to the wired counterpart of the network although the wireless connection is not uncommon especially for software APs. Such APs can be open if an attacker wants to attract a larger number of users or be protected with a shared key, if the insider wants to allow access to specific allies. The detection of such devices is a challenging task. Works such as [49] have studied this concept more extensively, while the work in [50] indicates methods to suspend rogue APs and response to their threats.

E. Categorization

In the previous subsections the described attacks were organized according to their conceptual similarity and ultimate purpose, in the following classes: (a) key cracking attacks, (b) keystream retrieving attacks, (c) denial of service, (d) man in the middle. Of course, this classification is not the only one. For example the authors of [51] categorize the attacks according to their target into: (a) network attacks, and (b) client attacks. We argue that such categorizations of the attacks may not be as valuable for intrusion detection purposes. In these scenarios the detection engine is trying to infer common patterns among the attacks of the same category. Hence, in the AWID dataset we have introduced a categorization according to the methodology of execution. As a result, attacks that have similar patterns of expression fall under one of the groups: (a) injection attacks, (b) flooding attacks, (c) impersonation attacks, (d) passive attacks. As expected, passive attacks are not included in the dataset as they leave no digital footprints. Table I presents alternative categorizations of the attacks described in this section.

IV. Evaluation of Attacks

We attempted to measure the impact of the attacks described in section III. To this end, several different devices were employed ranging from mobile devices such as a Nokia Lumia 800, an iPhone 2, a Samsung Nexus smartphone, a Samsung Galaxy Tab tablet, as well as two desktop PCs with a Linksys WUSB54GC and D-Link DWA-125 wireless USB adapters running Ubuntu Linux 12.04 and Windows 7 respectively. Standard wireless penetration testing tools were employed such as the Aircrack suite [7] and the MDK3 [52] tool. Whenever specific attacks were not offered by any publicly available tool, custom scripts were implemented. For example, Probe Request Flooding attack was fired with File2air tool [53] (using the Lorcon-old library [54]), while the Fake Power Saving and the Disassociation attacks were unleashed by custom C programs implemented using the Lorcon2 library [55]. The most important of the conclusions are denoted hereunder.

A. WEP Cracking Attacks

Most of the documented WEP cracking attacks are based on some kind of statistical observations of a network's traffic, however the amount of traffic needed to actually crack the key is non-deterministic. The basic characteristic of all these methods is that they require a large number of IVs which may be obtained by monitoring the traffic for encrypted data frames (such as ARP or IP packets). Such attacks can be absolutely passive and in this way totally untraceable. However, in practice, this is rarely the case as attackers inject traffic to the network (usually ARP packets) to trigger responses (enforcing the generation of new IVs), thus speeding up the process and making such attacks practical. Actually, several techniques including ARP amplification or double ARP amplification exist to boost the generation of IVs even further. Based on statistical observations Table II summarizes the estimated amount of IVs required for successful cracking by popular attacking methods.

TABLE II: Average IVs required for WEP cracking by various attacks

Attack	IVs (average)	Success	Year
FMS	5,000,000	50%	2001
KoreK	700,000-2,000,000	50%	2004
PTW	40,000-500,000	50%-95%	2007
VX	32,700	50%-95%	2007
Modified PTW	24,200	50%-95%	2008

To offer a clearer view of the amount of IVs required versus the amount of IVs generated through everyday traffic, we conducted several experiments on different use case scenarios. More specifically, we calculated the average amount of IVs per minute generated by applications such as video streaming, moderate web page browsing, file downloading, as well as intrusive scenarios such as ARP injection attacks. All scenarios assume having one client connected to the examined network. Figure 3 summarizes the results obtained per application.

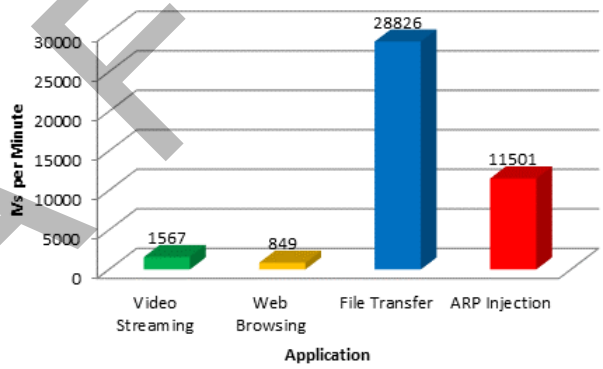


Fig. 3: Average amount of IVs per minute generated by various applications

From the figure it becomes obvious that in networks with low to moderate load the attacker must wait several hours (for the case of FMS) to several minutes (for the case of PTW) to gather the appropriate amount of IVs that will allow her to proceed further and unveil the key.

B. Deauthentication and Disassociation Flooding

As already mentioned in section III-C Deauthentication flooding attack is the most popular DoS attack in 802.11 networks.

In our experiments we used a range of different devices and measured the elapsed time from the moment a Deauthentication frame is sent by an aggressor to a victim STA, until the moment the STA gets fully re-associated to the AP. We noticed that in most of the cases these cycles are non-neglectible (e.g., greater than one second). This dictates that a relatively small number of packets per minute is enough to significantly disrupt a victim's communication, if not cause a complete DoS. On the one hand, this conclusion is contradictory to the common practice of the most popular modern wireless injection tools (Aircrack suite, MDK3) that aggressively transmit hundreds of

TABLE I: Different Categorization Schemes for 802.11 Attacks

Attacks	By Purpose				By Target		By Methodology			
	Key Cracking	Keystream	DoS	M-i-M	Network	Client	Passive	Injection	Flooding	Impersonation
FMS	✓				✓		✓			
Korek	✓				✓		✓			
PTW	✓				✓		✓			
ARP Injection	✓				✓			✓		
Dictionary	✓				✓		✓			
Chop-Chop		✓			✓			✓		
Fragmentation		✓			✓			✓		
Caffe Latte		✓			✓					✓
Hirte		✓			✓					✓
Deauthentication			✓			✓			✓	
Disassociation			✓			✓			✓	
Disassociation			✓			✓			✓	
Deauthentication broadcast			✓			✓			✓	
Disassociation broadcast			✓			✓			✓	
Block Acknowledge			✓			✓			✓	
Authentication Request			✓			✓			✓	
Fake Power Saving			✓			✓			✓	
CTS			✓			✓			✓	
RTS			✓			✓			✓	
Beacon			✓			✓			✓	
Probe Request			✓			✓			✓	
Probe Response			✓			✓			✓	
Honeypot				✓		✓				✓
Evil Twin				✓		✓				✓
Rogue AP				✓		✓				

Deauthentication frames per second. Further, judging by the experimental results in [56] (which was published in 2003), we can assume that nowadays manufacturers tend to construct NIC cards which complete the re-authentication cycle faster.

For the deauthentication attack we relied on the Aircrack suite, but due to the lack of support of a pure Disassociation attack by any of the existing penetration tools the attack was launched by a separate self-implemented tool. Figure 4 presents the time elapsed from the moment a Deauthentication/Disassociation frame is transmitted until the device gets fully re-associated with the network. By comparing the deauthentication cycles with the disassociation ones we noticed that the latter are noticeably greater. This conclusion contradicts to our initial hypothesis that the disassociation cycle is briefer because of the smaller number of actions involved. In practice, such cycles are longer due to the fact that upon receiving a disassociation frame the STAs will first issue a deauthentication frame to the AP and then go through a complete authentication and association/re-association cycle. This behavior is not according to the standard but has been observed for all of our test subjects. It is clear that in this way significant disruption of typical client routines (e.g., web browsing, app downloading, VoIP calling, video streaming) can easily occur with as few as 100 frames per minute. Actually, even the devices with the fastest re-authentication cycles will be crucially affected.

It is worth mentioning that the use of WPA over WEP did not have any substantial impact on the re-authentication/re-association cycle.

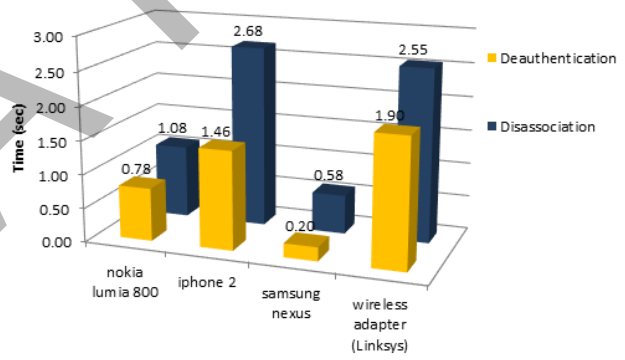


Fig. 4: Deauthentication vs. Disassociation Cycles for several Devices

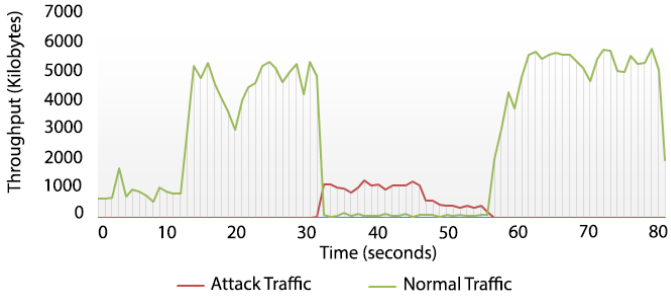
C. Probe Request Flooding

During all the experiments considering this attack we did not notice actual DoS against any number of users of the network. However, what was apparent was annoyance in the form of reduced throughput. While the theoretic ground of this attack is based on the goal of exhausting the physical resources of an AP, according to our experiments the main cause of commotion stems from the signaling overhead imposed on the wireless medium. It must be made clear that a single probe request frame triggers multiple responses from AP's in the vicinity simultaneously. Thus, it is easily understandable that the more APs exist in the neighborhood the more effective the attack gets.

