AN ENSEMBLE LEARNING BASED MULTI-LEVEL NETWORK INTRUSION DETECTION SYSTEM FOR WI-FI DOMINANT NETWORKS

A Thesis

Submitted to the Faculty

of

Purdue University

by

Francisco D. Vaca

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science

May 2019

Purdue University

Hammond, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF THESIS APPROVAL

Dr. Quamar Niyaz

Committee Chair, Department of Electrical and Computer Engineering

Dr. Xiaoli Yang

Committee Member, Department of Electrical and Computer Engineering

Dr. Bin Chen

Committee Member, Department of Electrical and Computer Engineering

Dr. Vijay Devabhaktuni

Department Chair, Department of Electrical and Computer Engineering

Approved by:

Dr. Xiaoli Yang

Head of the Graduate Program

ACKNOWLEDGMENTS

I would like to acknowledge the assistance of several individuals whose support contributed throughout the development of this work. First, I would like to acknowledge the support of my family: my wife, my children, my parents, and siblings. They have always motivated to continue working until the end. I would also like to acknowledge the assistance of my thesis advisor, Dr. Quamar Niyaz. I took several courses offered by Dr. Niyaz during my graduate studies. He always made an effort to help me apply the teaching from these courses in my thesis research. I acknowledge the assistance and directions from Professor Lucy (Xiaoli) Yang since the start of my graduate studies. Dr. Yang always showed the disposition to provide guidance when I started my research work until the finishing of it. I acknowledge the participation of my classmate, Ulises Morales who helped me to conduct the experiments presented in this work. I especially acknowledge the support of the Department of Electrical and Computer Engineering for the Teaching Assistantship support in my first semester of graduate studies. Moreover, the department provided funding for the materials needed for the experiments. I express my most sincere appreciation to all the mentioned individuals for being part of this research contribution.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

ABSTRACT

Vaca, Francisco D. MS, Purdue University, May 2019. An Ensemble Learning Based Multi-level Network Intrusion Detection System for Wi-Fi Dominant Networks. Major Professor: Quamar Niyaz.

Today, networks contribute significantly to everyone's life. The enormous usefulness of networks for various services and data storage motivates adversaries to launch attacks on them. Network Intrusion Detection Systems (NIDSs) are used as security measure inside the organizational networks to identify any intrusions and generate alerts for them. The idea of deploying an NIDS is quite known and has been studied and adopted in both academia and industry. However, most of the NIDS literature have emphasized to detect the attacks that originate externally in a wired network infrastructure. In addition, Wi-Fi and wired networks are treated the same for the NIDSs. The open infrastructure in Wi-Fi network makes it different from the wired network. Several internal attacks that could happen in a Wi-Fi network are not possible in a wired network. The NIDSs developed using traditional approaches may fail to identify these internal attacks.

The thesis work attempts to develop a Multi-Level Network Intrusion Detection System (ML-NIDS) for Wi-Fi dominant networks that can detect internal attacks specific to Wi-Fi networks as well as the generic network attacks that are independent of network infrastructure. In Wi-Fi dominant networks, Wi-Fi devices (stations) are prevalent at the edge of campus and enterprise networks and integrated with the fixed wired infrastructure at the access. The implementation is proposed for Wi-Fi dominant networks; nevertheless, it aims to work for the wired network as well. We develop the ML-NIDS using an ensemble learning method that combines several weak learners to create a strong learner.

1. INTRODUCTION

When discussing modern technology, the subject dominating conversations is the Internet and its continual progress. More specifically, how easy it has become to obtain access to it. This is the reason why the number of the Internet users has spiked up. Moreover, the Internet applications in various industries have increased as well providing a wider range of services. As a result, considerable amount of information are to the disposition of the users. Nevertheless, this information is available as well to the skilled network adversaries.

1.1 Background and Motivation

The 2018 Global Digital suite has reported that the number of the Internet users has reached 4 billion as shown in Figure 1.1 [1]. It is also revealed that only in 2017, the total number of people having interaction with the Internet for the first time was a quarter of a billion with Africa being the location with fastest growing pace by increasing 20% each year. The same source explains that the contribution to this accelerated number of people accessing the Internet can be attributed to the facilities now at the disposition to a large number of customers in relation to more affordable devices and mobile data plans. Another impressive statistic shared in this report is that more than 200 million acquired their first mobile device only in the course of the year 2017, and that now two-thirds of the inhabitants of the world have a mobile phone. From all these devices owned by two-thirds of inhabitants of the world, around half of them are considered smart, which only means they have access to content provided by the Internet.

Another factor that contributes to the increase of the Internet usage is the growth of social media. This multiplication of social media users can be appreciated by the

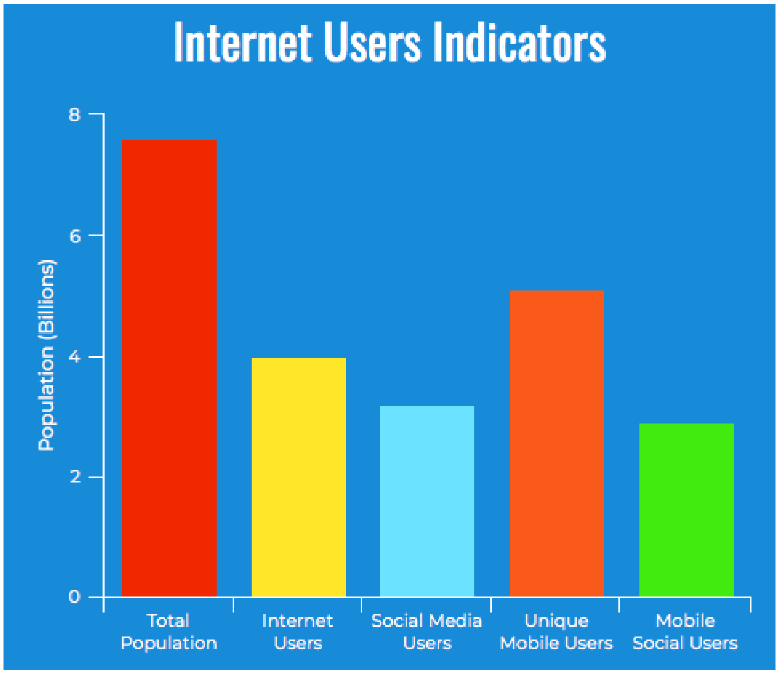


Fig. 1.1. Internet and Social Media Accessibility Statistics [1]

revealed number of new users using the top platform in each country, which is reported 1 million. An important point made in this report is that the Internet accessibility is not the only thing increasing, but the time spend on it has spiked up in the past couple of years showing that the average time that a regular Internet user spends on it is about 6 hours per day. However, there is another technology that plays a principal role in facilitating Internet connectivity. The technology that has demonstrated to be a slingshot for spreading Internet in terms of availability is wireless Internet access.

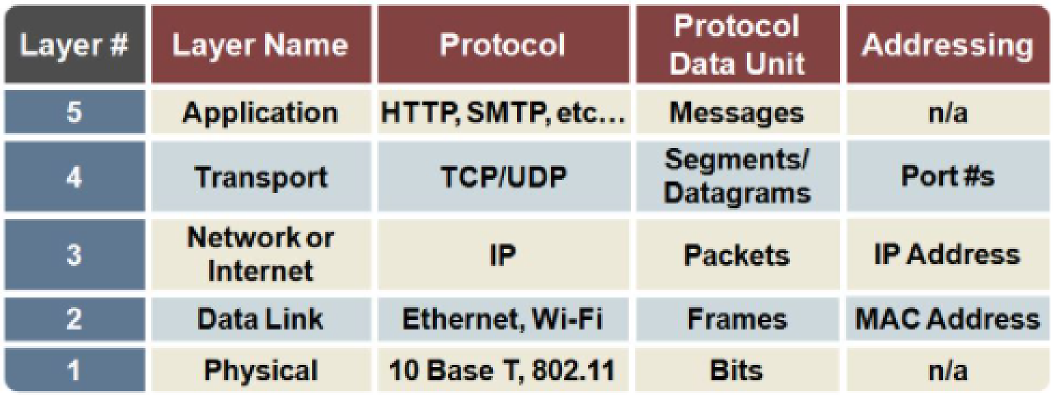
Linksys described in a report that how directly the Internet accessibility is related with wireless network availability [2]; for example, the average number of devices connected to the Internet at homes is 8, with 84% of users simultaneously performing video and music streaming, online gaming, to mention a few of these activities requiring Internet connection. Furthermore, among the many wireless technologies, it is interesting to see how fast Wi-Fi has spread either for personal or enterprise use. In the last few years, it has been impressive the pace at which the demands of users have been efficiently satisfied and even surpassed expectations in regards of communication, more specifically data transmission. It is critical for researchers to focus in the development of a mechanism to enforce protection against the ever-evolving network threats. Countless ongoing efforts address this issue and researchers speed up their pace to ensure protection in data transmission; however, it is imperative to acknowledge that on the other side of the subject, intruders constantly discover weaknesses in the complex structure of the currently most popular means of communication, the Internet.

As technology takes colossal steps in making available efficient connections for data transmission, it is necessary to emulate the same progress in improving the safety of this data, which gradually gains more and more value. Users are coming to the understanding of the importance of data. This understanding is obtained as users realize that there are so many processes that generate great amounts of data. Some process may handle data that seems useless, but some process manipulate highly sensitive data that is very appealing for attackers to obtain illegally. It is important to raise awareness of how much data is being created to understand well how valuable it is. Bernard Marr, a Forbes editor, wrote about the startling statistics regarding generation of data [3]. He stated the fact that the rate of data creation is 2.5 quintillion bytes per day, and the pace is continuing to accelerate, particularly with the advancements of the Internet of things (IoT). What concerns those who are aware of the value that data is gaining is that from all the data available, 90% of it was generated in the last two years. It is safe to assume that if the amount of existing data in general is increasing abruptly, the amount of sensitive data is increasing as well, therefore creating the need to increase the level of security to maintain it safe.

Social media plays an important role as well with videos, pictures, posts, comments, account creation, chats among many other sources of data being uploaded, posted or shared by the hundreds of millions by well known social network sites such as Twitter, Facebook, and Instagram. Every minute 16 million text messages, 156 million emails, 15,000 GIFs are sent through Facebook Messenger, 103,447,520 spam emails, and 152,200 calls on Skype are generated. It is frightening to think that so much data, which certainly includes sensitive data, is available and stored somewhere.