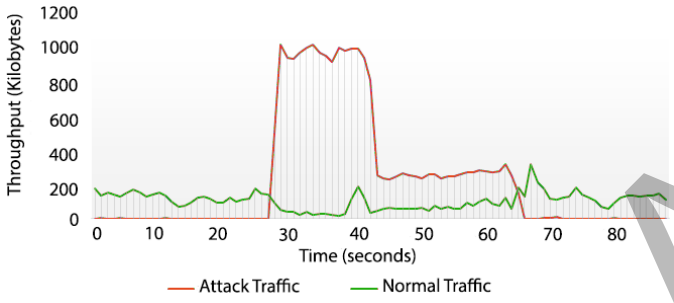
We believe that it is much more realistic for an attacker to cause havoc to a network in this way rather hoping for driving a contemporary AP (even a low-end home device) to its physical limitations and to force it to drop clients.

Our experiments were conducted with a custom-tailored version of the File2air tool that allowed us to send 5,000 Probe Request packets in total with variable MAC address fields (all corresponding to existing manufacturers) at variable rates. We

evaluated the results in both TCP (FTP file transfer) and UDP (Skype call) application scenarios. More specifically, in the UDP scenario we noticed that the throughput dropped from 145Kbps to 68Kbps and in the TCP scenario from 2Mbps to 269Kbps. This is translated to a loss of 53% and 87% respectively. Figure 5 presents the drop of throughput noticed for the TCP and UDP scenarios when a Probe Request flooding attack unfolds.



(a) Drop of Throughput in TCP Scenario



(b) Drop of Throughput in UDP Scenario

Fig. 5: Effect of Probe Request Flooding Attack in Throughput

D. Beacon Flooding

As already mentioned, this attack comes in two flavors: (a) transmitting beacons that advertise non-existing ESSIDs, and (b) transmitting beacons that all advertise an existing ESSID, but correspond to different (non-existing) BSSIDs.

The first case does not cause real DoS but may prove a factor of commotion. Actually, the eminence of nuisance depends upon the patience of a user locating the network of interest in a (unusually large) list of ESSIDs, most of which have random (thus meaningless and unusual) names.

For the second variation of this attack, we executed our experiments with the use of MDK3 by injecting Beacon frames that advertise the same ESSID as the legit AP, but with different (random but corresponding to existing manufacturers) BSSID. This attack successfully prevented the entry of new clients to the network for all handheld devices except Samsung Nexus. For the laptop machines the ones equipped with Windows 7 OS seemed to be immune to this attack. Still, the attack was successful against the Linux equipped machine.

As expected, this type of Beacon flooding attack had no success with the already connected devices.

E. Authentication Flooding

While in theory the Authentication Flooding attack attempts to exhaust the physical resources of the APs, experimental results indicate that contemporary devices can cope well, even for an extreme number of simultaneous authentication requests. Actually, even after 8 million authentication attempts on a single commodity AP that was used as our test subject, we observed no noticeable deviation from the AP's normal behavior (i.e., freeze or reset).

What is interesting however, is the fact that during the course of this attack even in its early stages (i.e., the first two seconds) the client was unable to perform authentication to enter the network. More specifically, all devices presented such behavior with the sole exception being the Samsung Nexus, which was able to connect but with a noticeable delay.

This attack may pose as a more effective equivalent of the Beacon Flooding. The above mentioned experiments were conducted with the use of the MDK3 tool and an average injection rate of 900 authentication frames per second. For a complete overview of flooding attacks in 802.11 along with simulated evaluations the interested reader should refer to [57].

F. ChopChop

We conducted our experiments with the Aireplay-ng tool (of the Aircrack suite). In the course of the attack we replayed packets of different sizes. We came to the conclusion that the amount of time required for the ChopChop method to fully analyze a given packet depends on the actual size of the packet. Some examples of various packet sizes and the corresponding requirements in number of packets to be injected and amounts of time are given in Table III.

TABLE III: Requirements in Number of Frames and Time for ChopChop Attack

Size	Frames Injected	Total Time
70	6550	131
80	9445	187
122	13255	264

From the table it is clear that a significant amount of packets needs to be replayed back to the AP for the ChopChop to complete successfully. However, even traffic of such magnitude can be camouflaged in busy networks if the packet size is the only criterion of detection. On the other hand, the replayed packets will have several identical fields, including the IV one. While fields such as IV are randomly selected, and as such they are subject to possible repetitions, it is highly unlikely that the IV field of numerous packets in a short amount of time, say, 1 sec, will be identical. This fact is illustrated in Figure 6.

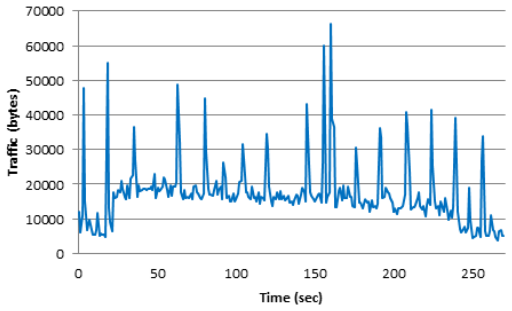
Table IV presents a summarizing overview and evaluation of the attacks discussed previously. These are included in the dataset which will be presented in the process.

V. Attack Signatures

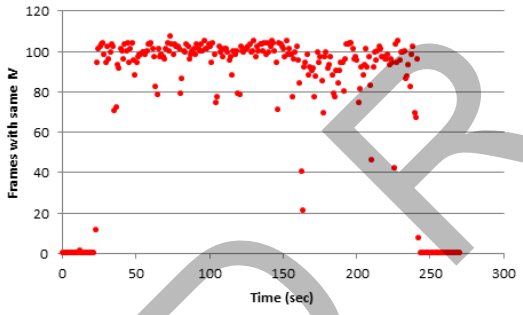
In this section we analyze the 15 attacks included in the training version of the reduced AWID dataset (AWID-ATK-R-Trn) detailed in section VI. This undertake aims in highlighting possible attack patterns from a theoretical as well as practical

TABLE IV: Summary and Evaluation of Attacks

Attack	Effect	Traffic Injected	Version	Difficulty	Comments	Threat
FMS	Secret Key Cracking	>2,000,000	WEP	Easy	Slow	★★
Korek	Secret Key Cracking	>700,000	WEP	Easy	Slow	★★
PTW	Secret Key Cracking	>50,000	WEP	Easy	Fast	★★★
Dictionary	Secret Key Cracking	1	WPA/WPA2	Easy	Requires resources depends on weak passwords	★★
Chopchop	Keystream Retrieval Packet Decryption	<=256*m	WEP	Moderate	-	★★
Fragmentation	Keystream Retrieval Packet Decryption	<=16	WEP	Moderate	Reveals up to 64	★★
Caffe Latte	Secret Key Cracking without AP	<=65280	WEP	Easy	Slow not possible against all OSs	★
Hitre	Secret Key Cracking without AP	1	WEP	Easy	Fast	★★★
Deauth	Loss of Connectivity	High	All	Easy	Can Target Client	★★★
Disassociation	Loss of Connectivity	High	All	Easy	Can Target Client	★★★
Deauth Broadcast	Loss of Connectivity	High	All	Easy	Affects All	★★★
Disassociation Broadcast	Loss of Connectivity	High	All	Easy	Affects All	★★★
Block Ack	Annoyance	Low	802.11n	High	Requires Accuracy	★
Authentication Request	Inability to join the network	High	All	Low	Ineffective Against Most Devices	★
Fake PS	Annoyance	High	All	High	Requires Accuracy	★
CTS Flooding	Annoyance	High	All	Low	Can Target Client	★
RTS Flooding	Annoyance	High	All	Low	Can Target Client	★
Beacon Flooding	Inability to join the network	High	All	Low	Effective Against Limited Devices	★
Probe Request	Annoyance	High	All	Low	Affects All	★★
Probe Response	Annoyance	High	All	Low	Can Target Client	★★
Honeypot	Loss of Privacy	None in the Network	All	Moderate	Relies on Naive Users	★★★
Evil Twin	Loss of Privacy		All	Moderate	Requires Knowledge of Secret Key	★★★
Rogue AP	Loss of Privacy	None in the Network	All	Moderate	Requires Access to the Wired Network	★★★



(a) Increase in Traffic



(b) Number of Packets with Repeated IVs

Fig. 6: ChopChop Attack

perspective. To the best of our knowledge this is the first documented attempt to fingerprint 802.11 attacks. We expect that this will lead to the better understanding of the structure and the characteristics of such anomalies when they occur in wireless networks.

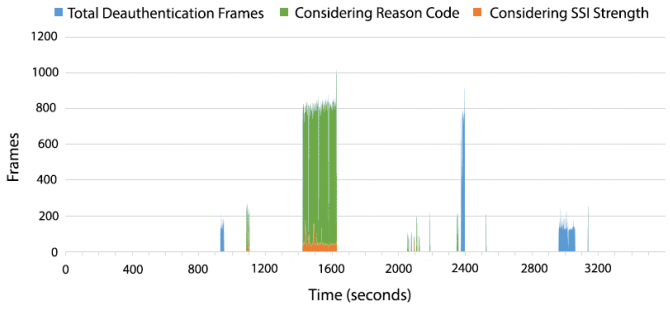
A. Flooding Attacks

Flooding attacks create a sudden increase in the management frames (in their majority) per time unit. Even though this makes it easy to discern a flooding attack from normal traffic, it is not always as straightforward to distinguish it from other attack types. Some assaults may cause a temporary increase of certain management frames as collateral damage. As will be explained in the process, additional hints and traces usually exist, although, the reader should keep in mind that they are contingent on the tools used during each attack. Whereas, there may be certain actions an aggressor can make to masquerade the traces produced by the specific tool in use, there is very little she can do to conceal the increase in the number of management frames as this is the basis of all flooding attacks.

The Deauthentication Flooding attack is considered as one of the most potent DoS attacks in the wireless realm, yet it is also one of the hardest to accurately identify. During its course, a burst of Deauthentication frames is generated. However, elevated levels of such frames may also be tracked in Amok, Dissassociation, Power Saving, Authentication Request Flooding attacks, as well as in the ARP Injection one when executed improperly. For example, consider the case where such frames are transmitted by intruders that possess or impersonate non-authenticated MAC addresses. It is important to keep in mind that only in the case of Deauthentication Flooding attack the Deauthentication frames are forged and transmitted by the adversary, while in the rest of the cases they are products of the AP itself as part of a valid response to the attack. Aireplay, which is the de-facto tool for launching Deauthentication flooding attacks, transmits management frames that have the same Reason Code (0x0007), and at the same time their Sequence Number field contains values that are outside the natural order for that session.

Figure 7a shows the total amount of Deauthentication frames throughout the entire duration of the reduced training set. Notice that only few timeslots contain Deauthentication frames that have the Reason Code field set to 0x0007. Even so, there are cases where Deauthentication frames with the exact same Reason Code do not correspond to a Deauthentication Flooding attack. For example, the timeframe between seconds 1050 to 1150 contains frames originating from an actual Deauthentication Flooding attack, while the timeframe between seconds 1400 to 1600 contains Deauthentication frames that are produced by the AP as a valid response to an ineffectual ARP Injection attempt. Figure 7b focuses in this time zone. Notice that the Signal Strength criterion that is applied as a last resort reveals of the actual Deauthentication Flooding attack. As observed from the figure, in the first time slot, there is

a significant percentage of packets that deviate from a certain threshold of Signal Strength, while in the second time slot this percentage is kept low.



(a) Deauthentication Management Frames

(b) Zoom on Deauthentication Management Frames during Seconds 950 to 160

Fig. 7: Patterns of Traffic during Deauthentication Flooding Attack

During an Authentication Request Flooding attack the Authentication frames are expected to show a significant increase. Naturally, increased numbers of Authentication Requests can also be noticed in the Amok as well as the Deauthentication Flooding attacks, but in the case of the Authentication Request Flooding the accumulated volume is much higher. This attack is mainly launched via the MDK3 tool which always transmits Authentication frames with a static Listen Interval field (value 0x0000). Additionally, the Tagged Parameters field always contains the same kind of parameters (remember this field is variable) which in numbers are fewer than the usual. Finally, the sequence number has always the constant value of 0.

A Beacon Flooding attack causes a vigorous increase in the quantity of Beacon frames. Typically, the advertised ESSIDs are new and short-lived (i.e., not many Beacon frames with the same SSID are transmitted), while frequently they have uncanny, randomly generated names. An increase in Beacon frames occurs naturally in all impersonation attacks too, but in such cases the ESSID bears the value of a network that already exists in the vicinity. The MDK3 tool is the only one that offers an implementation of this attack. Similarly to the Authentication Request Flooding attack, the generated frames have a Timestamp field of static value (0x0000000000000000). Secondly, the Sequence Number does not increase and remains 0 for all frames. Finally, the Short Preamble and Short Slot Time flags are simultaneously set to 0. After observing the beacon frames during the attack free periods the aforementioned characteristics seem as a statistical paradox. Figure 8 displays the total number of Beacon frames in the training set, as well as those Beacon frames in that set that meet the MDK3 signature attributes. Note that even with the use of the first filter alone (blue area) it is easy to identify the time frame within which a Beacon attack unfolds with high accuracy, however the use of the second filter (orange area) achieves optimal results.

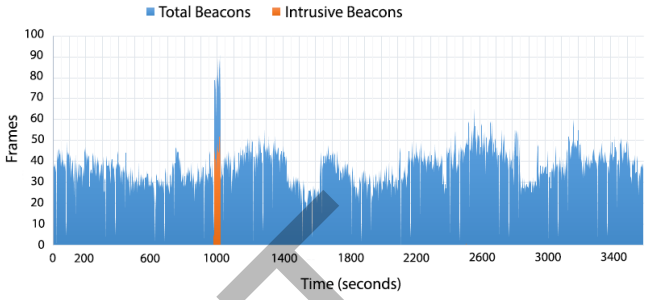


Fig. 8: Traffic Pattern during Beacon Flooding Attack

The Probe Response Flooding attack results in an outburst of Probe Response frames. An increase of such frames is also observed during the impersonation assaults but it is generally much milder. The Metasploit tool has a mode of attack (payload), which allows an aggressor to discharge such attacks. Probe response frames crafted with Metasploit have a totally random sender address, i.e., it may not have a valid Organizationally Unique Identifier (OUI), the Beacon Interval field does not have the usual value (which is 0.102400) but rather a random one, and the Sequence Number follows an out-of-order increment.

B. Injection Attacks

Injection attacks usually cause a deluge of validly encrypted data frames of smaller size.

In ARP Injection attacks the aggressor is inclined to transmit a large number of small data frames for a significant amount of time, hoping to evoke the appropriate response from the network. Currently, Aireplay is the preferred tool of hackers for unleashing attacks of this kind and by analyzing the structure of the frames this tool generates, it is obvious that they have identical IV values, something which is statistically impossible to occur in such brief timeframes under normal conditions. Additionally, the DS Status flag is set to 1 which in turn is another indication of an ARP Injection attack.

Figure 9 highlights the fact that small sized Data frames may occur under various conditions not exclusively by ARP Injection attacks. However, when seeking for small sized Data frames that have repeating IVs, one may identify ARP Injection attacks with satisfactory accuracy. The reader should notice time durations between the second 1,400 to 1,600 and 2,800 to 3,000 which refer to ARP Injection attacks. The first case represents a failed attempt since the amount of Data frames that have identical IVs is the same as the total amount of Data frames. On the other hand, the second timeframe corresponds to a successful attack as the number of total small sized Data frames (i.e., ARP Requests plus ARP Responses) is about three times the amount of the small sized

Data frames with repeating IVs (i.e., ARP Requests injected by the attacker).

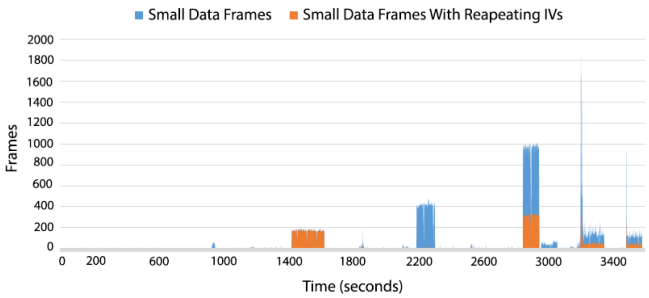


Fig. 9: Traffic Pattern during ARP Injection Attack

During a Fragmentation Attack the intruder injects a sequence of short, fragmented data frames. If successful, this process usually does not consume more than one second, however if not successful the same procedure will be repeated. The Aireplay tool contains an implementation of this attack and by examining the packets it produces we notice that all have a static, invalid value in the Destination Address (ff:ff:ff:ff:ff:ed) field, the DS status flag is set to 1, the length of the frame is small (but not fixed) and finally the sequence number is out-of-order. Not surprisingly, the More-Fragments flag is set to 1 and the fragment number field is greater than zero in all but one of the fragments in the chain.

C. Impersonation Attacks

Impersonation attacks introduce an additional AP in the neighborhood broadcasting Beacon frames that advertise a pre-existing valid network (i.e., that of the victim's). The common denominator of all Impersonation Attacks is that the number of Beacon frames of the victim network is approximately doubled. Quite frequently these attacks are combined with a short flood of Deauthentication frames as an initial step, so that the attacker may force the STAs to connect to its own rogue AP.

Typically, attackers rely on the Airbase tool of the Aircrack suite to launch Evil Twin attacks. As expected, additional Beacon frames are broadcasted but in this case they have significantly different characteristics. For example, the Timestamp field has a fixed value (0x0000000000000000) for all the forged beacon frames, and the Tagged Parameters field contains steadily a different number of parameters.

Figure 10 displays the number of Beacon Frames having the ESSID of the victim network. The reader should notice that there are timeframes during which the amount of these Beacons is almost doubled. These durations correspond to impersonation attacks, and this conclusion is verified by the fact that approximately half of these frames possess intrusive characteristics.

Caffe Latte attacks are more complex in nature. Since they fall into the Impersonation attacks category, they will introduce additional Beacon frames, all having the ESSID of the victim network. As expected, these frames will also bear the same signature characteristics as the ones transmitted during an Evil

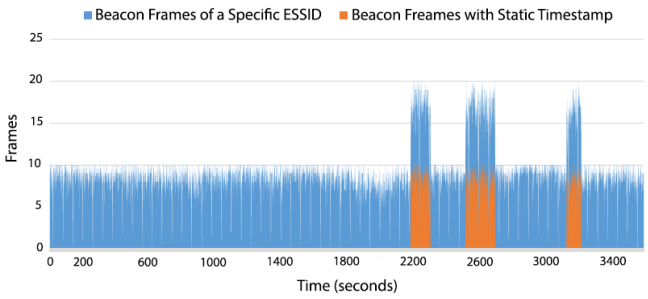


Fig. 10: Traffic Pattern during Evil Twin Attack

Twin attack when the Airbase tool is utilized. However, Caffe Latte assaults will simultaneously inject encrypted Data frames of small size, much like a normal injection attack, making it harder to clearly distinguish it from an ARP Injection or Evil Twin attack for instance.

As a final note, in all cases described above the received Signal Strength of all forged frames (as indicated by the corresponding Radiotap Header field) will probably fall within a different range of values (usually forged frames have higher Signal Strength) than that of the validly generated ones. This criterion is not undisputed but when applied as a statistical means and combined with other factors, it is usually indicative of an attack.

Works like [49, 58] propose possible ways for identifying rogue AP and evil twin attacks, while [44] propose systems for recognizing malicious CTS/RTS packets. We expect that the clear statement of attack signatures as the ones presented in this paragraph will contribute to the improvement of analogous defensive systems in the future.

VI. Dataset

This section describes the AWID family of datasets with respect to its collection methodology, structure and contents.

We anticipate that not only this dataset can act as a reliable testbed for intrusion detection experiments in wireless networks, but also its study can reveal valuable information about the conditions that take place on a singling level when different types of attacks occur on a real wireless network.

A. Data Gathering

For purposes of data gathering we created a physical lab which realistically emulates a typical SOHO infrastructure. A number of mobile and stationary STAs were used as the valid clients of the network, while a single mobile attacker was unleashing various attacks.