1.2 The Internet Stack

To demonstrate the risks involved on blindly putting our trust in modern data transmission and management, it needs to be exposed the Internet protocol stack, which depicts the layers that describe the phases through which the data traverses between the communicating parties. Each of these layers fulfills its own purpose and integrates with the others to provide the data transmission service. In Figure 1.2, each layer could be explored as its own world; nevertheless, the purpose of this work is to expose their main role in the process of exchanging information, their characteristics, and the weaknesses that are commonly targeted by attackers. At the highest level of the stack, Application layer which interacts directly with the users and programs. Right below appears the Transport layer that offers services to network applications. Network layer, which makes possible communication between networks by facilitating network addressing and routing. Data Link layer, which is responsible of data transmission between devices belonging to the same network. Finally, Physical layer that refers to the actual hardware employed for the transmission. Knowing the function-



Layer #	Layer Name	Protocol	Protocol Data Unit	Addressing
5	Application	HTTP, SMTP, etc...	Messages	n/a
4	Transport	TCP/UDP	Segments/ Datagrams	Port #s
3	Network or Internet	IP	Packets	IP Address
2	Data Link	Ethernet, Wi-Fi	Frames	MAC Address
1	Physical	10 Base T, 802.11	Bits	n/a

Fig. 1.2. A depiction of the 5 layers of the Internet Stack described by [4]

ality of these layers helps us understand that network attacks performed in different layers should be studied in a different fashion. In this initial stage of development of the NIDS, we set the focus to the Transport, Network, and Link layers. Studying thoroughly these layers helps us become more familiar with scenarios in which cyber-attackers exploit weaknesses. More specifically, it has been determined what classes of attacks involving breaching in the mentioned layers will be part of the study.

1.3 Classes of Network Attacks

Starting with flooding attacks, where the objective is usually to generate a denial of service effect on the victim. Flooding attacks overwhelm the communication service with the presence of a considerable amount of traffic, which most of the times results in the collapse of an already established communication between client and server. Another network attack taken into account is impersonation attack. Here, the intruder's ultimate goal is to falsify the identity of a trusted entity to obtain sensitive data. The third and last attack to be considered in the analysis of network attacks in this research is injection attack. This class of attack attempts to introduce maliciously input to a particular network, machine, or program. This action can result in several consequences ranging from denial of service to data theft [5]. Providing details regarding these attacks, their mechanism, and how they relate to the characteristics of the link layer in wireless networks is key in the discovery of an effective implementation of an IDS. An enlightening research work in regards of Wi-Fi network attacks makes available a public dataset that was created with the special purpose of the analysis of IDSs. This dataset is called the AWID dataset [6]. The background research made as part of the generation of this dataset provided basic network attack information as well as specifications for types and classes of attacks, which are very popular among network attackers.

1.4 Machine Learning in Network Security

After an overview of how the network attacks were performed and the data was collected, it presented the approach taken to develop the intrusion detection system, which is Machine Learning (ML). Nowadays, ML has become a very popular technology used in several fields including the information technology. Modern research has leaned toward this technology and has demonstrated that it is one of the most effective approaches to develop a model that can be trained to scan network traffic and detect if a network attack is being held. Even though, this technology appears to be emerging, research continues to evolve rapidly and makes available more efficient techniques. In recent years, it has become very popular the design of learning model in an ensemble fashion as the combination of several ML models causes every model to help each other in correcting weakness; therefore, the ensemble model will present an improved performance in comparison to the performance of each model considered alone. The outcome of including ensemble learning in the field of network security represents a great benefit since the ability to more accurately distinguish between normal and malicious traffic can prevent the compromise of sensitive data by its unauthorized use, misuse, or abuse [7] in any instance where the situation may happens.

It is critical as well to highlight that the technology of ML provides precise way in which the effectiveness of a model can be measured. Most ML packages normally provide accuracy, which considered in classifying if the network traffic is in normal state or an attack is present, becomes of great importance. Nevertheless, several other metrics can present deeper information in relation to how effectively the model is detecting intruders. For example, logarithmic cost, F1 score, mean absolute error, or mean squared error among many others. This metrics are as important as accuracy and in some cases probably even more, especially when the samples of one class are imbalanced compared to the samples of the other class [8]. This is exactly the case of data describing a network attack. In a realistic network, it is very unlikely that an attack is being held, so if a network is monitored for a certain amount of time, the time frame of the network operating in a normal state is extremely larger than the time frame of the network while it was victim of a network attack. Therefore, if an IDS classifies all network traffic as normal traffic, it will obtain an accuracy very high because the very few instances where it classified the network behavior incorrectly were attacks. This result is clearly misleading and strongly suggests that in the case of IDS development, there needs to be included as much precision metrics as possible. The mentioned dataset demonstrates what has just been discussed, presenting large number of records where the total number of packets during normal network behavior surpasses considerably to the number of total packets when an attack was ongoing.

During last year, considering up to the 10 most disrupting events regarding cyberattacks, they resulted in compromising of 5 million records with sensitive credit information in the mildest of these events up to breaching of 1 billion records in the worst case [9]. It can be stated that organizational data breaches generate a greater concern about the safety of personal data. The study of a wireless network that simulates more the set up a personal network does not restrict the findings to benefit these type of networks, and even though security levels may vary from a personal wireless network to an organizational one, the principles followed to compromise a network are the same. However, network communication is not just of one type in an organization as they are traditionally in a personal network. Therefore, it is o greater benefit to study the development of an IDS that is capable of scan the behavior of a network with a structure that combines wired as well as wireless networks. Even though the data, which was previously exposed, is comprehensive and provides a considerable amount of data to develop an effective model, but it lacks the combination of different types of networks that as mentioned before are more common in organization. An experimental set up that includes these requirements is necessary to simulate the performance of an IDS against network threats presents in this network architecture.