More specifically, the valid network consisted of 1 desktop machine, 2 laptops, 2 smartphones, 1 tablet and 1 smart TV. The position of the desktop machine and smart TV remained static throughout the course of all the experiments. The smartphone devices displayed high mobility, i.e., they changed position inside the facilities of the lab and joined/left the network numerous times throughout the course of the experiments. Finally, the laptop machines were semi-static, i.e., they rarely changed their position. The services running

TABLE V: Specifications of the Equipment Used in the Experiments

Node	Type	Brand	OS	Network Card	CPU
Client1	Desktop	Custom	Ubuntu Linux 12.04 LTS	Netgear WNA3100 N300	Intel Core i7 3.2GHz
Client2	Laptop	Fujitsu-Siemens	Ubuntu Linux 12.04 LTS	Intel 3945ABG	Intel Core Duo T2050 1.6GHz
Client3	Laptop	Acer	Ubuntu Linux 12.04 LTS	Qualcomm Atheros AR9462	Intel Core i5 1.7GHz
Client4	Smartphone	iPhone 3G	iOS 4.2	NA	Samsung 32-bit RISC ARM 620MHz
Client5	Other	iPod Touch	iOS 3.1	NA	Samsung 32-bit RISC ARM 533MHz
Client6	Laptop	Acer Aspire 5750G	Windows 7	Broadcom BCM943227HM4L	Intel Core i5 2.8GHz
Client7	Smartphone	HTC Diamond	Windows Phone 6.1	NA	528 MHz ARM 11
Client8	Smartphone	Samsung Nexus	Android 4.2	NA	dual-core ARM Cortex-A9 1.2 GHz
Client9	Tablet	Samsung Galaxy Tab	Android 2.2	NA	Cortex-A8 1 GHz
Client10	Smart TV	LG 42LM7600S	Linux	NA	NA
Attacker	Laptop	Acer Aspire 5750G	Kali Linux 1.0.6	D-Link DWA-125/Linksys WUSB54GC	Intel Core i5 2.8GHz
Monitor Node	Desktop	Custom	Linux Debian 7.3	Alpha AWUS036H	Core i7 2.4Ghz

on the clients which were responsible for producing traffic were web browsing, VoIP, and file downloading.

The network was covered by a single AP, which was a Netgear N150 WNR1000 v3 device (Firmware Version V1.0.2.54_60.0.82), protected by the unreliable WEP encryption, supporting up to 54Mbps transfer rates.

All attacks were unleashed by a single attacking node using an Acer Aspire 5750G laptop running Kali Linux 1.0.6 64 bit. The attacker was equipped with a D-Link DWA-125 card in promiscuous mode for injecting packets. The assailant frequently changed her MAC address among the various attacks. To implement the attacks various tools were used including the Aircrack-ng suite, the MDK3 tool, the Metasploit framework [59] as well as custom made ones implemented in C language using the Lorcon2 library. The intruder was mobile and was acting outside the perimeter of the facilities of the lab.

For capturing the wireless traffic a separate device was introduced as a monitor node. This node was placed inside the network coverage but was never associated with it. The monitor node was a desktop machine, running on Linux Debian 7.3, equipped with a Samsung 840 series SSD hard drive capable of writing 130 MB/s and an Alpha AWUS036H card, set in promiscuous mode. The Tshark application (which is the terminal version of the Wireshark [60]) was installed on that node and used for logging the traffic in several pcap files of smaller size (contain traffic captured during 1 hour). Note that this approach does not guarantee that all packets will be captured. Yet, we highlight the simplicity and cost efficiency of this method and argue that it is the most reliable data capturing approach for resource constrained environments such as SOHO ones. After the monitoring phase, an intermediate one took place where the produced CSV files were subjected to a process of normalization for specific fields (e.g., MAC addresses were represented as integers). For the interested reader, the corresponding scripts are included in the online resources [61, 62] of the manuscript.

Figure 11 illustrates the blueprints of the lab and the relative positions of the nodes inside the lab facilities throughout the course of dataset collection, while Table V contains detailed description of the equipment used throughout the course of our experiments.

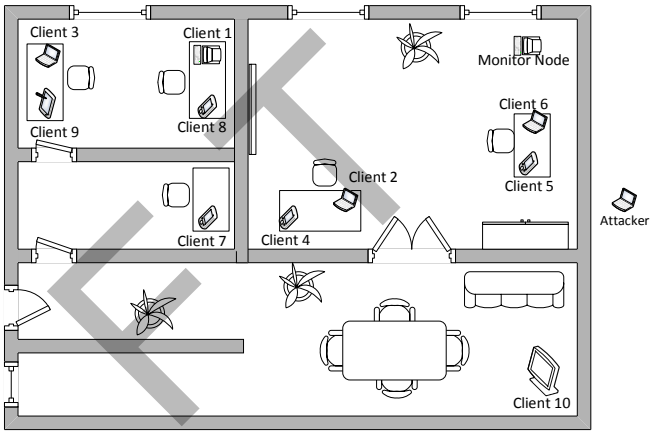


Fig. 11: Lab Blueprints

B. Types of Dataset

The AWID collection of datasets is comprised of two equal sets which defer merely on the labelling method (AWID-CLS, AWID-ATK). The first one is labelled according to the classification introduced in section III-E (4 classes), while the latter follows a more detailed classification based on the actual attacks (16 classes).

Each of the two sets is comprised of a full subset (namely, AWID-ATK-F and AWID-CLS-F) and a reduced one (namely, AWID-ATK-R and AWID-CLS-R). The reduced versions are not derivatives of the full ones in any way whatsoever and do not contain artificial data. On the contrary, both these subsets were produced via live network utilization, over two distinct capturing sessions. The reduced subsets are better suited for the early stages of a researcher's experiments due to their smaller size and their ability to be analyzed efficiently by a single node. On the other hand, the full sets are significantly larger in size and reflect the need of modern wireless IDS to cope with large volume of data by possibly employing big data analysis techniques.

Finally, each subset has two versions; a training (AWID-ATK-F-Trn, AWID-CLS-F-Trn, AWID-ATK-R-Trn, AWID-CLS-R-Trn) and a testing one (AWID-ATK-F-Tst, AWID-CLS-F-Tst, AWID-ATK-R-Tst, AWID-CLS-R-Tst). The training versions are necessary for building models of "normal" and "abnormal" traffic during the learning phase.

Both the AWID-ATK-R-Trn and AWID-CLS-R-Trn contain 1,795,575 records. Out of that volume 1,633,190 is normal traffic and the rest of the dataset is comprised of records classified as intrusive (162,385 records). These datasets were generated by monitoring the test network for 1 hour, with the attack-free traffic spanning 45 minutes and the traffic that

contains attacks lasting for 15 minutes. This corresponds to a 3 to 2 ratio of normal to attack with respect to time invested. The raw (pcap) file occupies 948 MB on the disk while the corresponding extracted dataset is a single Comma Separated Values (CSV) file of 935 MB (or merely 68 MB if compressed with gzip). Likewise, the AWID-ATK-F-Trn and AWID-CLS-F-Trn contain 37,817,835 records, 1,085,372 of which are some kind of attack (in this dataset the normal to attack time ratio is 9:1). The dataset is spread over 96 files each containing (a variable number of) records that correspond to 1 hour of network monitoring. The accumulated size of the dataset is approximately 15 GB and it was produced from over 16.3 GB of raw information.

Table VI summarizes the main characteristics and file structure of the AWID collection of datasets including references to the test versions of the dataset. A snapshot of all the current and future versions will be made available (upon request) by visiting the online resources of the manuscript [63].

C. Composition of Dataset

This subsection decomposes the AWID-CLS-R-Trn and AWID-ATK-R-Trn subsets and analyzes their contents. Both AWID-CLS-R-Trn and AWID-ATK-R-Trn subsets were generated by 1 hour utilization of the test network. In our experiments the attack-free traffic lasted 35 minutes (60% of the time) while the rest 25 minutes (40% of the time) were dedicated to exploiting vulnerabilities of the test network. During the attack free traffic the users of the network were conducting ordinary activities such as file transfers, web browsing and video streaming or for some periods of time the network was dormant. During the attack window a single node unleashed a set of 15 unique attacks and multiple variations of them in a non-random way, i.e., she sequentially executed attacks with the necessary order to achieve a specific task such as cracking the network key. In more detail, in the AWID-CLS-R-Trn and AWID-ATK-R-Trn subsets 54.5% of the attack time was dedicated to injection attacks, 18.5% to flooding attacks and 26.8% to impersonation attacks. The AWID-CLS-R-Trn and AWID-ATK-R-Trn subsets include: Fragmentation and ARP Injection attacks, Deauthentication, Authorization Request, Beacon, Probe Response flooding attacks as well as Evil Twin and Caffe Latte impersonation attacks. On the other hand, the AWID-CLS-R-Tst and AWID-ATK-R-Tst contains all the aforementioned attacks plus: ChopChop, CTS, RTS, Disassociation, Power Saving, Probe Request and Hirte attacks, which are considered novel by the standards of the training set. The full versions of the dataset, namely AWID-CLS-R-Trn and AWID-ATK-R-Trn are based on similar principles.

Take into account that the percentages above refer to time durations and do not correspond to actual number of records. For instance, flooding attacks occupied 18.5% of the attack time (or 7.5% of the total experiment time) in the AWID-CLS-R-Trn set. During this duration 48,484 malicious packets were introduced which may be as high as 29.8% of the entire attack traffic but at the same time it is just 2.7% of the sum of traffic in that set. The type of attacks included in the training and test versions of the AWID dataset along with the corresponding normal to attack traffic ratio with respect to time as well as traffic, are illustrated in figure 13 and 14. Moreover, the complete sequence of intrusive events, and their duration through time is illustrated in figure 12.

D. Record Scheme

Each packet in the dataset is represented as a vector of 156 attributes, with the last attribute being the corresponding class (for AWID-CLS-R-Trn, AWID-CLS-R-Tst) or the corresponding attack (for AWID-ATK-R-Trn and AWID-ATK-R-Tst) of the record. The set of attributes is static which means that a packet is described by the same number of attributes independently of its type and subtype. For creating a well defined dataset we were indebted to construct a rather verbose universal 802.11 frame type. This theoretic representation of a frame bears (almost) all possible 802.11 fields and therefore the values of -1 or "-" were assigned to the fields that do not apply to a specific header type. Note that the actual data field was considered irrelevant and was not included in the dataset. Moreover, extremely rare fields (such as vendor depended ones) or custom tags were filtered out beforehand.

Each record is composed mainly by MAC layer information. Such attributes include: Source Address (wlan_sa), Destination Address (wlan_da), Initialization Vector (wlan_wep_iv), the ESSID (wlan_mgt_ssid) and others. Additionally, all Radiotap information such as the Signal Strength (radiotap_dbm_antsignal), as well as general frame information such as Packet Number (frame_number) are included in each record.

All attributes in the dataset have numeric or nominal values except for the SSID value which takes string values. Hexadecimal values or fields that represent MAC addresses were transformed to their corresponding integer values on a separate preprocessing step. A typical MAC address corresponds to an integer value of 82468889197, while a typical value of signal strength field is -33. It is obvious that the scales of the attributes on the dataset are heavily imbalanced. We argue that a normalization step would be beneficial prior applying any kind of ML algorithm to the dataset.

VII. Dataset Evaluation

A. Machine Learning Classification

We evaluated the AWID dataset against several soft computing algorithms in an attempt to give pointers towards the algorithms that behave best with the AWID dataset. Our experiments were conducted with the Weka [64] framework on an 8-core, Ubuntu 12.04 server, virtual machine with 56 GB RAM, located on the Azure cloud service. The chosen datasets were AWID-CLS-R-Trn and AWID-CLS-R-Tst for training and testing purposes respectively. A summary of the results is shown in Table VII.

The J48 algorithm achieved the optimal FP, TP rates but on the downside it required dramatically more time than any other method to construct its model from the training data (3921.68 seconds). The Random Forest and OneR achieved the second best TP and FP rate respectively, both an order of magnitude faster.

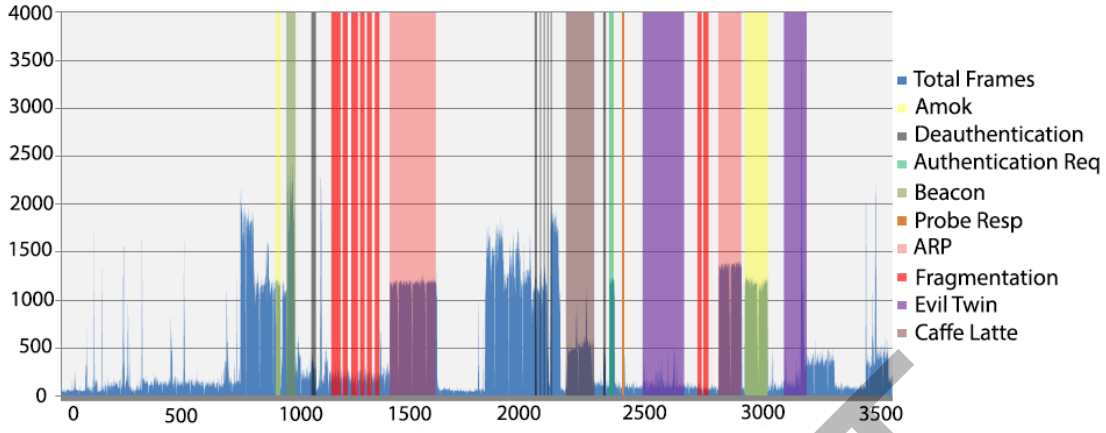
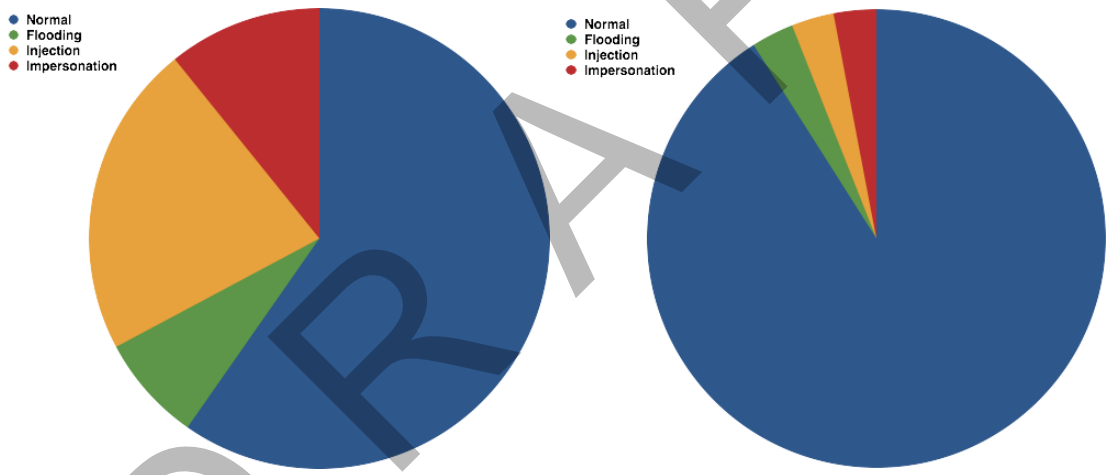
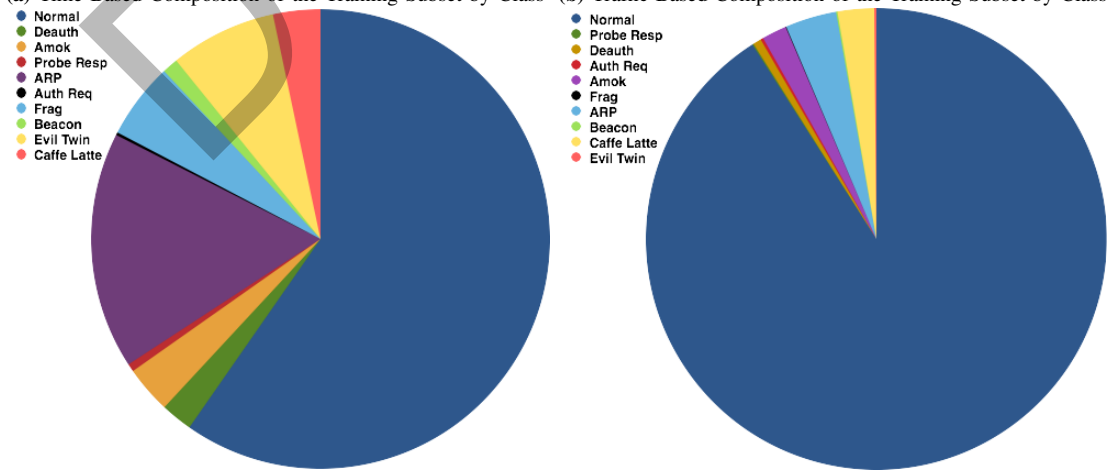


Fig. 12: Sequence of Attacks in the Training Set



(a) Time Based Composition of the Training Subset by Class **(b)** Traffic Based Composition of the Training Subset by Class



(c) Time Based Composition of the Training Subset by Attack **(d)** Traffic Based Composition of the Training Subset by Attack

Fig. 13: Attack vs. Normal Traffic in the Reduced Training sets

TABLE VI: File Structure of the AWID Collection

Filename refers to the code name of the CSV dataset file, *Classes* refers to the number of classes contained in that particular version of the dataset, *Size* refers to whether the version of the dataset is full or reduced, *Type* refers to whether the version of the dataset is meant for training or testing purposes, *Hours* refers to the number of hours invested in monitoring the network to produce the dataset, *Total Recs* refers to the number of records in the file, *Normal Recs* refers to the normal records in the file, *Attack Recs* refers to the attack records in the file, *Ratio* refers to time analogy of normal to attack traffic in the file, *Filesize* refers to the size of the resulting CSV file in MB, *Compressed* refers to the size of the CSV file compressed with gzip in MB, *Raw* refers to the size of the source PCAP file in MB.

Filename	Classes	Size	Type	Hours	Total Recs	Normal Recs	Attack Recs	Ratio	Filesize	No Files	Raw
AWID-CLS-F-Trn	4	Full	Training	96	37817835	36732463	1085372	9:1	15100	96	16300
AWID-CLS-F-Tst	4	Full	Test	12	4570463	4373934	196529	9:1	1700	12	1900
AWID-CLS-R-Trn	4	Reduced	Training	1	1795575	1633190	162385	3:2	935	1	948
AWID-CLS-R-Tst	4	Reduced	Test	1/3	575643	530785	44858	3:2	297	1	318
AWID-ATK-F-Trn	16	Full	Training	96	37817835	36732463	1085372	9:1	15100	96	16300
AWID-ATK-F-Tst	16	Full	Test	12	4570463	4373934	196529	9:1	1700	12	1900
AWID-ATK-R-Trn	16	Reduced	Training	1	1795575	1633190	162385	3:2	935	1	948
AWID-ATK-R-Tst	16	Reduced	Test	1/3	575643	530785	44858	3:2	297	1	318

At a first glance, algorithms such as Adaboost, Hyperpipes, ZeroR which present TP rate of 0.922 may seem satisfactory. This illusion is created by the fact that the portion of normal traffic greatly outnumbers the abnormal one. In practice, what these algorithms do is incorrectly assign any record to the normal class, thus achieving TP rates equal to the records of that class. Nevertheless, a closer look at the comparative confusion matrices in Table VIII can verify that these algorithms misclassify all intrusive records. As far as separate classes are concerned the normal class was almost infallibly predicted by the OneR algorithm in 99.99% rate. Naive Bayes managed to correctly recognize 72.69% of the flooding records. The injection class was very accurately predicted by the J48 algorithm (99.98%) but records of impersonation class were the toughest to recognize with the top performer for the class, namely Random Tree, being able to correctly classify only 7.5% of these records.

B. Manual Attribute Selection Based on Theoretic Criteria

Out of the 156 attributes contained in the dataset it is inevitable that some of them play a less important role, others are insignificant, while in the worst case, a number of attributes may act as noise and therefore misdirect the machine learning process. This pleonasm is expected to have a negative impact on the training speed and it is highly probable that it will lead to lower detection rates. We assert that the process of locating and eliminating the redundant attributes is a necessary step before moving on to any form of automatic intrusion detection, and as explained further down, it will be of benefit when applied to the AWID dataset too. The work in [65] stresses the importance of feature/attribute selection in wireless intrusion detection and ranks the most important features/attributes of the MAC header.