The result of the combination of these two different type of networks is motivation for the development of an IDS that is capable of scanning and analyzing not only wireless traffic, but Ethernet traffic as well. The inclusion of wired networks allow this research to include attacks in this type of networks. Wireless packets provide very useful information in their headers pertaining to the link layer, which is one of the layers said to be focused on. Moreover, there is also a demonstration of how much information, probably more that it is needed, is present in packets at the link layer in the AWID dataset previously exposed. This dataset contains 155 different characteristics, called attributes, that can be displayed for each Wi-Fi frame. So far, it has been referring to the unit of measure for network traffic as frame for wireless connection. A different way of analyzing the traffic for the layers above Link layer is employed. A unit flow is used, which is a condensed structure that describes the characteristics of a network traffic in aggregation. Attacks against a wired network in the link layer are not as common as attacks to the higher layers where several varieties of techniques designed to create breaches. Therefore, an expected characteristic of an IDS that analyzes wired traffic is that it should detect only high-layer layer attacks. Also, it is necessary to take into account that wireless networks can be victims of the same type of attacks for the higher layers; nevertheless, wireless traffic is very well known to be the target of link layer attacks as this network is open to all devices within range. For this reason another component in the section of intrusion detection that should exist in the IDS is the capability of analyzing Wi-Fi frames and performing the respective data processing to detect either in a wired network attacks or wireless attacks. Moreover, there should be another component that focuses its analysis on link layer attack, and performs its analysis and attack detection on wireless frames.

Network attacks that are performed either in wired or wireless environment are commonly studied from a perspective that is leaned towards the type of attack, whether it is flooding, impersonation, or injection type; or more focused on the structure of the network, whether it is a home network set up, or enterprise network set up; or it can stress more the Internet layer which the intruder it aiming at, whether the network layer or link layer as discussed in the preceding paragraph. All these perspectives are critical for a comprehensive study of cybersecurity, but this work proposes the consideration of all the mentioned factors from the standpoint of subjects that already are members of the network and exploit this privilege to trespass security barriers. This particular scenario is referred to as internal network attacks, and it is considered the most preferred scenario for network intruders to put into practice their malicious techniques. A study has demonstrated that at enterprise level, 74 percent of security incidents were as a result of extended enterprise. From this 74 percent, 42 percent represent actual inadvertent employees. The total 74 percent is reached considering, besides employees, customers, and suppliers, entities that are know to the company. The remaining 26 percent is attributed to parties unknown to the organization [10]. Insider attacks is not a modern issue. Another study makes reference to a survey made to security personnel member of US corporations and agencies, where insider incidents were cites 59 percent of times [11].

One of the most shocking events exposing the catastrophic outcomes of insider attacks was portrayed in the account that relates the exposure of sensitive information in a data breach occurred at Nuance Health care in 2017. Personal health information for 45,000 patients were stolen in the incident. After the Federal investigation, it was found that a former Nuance employee had been the principal responsible of the trespass [12]. Even though this organization attack instance is not an exact duplicate of the experiment studied in this work, it was exposed because it displays an outcome that well could have been achieved from an employee who possesses the privilege of being already connected to the communications network, which such case would be a very close scenario to the one mainly studied in this research. It can clearly be noted that the real case is not far from our proposed experiment. It has to be acknowledged that incidents, not as serious, that do emulate our experiment can be spotted. However, this example presents in a very clear manner the gravity of the harm that can result as a consequence from insider attacks, and reminds to the research community that it is imperative to start multiplying the efforts in providing safer data management systems capable of protecting data even from those who have special privileges in a system, such as being an employee of a particular organization.

1.5 Thesis Outline

The Thesis report is organized in the following manner:

- In Chapter 1, we discussed the motivation of this study. We also covered key concepts considered to present the proposed solution.
- In Chapter 2, we provide literature survey for several works that presented comparable solutions to the same problem exposed in this work, detecting network intrusions.
- Chapter 3 introduces the first stage of this study, where an NIDS is designed and implemented that focuses on Wi-Fi network attacks.
- In Chapter 4, we discuss the architecture of an ensemble learning based multilevel network intrusion detection system along with its evaluation.
- Chapter 5 concludes the thesis and provides an insight for future work.

2. LITERATURE SURVEY

In this chapter, we briefly review the literature for the existing work, where researchers present the design and implementation of network intrusion detection systems (NIDS).

2.1 Wi-Fi Focused Research

Starting with Kolias et al. [6], who extensively studied the attacks against Wi-Fi networks and categorized them. The contribution that proves to be the highlight of this research is introduction of the **Aegean Wi-Fi Intrusion Detection (AWID)** dataset. In addition, this work also included the processing and analysis of the dataset. The dataset was fed into several machine learning algorithms. There are several characteristics that made this work to stand out from similar research made in the field. The authors provided a wide overview of attacks being performed in the 802.11 standard in general. They constructed the dataset with well supported assumptions, which is what makes this work recognized. This work also excelled at exposing a complete analysis of normal 802.11 network traffic containing normal network traffic behavior and behavior that describes a network attack. For the study of Wi-Fi intrusion detection systems, this dataset became a pillar, providing a format that is easy to distribute. and very high quality content as it is composed of real traces. The authors achieved the best accuracy using the J48 algorithm. They reported 96.20% detection rate with all the features in the dataset and 96.26% with 20 features.

Alotaibi et al. [13] attempted to improve the accuracy by applying the majority voting technique in which several machine learning algorithms were used with the AWID dataset and then voting was performed on their results for the final prediction. Another singularity proposed in this work is the use of a technique based on the ensemble method of Extra Trees, which not only improves performance, but was used for feature selection as well. The proposed solution was based on the combination of several machine-learning algorithms to learn patterns for different network behaviors. The initial procedure where patterns are constructed is called offline stage. It was then followed by the online stage, where the classification of network attacks actually occurs. Before the intrusion detection, a filtering process was placed where all unnecessary features were disposed to only leave those that contribute to the classification. This work used the similar ensemble algorithms that we use in our work, the combination of machine-learning algorithms, with a different approach and features. The participating algorithms used were Bagging, Random Forest, and Extra Trees. Although these ML algorithms were used for the classification task, the Majority Voting, the voting technique employed, is what finally determined the classification output. The authors clearly specified that the voting technique was used particularly to improve the accuracy. As desired by the authors, it outperformed the result from Kolias et al. and reported accuracy of 96.32%.

We also found studies that make use of the Deep Learning (DL) approach. In [14] the author used DL for feature extraction, leaving the classification task to a Stacked-Auto encoder (SAE) classifier. The author mentioned that it is the first work considering this approach for IEEE 802.11 networks. The author presented the implementation of the neural network structure used for this classification problem. The neural network was composed of several layers. The first 3 layers were placed in the neural network with the principal purpose of learning what features were the most useful to determine patterns. These patterns aided in classifying network traffic behavior as an attack. The author used an emerging option to be employed as the activation function, which is the Rectified Linear Unit (ReLU) function as an alternative to the traditionally used Sigmoid function for the deep learning models. This process was performed, as stated, to provide a self learning characteristic to this classification model only to determine or “learn” the most effective features to be considered for the actual classification. Later on, the author came to expose the type of classifier used for the actual anomaly detection, which is the Softmax Regression, a classifier capable of handling multi-class classification. An accuracy of 98.66% obtained after the mentioned techniques were applied, however information in relation to data preparation was very limited.

In the AWID dataset, many features have missing values for a large fraction of them. Also, the features found in the dataset are of different types including numeric values and hexadecimal characters. Therefore, it is essential to preprocess the dataset before we use it. We noticed that most of the literature did not discuss the data preparation. In contrast, we report that data preparation in our work in detailed manner.

2.2 Research Focused on Ensemble Models and Multi-level Intrusion Detection

Zaman et al. [15] proposed a more detailed perspective of the analysis of network traffic for intrusion detection. In this work, the authors focused on the different layers of the Internet stack. They proposed 4 different solutions that correspond to attacks analyzed from the perspective of the 4 upper layers in the Internet Stack. The intrusion detection systems are categorized as follows: Application Layer IDS, Transport Layer IDS, Network Layer IDS, and Link Layer IDS. The authors claimed that the results of this approach demonstrate improvement in system performance and scalability. For the task of feature selection, the authors employed Fuzzy Enhanced Support Vector Decision Function. As a result, the highest accuracy reported was of 99.84% for the Transport layer IDS, which used Neural Networks for classification. The best accuracy considering all Layers was of 99.41% using a Neural Network classifier.

Zainal et al. [16] proposed an approach where classifiers with different learning paradigms are combined into a single Ensemble model. The paradigms employed are the following: Linear Genetic Programming, Adaptive Neural Fuzzy Inference

System, and Random Forest (RF). The objective of the authors is to improve the accuracy and reduce the false alarm rate in intrusion detection. Two principal steps were exposed in this work. First, selecting relevant features. Second, developing an ensemble model composed of classifiers with different learning paradigms. They demonstrated that the performance of the ensemble model surpassed performance of the three models used separately for all types of attacks.

Li et al. [17] presented the use of Rough Set Theory and and Quantum Genetic Algorithm for attribute reduction and as a method of classification. The author mentioned that attributes from network packets were reduced using the Quantum Genetic Algorithm. Rough Set Theory was used by them to implement a rough meta-learning classification strategy. This strategy combined multiple rough learning methods. After the experiment, the author demonstrated that detection rate was noticeably improved when Ensemble-Rough Classifiers were used versus the use of Single-Rough classifiers. Particularly, the detection rate increased from 76.86% to 86.25% for DoS attacks.

Wang et al. [18] proposes the use of ensemble learning by using a Bayesian Network and Random Tree as base classifiers. These algorithms were combined with meta learning algorithms using “Random Committee”. Then, voting was performed for the classification task. In this work, authors mentioned that the KDDcup99 dataset was used. One of the main objectives in this work was tackling the unbalanced nature of this dataset using ensemble learning. The model is evaluated using receiver operating characteristic (ROC) curves. For more specific results, the authors computed the area under the ROC curves (AUC). In the results, it was found that the ensemble model outperforms the single based models.

In [19] Nenekazi et al., uncovers the issue of researchers not being able to determine the performance of an ensemble based NIDS until after it is implemented. This work is based on the study of average information gain, which determines performance. This average information gain is associated with the features. Adaboost, which is the weak classifier in the ensemble, is used to obtain the average information gain. The NSL KDD dataset was used, and accuracy was the metric considered to measure performance in this work. The author demonstrates that average information gain lies in the rage of 0.045651 and 0.25615 when accuracy will reach as much as 90%.

3. AN ENSEMBLE LEARNING APPROACH FOR WI-FI NETWORK INTRUSION DETECTION SYSTEM

Wi-Fi (IEEE 802.11) has become the predominant technology in the communication world. Almost all the places, we find Wi-Fi based Internet connectivity. In this chapter, we provide an overview of Wi-Fi infrastructure network and the attacks associated with it. Then, we discuss an ensemble learning based model for Wi-Fi network intrusion detection system (WNIDS).

3.1 Overview of Wi-Fi infrastructure network

Wi-Fi comprises several standards; however, all of them have some essential characteristics in common such as architecture, medium access protocol, and frame structure. There are two types of Wi-Fi networks: i) infrastructure and ii) ad hoc. The main difference between the two architecture is that a central station, access point, is present in the infrastructure mode, whereas the ad hoc mode is a group of stations with no access point. Figure 3.1 shows an infrastructure Wi-Fi network architecture. Wireless devices exchange frames with each other via the access point. The control and management frames are exchanged between a station and access point only [20]. In this work, we focus on the ML model development of an intrusion detection system for infrastructure Wi-Fi networks.