As detailed in section V most attacks have inherent characteristics that attest their presence. For example, the fragmentation attack is based on the injection of a large number of frames that have the fragment number field greater than 0. Moreover, besides the theoretic attack signatures most penetration tools leave digital footprints of their own. A vivid example is that of the Authentication Request attack which is implemented solely by the MDK3 tool. During this mode of execution the tool generates a flood of frames of subtype 0x0b, that all have Sequence Number field of value 0. This would strike as a statistical anomaly as it is impossible to locate sequential records with such characteristics on the normal class.

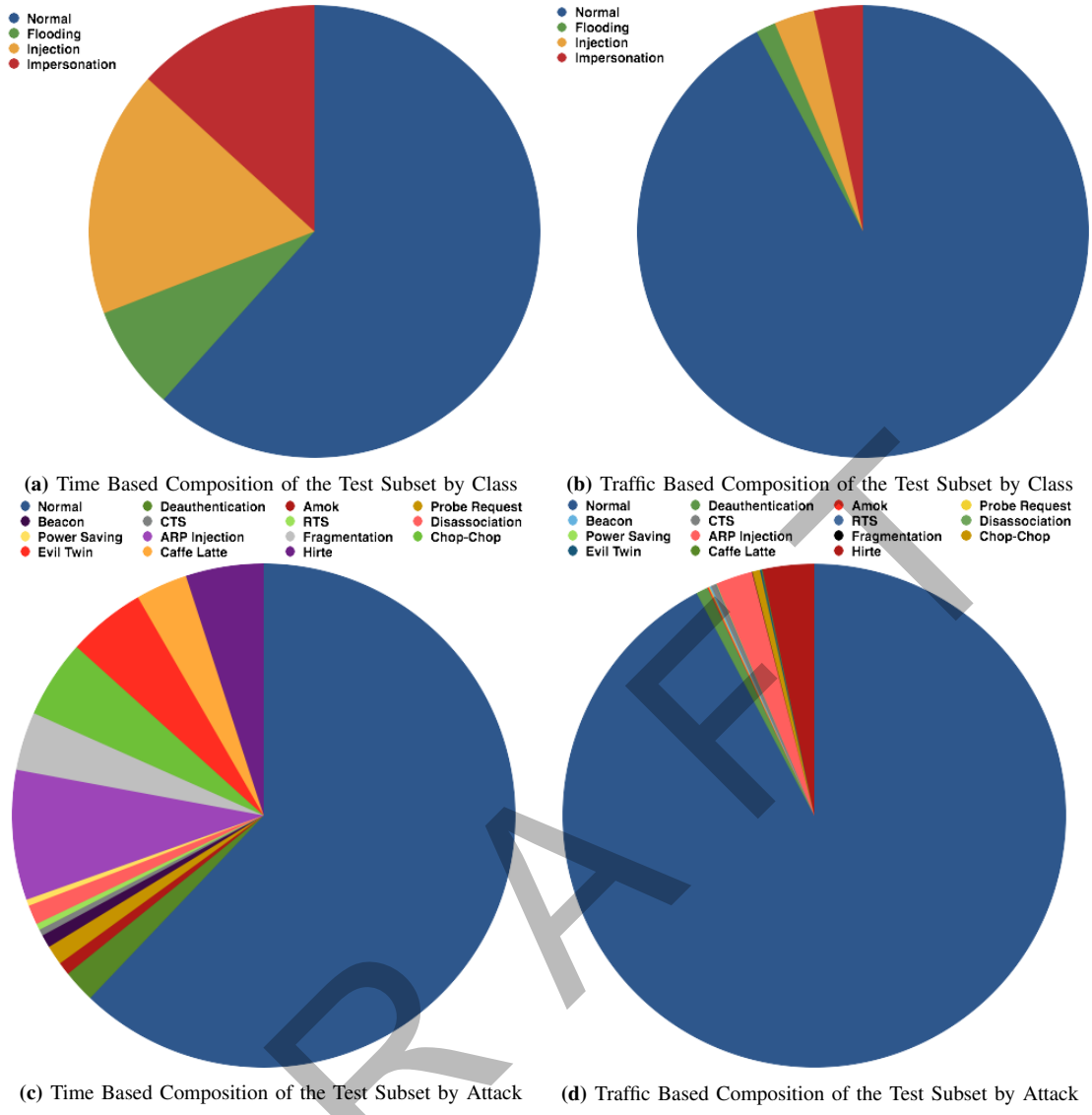
After thoroughly examining the fields of intrusive frames we attempted manual attribute selection. Based on the observations and theoretical analysis of the attack patterns we deduced that only 20 attributes play a crucial role when it comes to identifying attacks, so we moved to a second round of the previously described experiments. The results of the evaluation which can be seen in Table IX indicate that while there was a small (and in many cases negligible) increase in the overall accuracy, there was a definite boost in the training speed of almost all algorithms. More specifically, the shrinkage in training time varied from 10.75% for the Random Forest algorithm to as high as 89.35% for the Adaboost algorithm. The only exception to this rule was the ZeroR algorithm which actually required more time (0.63 seconds with the 156-feature sets vs. 3.65 seconds with the 20-feature sets).

Likewise, we observed an augmentation in the classification precision which unfortunately was of milder intensity. More specifically, the achieved improvement ranged from 0.1% for the Random Forest algorithm to 4.5% for the Random Tree one. Despite the overall increase in precision, only the Naive Bayes algorithm showcased a significant one in prediction accuracy of the impersonation attacks, but with 4,419 correctly classified out of 20,079 class instances (22%) the achievements are still unsatisfactory.

In this round of experiments the Naive Bayes algorithm had the lowest FP rate, the J48 achieved the highest TP rate while Random Forest had the best accuracy for the Normal class, Naive Bayes for the Flooding and Impersonation classes, and J48 for the Injection one.

VIII. Conclusions and Discussion

This work gathers, describes and evaluates the most widespread attacks in the IEEE 802.11 standard. Based on real-life network utilization we have compiled the AWID family of datasets that contain both normal and abnormal traffic. The results of the evaluation have aided maintaining the contents of the dataset as realistic as possible in matters of attack sequence and normal to attack traffic ratio. The dataset



(a) Time Based Composition of the Test Subset by Class

(b) Traffic Based Composition of the Test Subset by Class

(c) Time Based Composition of the Test Subset by Attack

(d) Traffic Based Composition of the Test Subset by Attack

Fig. 14: Attack vs. Normal Traffic in the Reduced Testing Sets

TABLE VII: Evaluation of Various Classification Algorithms on the 156 Feature Set. Best performer in red.

Algorithm	Correctly Classified%	Incorrectly Classified%	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Time
AdaBoost	92.2073	7.7927	0.922	0.922	0.85	0.922	0.885	0.806	1513
Hyperpipes	92.2073	7.7927	0.922	0.922	0.85	0.922	0.885	0.952	7.79
J48	96.1982	3.801	0.962	0.437	0.954	0.962	0.948	0.759	3921.68
Naive Bayes	89.4323	10.5677	0.894	0.768	0.891	0.894	0.877	0.594	188.21
OneR	94.5758	5.4242	0.946	0.642	0.9	0.946	0.922	0.652	156.98
Random Forest	95.5891	4.4109	0.956	0.52	0.958	0.956	0.941	0.955	828.95
Random Tree	91.4379	8.5621	0.914	0.449	0.914	0.914	0.91	0.733	88.43
ZeroR	92.2073	7.7927	0.922	0.922	0.85	0.922	0.885	0.5	0.63

was harvested from a WEP protected network but we strongly assert that the same data is possible to be used for WPA/WPA2 intrusion detection, since the existing threats of the latter are practically a subset of first. With the same mindset, the AWID dataset is possible to be utilized for experimentation with alternative wireless technologies (WiMAX, UMTS, LTE) as they are proved to have similar security holes and their attacks follow resembling patterns [66] and [67].

Our experiments indicated an inherent advantage of the Random Forest and J48 classification algorithms. This conclusion may pave a walkway for the development of robust detection algorithms for the specific dataset in the future.

After carefully examining the traffic patterns we concluded that theoretically, out of the 156 attributes chosen to be included in the dataset, only 20 seem to play a more significant role in legacy intrusion detection. This hypothesis was indeed confirmed by the results of several tests which exhibited a slight increase in detection rate, the overall accuracy of most classifiers as well as a definite increase in the learning speed.

These experiments were not exhaustive and numerous promising machine learning techniques were left for future research. In particular, Swarm Intelligence techniques [68],

TABLE VIII: Confusion Matrices of Various Classification Algorithms on the 156 Feature Set. Best performer in red.

Normal	Flooding	Injection	Impersonation	Classified As
530785	0	0	0	Normal
8097	0	0	0	Flooding
16682	0	0	0	Injection
20079	0	0	0	Impersonation

(a) Adaboost

Normal	Flooding	Injection	Impersonation	Classified As
530785	0	0	0	Normal
8097	0	0	0	Flooding
16682	0	0	0	Injection
20079	0	0	0	Impersonation

(b) Hyperpipes

Normal	Flooding	Injection	Impersonation	Classified As
530771	8	0	6	Normal
2641	4857	0	599	Flooding
2	0	16680	0	Injection
18629	0	0	1450	Impersonation

(c) J48

Normal	Flooding	Injection	Impersonation	Classified As
508621	22164	0	0	Normal
2189	5908	0	0	Flooding
16400	0	282	0	Injection
18750	1329	0	0	Impersonation

(d) Naive Bayes

Normal	Flooding	Injection	Impersonation	Classified As
530775	0	7	3	Normal
8097	0	0	0	Flooding
3038	0	13644	0	Injection
20079	0	0	0	Impersonation

(e) OneR

Normal	Flooding	Injection	Impersonation	Classified As
530729	1	54	1	Normal
4077	4020	0	0	Flooding
2470	0	14212	0	Injection
18760	0	28	1291	Impersonation

(f) Random Forest

Normal	Flooding	Injection	Impersonation	Classified As
518657	906	716	10506	Normal
3854	4243	0	0	Flooding
338	0	1930	14414	Injection
17550	0	1003	1526	Impersonation

(g) Random Tree

Normal	Flooding	Injection	Impersonation	Classified As
530785	0	0	0	Normal
8097	0	0	0	Flooding
16682	0	0	0	Injection
20079	0	0	0	Impersonation

(h) ZeroR

TABLE IX: Evaluation of Various Classification Algorithms on the 20 Feature Set. Best performer in red.