Another important aspect relevant to the understanding of how network communication works in a wireless network is the study of the types of frames and their function. Infrastructure wireless networks make use of management, control, and data frames to handle different type of situations [21].

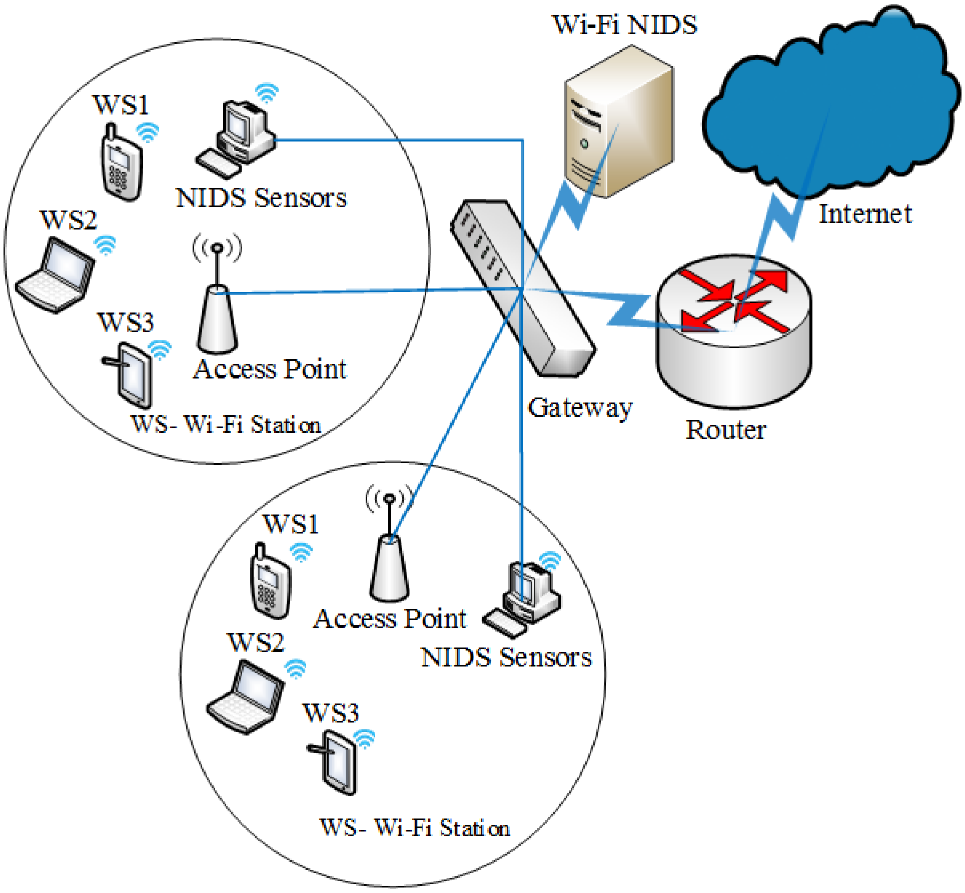


Fig. 3.1. An IDS integrated with infrastructure Wi-Fi network

3.1.1 Wi-Fi Frames

Management Frames

This type of frame has the responsibility of managing the communication between the wireless stations and access point. Services that result from the use of management frames in the network are commonly authentication, association, and synchronization [21]. Following is the description for a few management type frames:

- **Authentication Frame**: Authentication is the first process that needs to be followed by an entity to establish a connection with the network, and it happens once the client has decided to connect to the access point after receiving information through a probe response frame as shown in Figure 3.2. In this process, the role of authentication frames is to demonstrate a membership status in relation to the targeted network. After the authentication identity is presented, the connection provider, generally the access point, either accepts or denies permission to be part of the network. In terms of frames, the process is performed by sending back and forth authentication frames.

- **De-authentication Frame**: The purpose of de-authentication frames is to terminate a connection that had previously been established between a wireless station and an access point. The process is simple and depends on one side of the association. If a de-authentication frame is sent, it must be accepted and the connection is terminated immediately.

- **Association Request Frame**: The association request frame provides information about a network interface card (NIC) requesting connection and information about the service set identifier (SSID) of the target network. It helps the access point to reserve resources for the NIC.

- **Association Response Frame**: It is the response for association request frame as depicted in Figure 3.2. It is sent by the access point to notify the acceptance or denial of association to the NIC that sent the request. When accepted, the NIC can interact with other NICs who are members of the network through the access point.

- **Probe Request Frame**: This frame is used as a tool by NIC to discover the access points in the range.

- **Probe Response Frame**: It is the response to the probe request frame and contains capability information of the access point provided to the client as shown in Figure 3.2.

- **Re-association Request Frame**: It has a similar purpose as the association frame in the sense that it desires to establishes an association. However, in

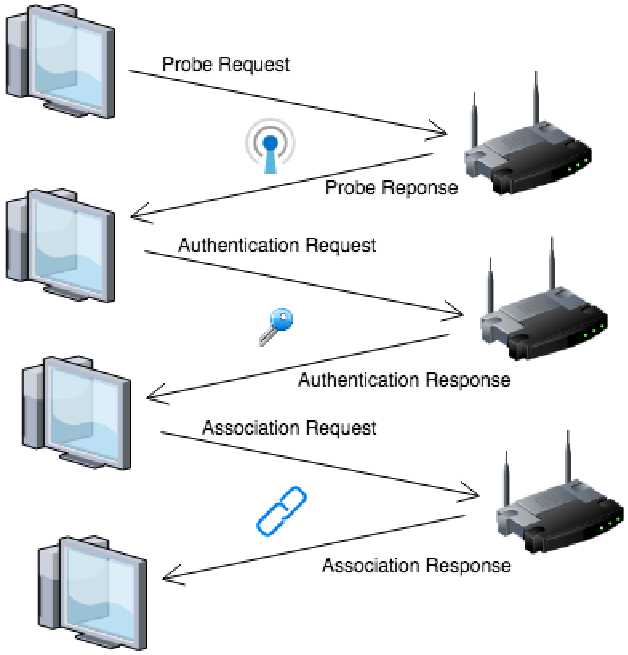


Fig. 3.2. A depiction of the order in which probe, authentication and association frames are sent

this case, the NIC was already associated to a network access point, but looses reception and desires to associate to another access point in the same network. In this process, synchronization is provided to handle situations such as data that was scheduled to be transmitted from the previous access point and now needs to be sent by the new access point.

- **Re-association Response Frame**: Fulfills the same responsibility as the association response frame, and determines and notifies the requesting NIC if association is granted or not.
- **Dis-association Frame**: This frame is used to notify the access point that it is desired to terminate an association. It is useful for the access point to reorganize the the resource allocation.

- **Beacon Frame**: This frame is used by the access point to advertise its presence. It provides information regarding the connectivity conditions of the access point to all the NICs located within the access point range.

Control Frames

Control frames are mainly used to facilitate the actual data exchange. These frames have the purpose of providing an effective data delivery service [21].

- **Request to Send**: Request to send (RTS) frames prevent collisions and belong the initial process called the two-way handshake, which is required before actual data is sent.
- **Clear to Send**: A broadcast frame sent in response to the RTS frame that indicates the sender to send the frame and instructs other stations back-off from sending frames for the reserved duration.
- **Acknowledgment Frame**: The acknowledgment (ACK) frame is used to notify the sender of any errors present in the frame sent. If the frame is clear, the ACK frame is sent as response. For any frame, if an ACK frame was not received as response, the frame is retransmitted.

Data Frames

Data frames are used to carry the actual data desired to be transmitted. The information contained in these frames is used generally at a higher-level such as web site content, email content, etc.

3.1.2 Attacks on Wi-Fi

Wi-Fi frames in general fulfill very critical functions. A profound understanding of their usage permits the improvement of performance in network communication.

However, the same amount of understanding possessed by entities with malicious intents can develop breaches to infect or break down wireless networks. Management frames, for example, can be used to launch a denial of service (DoS) attack known as beacon flooding that causes the network to collapse and bring down connectivity for clients. Control frames as well as management frames are not encrypted like data frames, so attackers can manipulate these frames to compromise the secure communication between the member of the network. Many publicly available penetration testing tools such as Aircrack-ng [22] and Metasploit [23] can be used to launch attacks against Wi-Fi networks. Although these tools are meant to assess network security, adversaries use them for attack purposes.

There are several attacks possible in Wi-Fi networks which are categorized as: i) flooding ii) impersonation and iii) injection by [13]. In a flooding attack, attackers send noticeably increased number of frames. In some cases, this prevents the victim from being able to find a connection and causes a denial-of-service (DoS) attack. A well-known flooding attack is “Beacon flooding”. The attack exploits the functionality of beacon frames, which are used to provide information regarding connectivity for the clients seeking network connection. During beacon flooding, a large number of fake beacon frames are sent to the victim [24]. In an impersonation attack, an unauthorized access point is used by the intruder that advertises network connectivity to the victims. An example of impersonation attack is the “Evil Twin”. A malicious access point is placed in a visible spot in the network to launch this attack. The first purpose of the attacker is to make this access point look genuine by using the SSID name of an existing Wi-Fi network. Once a victim connects to the Internet through this access point, it can take advantage of the victim in several ways. It can capture the information exchanged between the victim and the websites being visited, which can cause serious consequences [25]. An injection attack interferes with an existing connection by constructing forged frames that appear as legitimate. A successful attack allows intruders to intercept and alter Wi-Fi frames exchanged between two communicating parties. This type of attack is commonly used to launch man-in-the-middle (MITM) and DoS attacks. Attackers usually send a large number of small frames in a short interval to make this attack successful. One popular example of this attack is “ARP injection” in which an attacker sends spoofed ARP messages to the victim host with an aim to bind the MAC address of the attacker host with the IP address of a different host. Thus, traffic routing from the victim host to the destination host with the corresponding IP address takes place via the attacker host [26].

3.2 WNIDS Implementation

The proposed installation of our WNIDS is shown in Figure 3.1. The WNIDS sensors will collect Wi-Fi frames and send them to the WNIDS for intrusion detection. As the NIDS is implemented using machine learning approach, we need to train and validate it before deployment using existing Wi-Fi frames dataset; it is the AWID dataset in our case. Therefore, we will start with the description of the dataset used in our work along with the steps taken for the data preprocessing. A brief introduction of various ensemble algorithms used for the model development are provided as well. Following that, the application of the algorithms on the training data to identify the best one and its evaluation on the test data will be discussed. The steps for WNIDS implementation are summarized in Figure 3.3.

3.2.1 AWID dataset

The AWID Wi-Fi intrusion dataset was collected in an environment that models a common Wi-Fi network. Several Wi-Fi stations including a desktop, a smart TV, a tablet, laptops, and smartphones were connected with an access point for the Internet connection. One machine was used as an intruder in charge of launching the attacks. A desktop computer configured with monitor mode was used to capture Wi-Fi frames. It was equipped with high processing capabilities to be able to capture a large number of frames at very high-speed. All these equipment for the experimental set-up had a variety of hardware, operating systems, and other characteristics. Furthermore, to maintain the data capturing as realistic as possible, mobile devices, such as smartphones and tablets, were kept in constant motion, laptops experienced sporadic movement, and the desktop and the smart TV were kept in fixed locations during the experiment. The Wi-Fi frames were collected for around five days to release different datasets version, discussed below, for the intrusion detection.