Algorithm	Correctly Classified%	Incorrectly Classified%	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Time
AdaBoost	92.2073	7.7927	0.922	0.922	0.85	0.922	0.885	0.673	161.12
Hyperpipes	92.2363	7.7637	0.922	0.919	0.879	0.922	0.885	0.935	3.52
J48	96.2574	3.7426	0.963	0.436	0.962	0.963	0.948	0.752	568.92
Naive Bayes	90.5504	9.4496	0.906	0.399	0.917	0.906	0.909	0.774	29.67
OneR	94.5741	5.4259	0.946	0.642	0.9	0.946	0.922	0.652	156.98
Random Forest	95.8247	4.1753	0.958	0.493	0.959	0.958	0.944	0.958	739.78
Random Tree	96.2258	3.7742	0.962	0.438	0.959	0.962	0.948	0.762	49.3
ZeroR	92.2073	7.7927	0.922	0.922	0.85	0.922	0.885	0.5	3.65

TABLE X: Confusion Matrices of Various Classification Algorithms on the 20 Feature Set. Best performer in red.

Normal	Flooding	Injection	Impersonation	Classified As
530785	0	0	0	Normal
8097	0	0	0	Flooding
16682	0	0	0	Injection
20079	0	0	0	Impersonation

(a) Adaboost

Normal	Flooding	Injection	Impersonation	Classified As
530785	0	0	0	Normal
8097	0	0	0	Flooding
16515	0	167	0	Injection
20079	0	0	0	Impersonation

(b) Hyperpipes

Normal	Flooding	Injection	Impersonation	Classified As
530588	116	6	75	Normal
2553	5544	0	0	Flooding
2	0	16680	0	Injection
18644	148	0	1287	Impersonation

(c) J48

Normal	Flooding	Injection	Impersonation	Classified As
497199	8971	11899	12716	Normal
2123	5974	0	0	Flooding
3027	0	13655	0	Injection
14187	1473	0	4419	Impersonation

(d) Naive Bayes

Normal	Flooding	Injection	Impersonation	Classified As
530765	0	14	6	Normal
8097	0	0	0	Flooding
3038	0	13644	0	Injection
20079	0	0	0	Impersonation

(e) OneR

Normal	Flooding	Injection	Impersonation	Classified As
530746	1	1	37	Normal
2600	5497	0	0	Flooding
2763	0	13893	0	Injection
18607	0	28	1472	Impersonation

(f) Random Forest

Normal	Flooding	Injection	Impersonation	Classified As
530700	3	0	82	Normal
2442	5494	161	0	Flooding
273	0	16253	156	Injection
18609	0	0	1470	Impersonation

(g) Random Tree

Normal	Flooding	Injection	Impersonation	Classified As
530785	0	0	0	Normal
8097	0	0	0	Flooding
16682	0	0	0	Injection
20079	0	0	0	Impersonation

(h) ZeroR

Markov Chain [69] based ones as well as other soft computing methods [70] seem to possess desirable characteristics in theory, and in practice have regularly led to the construction of robust IDS.

According to [71] big data are high volume, high velocity, and/or high variety information assets. Wireless network

communication satisfies all 3 characteristics of this model, commonly referred to as the "3V" and the structure of the AWID dataset reflects the first two aspects. That is, the AWID-CLS-F-Trn/AWID-ATK-F-Trn sets contain traffic collected in 4 days from a typical SOHO installation but analogous volume of traffic may be generated in matters of hours from large 3G/4G and beyond networks. It is apparent that learning speed, a commonly neglected aspect of IDS, should be taken into consideration and be optimized in such fashion that will fulfil the high requirements of the big data realm.

In addition to high analysis speeds we recognise 4 more factors that have a key role in the construction of efficient IDS for the wireless realm:

(a) Be extremely low on false positive alerts - due to the large volume of data, a FP rate of 1% may generate a great number of false alerts on daily basis.

(b) Be highly adaptive to drastic network behavioral changes - due to unpredicted events or natural changes in equipment, network behavior that once seemed normal may start looking suspicious. IDS that are adaptive guarantee that the requirement for re-training (if any) will be minimized.

(c) Be able to detect novel attacks - as wireless technologies mature, new vulnerabilities are discovered perennially, rendering the need for misuse detection a constant necessity.

(d) Have independence from supervised learning - labelling of data is a tedious process conducted manually by field experts. When datasets reach multigigabyte proportions this process renders is rendered impractical.

We aspire that the conclusions presented in this work along with the contribution of the AWID dataset, will significantly benefit research on the intrusion detection for wireless networks.

Regarding AWID, our short-term goals lie in (a) the creation of an even larger version of the dataset, involving several weeks of capture, (b) the construction of the equivalent version of the dataset for an EAP/TLS protected wireless network, as well as (c) an Ad-Hoc wireless network. Of course, AWID is committed into constantly enriching its database and expanding itself with each new versions of the protocol so the long-term plans involve the 802.11ac version of the standard.

REFERENCES

[1] Karl Bode. *Wireless Traffic to Reach 11.2 Exabytes a Month*. Nov. 2014. URL: http://www.dslreports.com/shownews/Cisco-Wireless-Traffic-to-Reach-112-Exabytes-a-Month-By-2017-123040.

[2] IEEE. *802.11-1997 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*. Nov. 2014. URL: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=654749.

[3] LIU Yong-leia, JIN Zhi-gangb, CHEN Zheb, and LIU Jing-weia. "Design and Implementation of High-speed Brute Forcer for wpa/wpa2-psk [J]". In: *Computer Engineering* 10 (2011), p. 044.

[4] Thoughcrime Labs. *CloudCracker*. Nov. 2014. URL: https://www.cloudcracker.com.

[5] IEEE. *802.11w-2009 - 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 Amendment 4: Protected Management Frames*. Nov. 2014. URL: http://standards.ieee.org/findstds/standard/802.11w-2009.html.

[6] Md Sohail Ahmad and Shashank Tadakamadla. "Short paper: security evaluation of IEEE 802.11 w specification". In: *Proceedings of the fourth ACM conference on Wireless network security*. ACM. 2011, pp. 53–58.

[7] *Aircrack-ng*. Nov. 2014. URL: http://www.aircrack-ng.org/.

[8] Changhua He John C Mitchell. "Security Analysis and Improvements for IEEE 802.11 i". In: *The 12th Annual Network and Distributed System Security Symposium (NDSS'05) Stanford University, Stanford*. Citeseer. 2005, pp. 90–110.

[9] Li Wang and Balasubramaniam Srinivasan. "Analysis and improvements over DoS attacks against IEEE 802.11 i standard". In: *Networks Security Wireless Communications and Trusted Computing (NSWCTC), 2010 Second International Conference on*. Vol. 2. IEEE. 2010, pp. 109–113.

[10] Cisco. *Cisco Adaptive Wireless IPS Software*. Nov. 2014. URL: http://www.cisco.com/en/US/prod/collateral/wireless/ps9733/ps9817/data_sheet_c78\-501388.html.

[11] Alexandros Tsakountakis, Georgios Kambourakis, and Stefanos Gritzalis. "Towards effective wireless intrusion detection in IEEE 802.11 i". In: *Security, Privacy and Trust in Pervasive and Ubiquitous Computing, 2007. SECPerU 2007. Third International Workshop on*. IEEE. 2007, pp. 37–42.

[12] Nikita Borisov, Ian Goldberg, and David Wagner. "Intercepting mobile communications: the insecurity of 802.11". In: *Proceedings of the 7th annual international*

conference on Mobile computing and networking. ACM. 2001, pp. 180–189.

[13] Kwang-Hyun Baek, Sean W Smith, and David Kotz. "A Survey of WPA and 802.11 i RSN Authentication Protocols". In: *Dartmouth Computer Science Technical Report2004* (2004).

[14] Richard Gass, James Scott, and Christophe Diot. "Measurements of in-motion 802.11 networking". In: *Mobile Computing Systems and Applications, 2006. WMCSA'06. Proceedings. 7th IEEE Workshop on*. IEEE. 2005, pp. 69–74.

[15] *The KDD99 Dataset*. Nov. 2014. URL: http://kdd.ics.uci.edu/databases/kddcup99/task.html.

[16] IEEE. *802.16e-2005, I.S. IEEE Standard for Local and Metropolitan area networks Part 16: Air Interface for Fixed Broadband Wireless Access Systems*. Nov. 2014. URL: http://standards.ieee.org/getieee802/download/802.16e-2005.pdf.

[17] ETSI. *Universal Mobile Telecommunications System (UMTS); User Equipment (UE) radio transmission and reception (FDD)*. Nov. 2014. URL: http://www.3gpp.org/specifications/79-specification-numbering.

[18] 3GPP. *LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) radio transmission and reception (3GPP TS 36.101 version 10.3.0 Release 10)*. Nov. 2014. URL: http://www.3gpp.org/DynaReport/36-series.htm.

[19] Rodrigo do Carmo and Matthias Hollick. "DogoIDS: a mobile and active intrusion detection system for IEEE 802.11 s wireless mesh networks". In: *Proceedings of the 2nd ACM workshop on Hot topics on wireless network security and privacy*. ACM. 2013, pp. 13–18.

[20] Nikita Lyamin, Alexey Vinel, Magnus Jonsson, and Jonathan Loo. "Real-Time Detection of Denial-of-Service Attacks in IEEE 802.11 p Vehicular Networks". In: *IEEE communications letters* 18.1 (2014), pp. 110–113.

[21] IEEE. *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. Amendment 4: Enhancements for Very High Throughput for Operation in Bands below 6 GHz*. Nov. 2014. URL: http://standards.ieee.org/getieee802/download/802.11ac-2013.pdf.

[22] L Devi and A Suganthi. "Denial of Service Attacks in Wireless Networks: The Case of Jammers". In: (2014).

[23] Scott Fluhrer, Itsik Mantin, and Adi Shamir. "Weaknesses in the key scheduling algorithm of RC4". In: *Selected areas in cryptography*. Springer. 2001, pp. 1–24.

[24] Andrea Bittau. "Additional weak IV classes for the FMS attack". In: *Department of Computer Science, University College London* (2003).

[25] Hal Berghel and Jacob Uecker. "WiFi attack vectors". In: *Communications of the ACM* 48.8 (2005), pp. 21–28.

[26] Rafik Chaabouni. *Break wep faster with statistical analysis*. Tech. rep. 2006.

[27] Erik Tews, Ralf-Philipp Weinmann, and Andrei Pyshkin. "Breaking 104 bit WEP in less than 60 seconds". In: *Information Security Applications*. Springer, 2007, pp. 188–202.

[28] Andreas Klein. "Attacks on the RC4 stream cipher". In: *Designs, Codes and Cryptography* 48.3 (2008), pp. 269–286.

[29] Robert Moskowitz. "Weakness in passphrase choice in WPA interface". In: *ICSA Labs* (2003).

[30] Yonglei Liu, Zhigang Jin, and Ying Wang. "Survey on security scheme and attacking methods of WPA/WPA2". In: *Wireless Communications Networking and Mobile Computing (WiCOM), 2010 6th International Conference on*. IEEE. 2010, pp. 1–4.

[31] KoreK. *ChopChop (Experimental WEP attacks)*. Nov. 2014. URL: http://www.netstumbler.org/showthread.php?t=12489.

[32] Andrea Bittau. "The Fragmentation Attack in Practice". In: *IEEE Symposium on Security and Privacy, IEEE Computer Society*. 2005.

[33] Md Sohail Ahmad and Vivek Ramachandran. *Cafe latte with a free topping of cracked wep retrieving wep keys from road warriors*. 2007.

[34] *Hirte Attack*. Nov. 2014. URL: http://www.aircrack-ng.org/doku.php?id=airbase-ng#hirte_attack_in_access_point_mode.

[35] IEEE. *802.11n-2009 - IEEE Standard for Information Technology- Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements. Part 11: Wireles LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. Amendment 5: Enhancements for Higher Throughput.* Nov. 2014. URL: http://standards.ieee.org/getieee802/download/802.11n-2009.pdf.

[36] Kemal Bicakci and Bulent Tavli. "Denial-of-Service attacks and countermeasures in IEEE 802.11 wireless networks". In: *Computer Standards & Interfaces* 31.5 (2009), pp. 931–941.

[37] Thuc D Nguyen, D Nguyen, Bao N Tran, Hai Vu, and Neeraj Mittal. "A lightweight solution for defending against deauthentication/disassociation attacks on 802.11 networks". In: *Computer Communications and Networks, 2008. ICCCN'08. Proceedings of 17th International Conference on*. IEEE. 2008, pp. 1–6.

[38] Ilenia Tinnirello and Sunghyun Choi. "Efficiency analysis of burst transmissions with block ACK in contention-based 802.11 e WLANs". In: *Communications, 2005. ICC 2005. 2005 IEEE International Conference on*. Vol. 5. IEEE. 2005, pp. 3455–3460.

[39] *Review of A MPDU DoS Issue,* Nov. 2014. URL: https://mentor.ieee.org/802.11/file/07/11-07-2163-01-000n-a-mpdu-security-issues.ppt.

[40] Chibiao Liu, James Yu, and Gregory Brewster. "Empirical studies and queuing modeling of denial of service attacks against 802.11 WLANs". In: *World of Wireless Mobile and Multimedia Networks (WoWMoM), 2010 IEEE International Symposium on a*. IEEE. 2010, pp. 1–9.

[41] Leandro F Meiners. "But... my station is awake! Power Save Denial of Service in 802.11 Networks". In: (2009).

[42] Wenjun Gu, Zhimin Yang, Dong Xuan, Weijia Jia, and Can Que. "Null data frame: A double-edged sword in IEEE 802.11 WLANs". In: *Parallel and Distributed Systems, IEEE Transactions on* 21.7 (2010), pp. 897–910.

[43] Mina Malekzadeh, Abdul AA Ghani, Jalil Desa, and Shamala Subramaniam. "Empirical analysis of virtual carrier sense flooding attacks over wireless local area network". In: *Journal of Computer science* 5.3 (2009), p. 214.

[44] Supriya S Sawwashere and Sonali U Nimbhorkar. "Survey of RTS-CTS Attacks in Wireless Network". In: *Communication Systems and Network Technologies (CSNT), 2014 Fourth International Conference on*. IEEE. 2014, pp. 752–755.

[45] Asier Martínez, Urko Zurutuza, Roberto Uribeetxeberria, Miguel Fernández, Jesus Lizarraga, Ainhoa Serna, and Iñaki Vélez. "Beacon Frame Spoofing Attack Detection in IEEE 802.11 Networks". In: *Availability, Reliability and Security, 2008. ARES 08. Third International Conference on*. IEEE. 2008, pp. 520–525.

[46] Francesco Ferreri, Massimo Bernaschi, and Leonardo Valcamonici. "Access points vulnerabilities to DoS attacks in 802.11 networks." In: *WCNC*. 2004, pp. 634–638.

[47] Noor Al-Gharabally, Nosayba El-Sayed, Sara Al-Mulla, and Imtiaz Ahmad. "Wireless honeypots: survey and assessment". In: *Proceedings of the 2009 conference on Information Science, Technology and Applications*. ACM. 2009, pp. 45–52.

[48] Yimin Song, Chao Yang, and Guofei Gu. "Who is peeping at your passwords at Starbucks?To catch an evil twin access point". In: *Dependable Systems and Networks (DSN), 2010 IEEE/IFIP International Conference on*. IEEE. 2010, pp. 323–332.

[49] Raheem Beyah and Aravind Venkataraman. "Rogue-access-point detection: Challenges, solutions, and future directions". In: *IEEE Security and Privacy* 9.5 (2011), pp. 56–61.

[50] Ajay M Patel, Ashok R Patel, and Hiral R Patel. "RAP Problems and Solutions in 802.11 Wireless LAN". In: *International Journal* 2.1 (2014).

[51] Vivek Ramachandran. *Backtrack 5 Wireless Penetration Testing: Beginner's Guide*. Packt Publishing Ltd, 2011.

[52] *MDK3*. Nov. 2014. URL: http : / / homepages . tu - darmstadt.de/~p_larbig/wlan/\#mdk3.

[53] *File2air*. Nov. 2014. URL: http://www.willhackforsushi.com/?page_id=19.

[54] *Lorcon-old source code*. Nov. 2014. URL: https://aur.archlinux.org/packages/lorcon-old-git/?setlang=en.

[55] *Lorcon2 Library*. Nov. 2014. URL: https://code.google.com/p/lorcon/.

[56] John Bellardo and Stefan Savage. "802.11 Denial-of-Service Attacks: Real Vulnerabilities and Practical Solutions." In: *USENIX security*. 2003, pp. 15–28.

[57] L Arockiam, B Vani, S Sivagowry, and A Persia. "A solution to prevent resource flooding attacks in 802.11 WLAN". In: *Global Trends in Computing and Communication Systems*. Springer, 2012, pp. 607–616.

[58] Harold Gonzales, Kevin Bauer, Janne Lindqvist, Damon McCoy, and Douglas Sicker. "Practical defenses for evil twin attacks in 802.11". In: *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*. IEEE. 2010, pp. 1–6.

[59] *Metasploit*. Nov. 2014. URL: http://www.metasploit.com/.

[60] *Wireshark*. Nov. 2014. URL: http://www.wireshark.org.

[61] AWID. *capt - Dataset Capturing Script*. Nov. 2014. URL: www.icsd.aegean.gr/awid/downloads.

[62] AWID. *pcap2csv - PCAP to CSV Transformation Script*. Nov. 2014. URL: www.icsd.aegean.gr/awid/downloads.

[63] AWID. *Aegean Wireless Intrusion Dataset*. Nov. 2014. URL: www.icsd.aegean.gr/awid/downloads.

[64] University of Waikato. *Weka 3: Data Mining Software in Java*. Nov. 2014. URL: http://www.cs.waikato.ac.nz/ml/weka/.

[65] N Pratik Neelakantan and C Nagesh. "Role of Feature Selection in Intrusion Detection Systems for 802.11 Networks". In: *International Journal of Smart Sensors and Ad Hoc Networks (IJSSAN) Volume-1, Issue-1* (2011), pp. 98–101.

[66] Georgios Kambourakis, Constantinos Kolias, Stefanos Gritzalis, and Jong Hyuk Park. "DoS attacks exploiting signaling in UMTS and IMS". In: *Computer Communications* 34.3 (2011), pp. 226–235.

[67] Constantinos Kolias, Georgios Kambourakis, and Stefanos Gritzalis. "Attacks and countermeasures on 802.16: analysis and assessment". In: *Communications Surveys & Tutorials, IEEE* 15.1 (2013), pp. 487–514.

[68] Constantinos Kolias, Georgios Kambourakis, and M Maragoudakis. "Swarm intelligence in intrusion detection: A survey". In: *computers & security* 30.8 (2011), pp. 625–642.

[69] Jiankun Hu, Xinghuo Yu, Dong Qiu, and Hsiao-Hwa Chen. "A simple and efficient hidden Markov model scheme for host-based anomaly intrusion detection". In: *Network, IEEE* 23.1 (2009), pp. 42–47.

[70] Shelly Xiaonan Wu and Wolfgang Banzhaf. "The use of computational intelligence in intrusion detection systems: A review". In: *Applied Soft Computing* 10.1 (2010), pp. 1–35.

[71] Mark A Beyer and Douglas Laney. "The importance of'big data': a definition". In: *Stamford, CT: Gartner* (2012).