The AWID dataset comes in two different versions: i) attack-class and ii) attack-specific. In the attack-class version, each record in the dataset is labeled as either one of the three attacks or normal. The attack classes are flooding, impersonation, and injection. If the classification is intended to identify the records with the specific attack, the attack-specific version labels the same records with 17 different Wi-Fi attacks or normal.

Each version is also released as large and reduced sizes to make them available for systems with different processing capacity. Overall, there are four versions of the AWID dataset. In our work, we use the reduced size attack-class version of the dataset. The distribution of the records in the dataset is shown in Table 3.1. We see from the distribution that the AWID is an imbalanced dataset, more than 90% of the records in both the training and test sets are normal. It is important to mention that the relationship between the large and the reduced sets is merely in the number of features and the labels. The reduced dataset was collected independently from the large dataset at different times with different tools. Therefore, they do not have any relationship besides the number of features and labels.

Each record in the AWID dataset is a Wi-Fi frame having 155 features [27] in all the versions of the dataset. These features represent various header fields values in each captured frame. They can define some traffic patterns helpful to detect intrusions. Nevertheless, some features may represent noise due to the raw state of the dataset. Moreover, Wi-Fi frames include management, control, and data frames. As a result, not all the features will apply to all the frames. Therefore, some records may have missing values for those features.

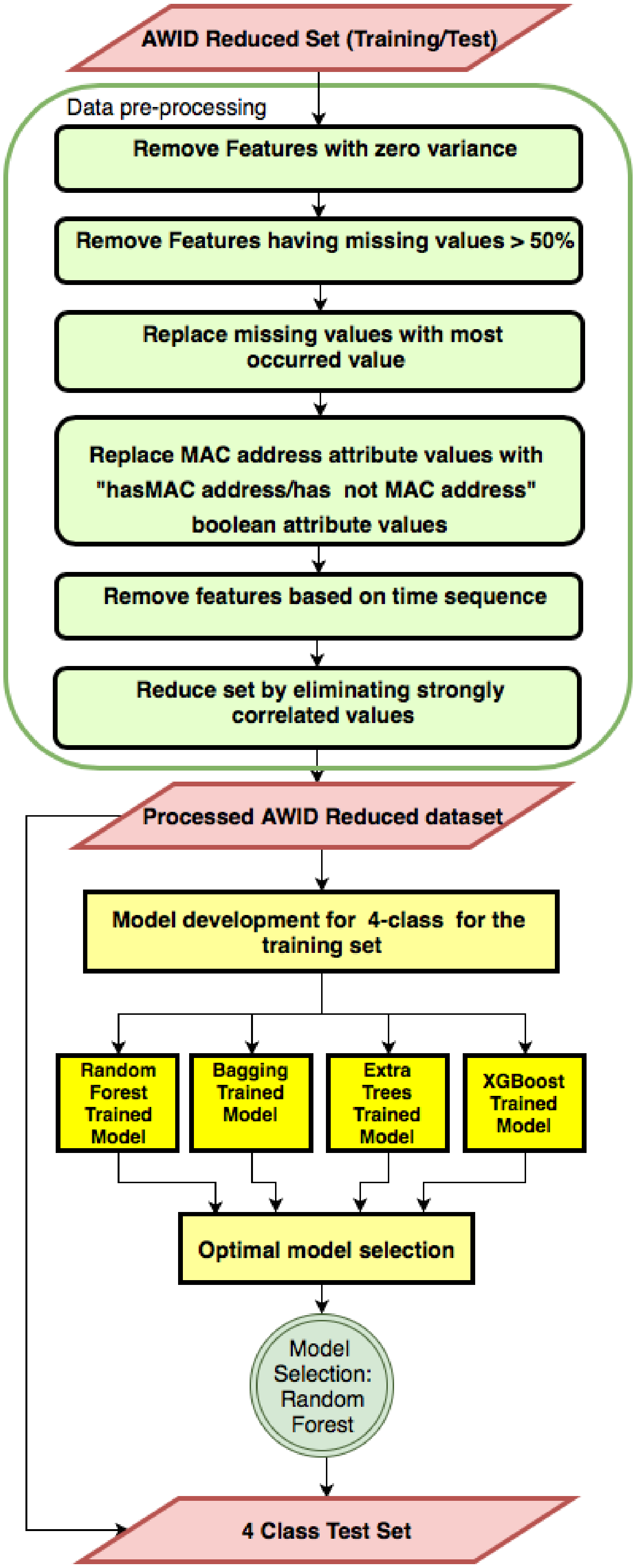

AWID Reduced Set (Training/Test)
Data pre-processing
Remove Features with zero variance
Remove Features having missing values > 50%
Replace missing values with most occurred value
Replace MAC address attribute values with "hasMAC address/has not MAC address" boolean attribute values
Remove features based on time sequence
Reduce set by eliminating strongly correlated values
Processed AWID Reduced dataset
Model development for 4-class for the training set
Random Forest Trained Model
Bagging Trained Model
Extra Trees Trained Model
XGBoost Trained Model
Optimal model selection
Model Selection: Random Forest
4 Class Test Set

Fig. 3.3. Steps involved in the WNIDS implementation

Table 3.1.
Traffic records distribution for the training and test sets in the reduced AWID dataset [13]

	Training Set (in%)	**Test Set (in%)**
Normal	1633190 (91%)	530785 (92.2%)
Flooding	48484 (2.7%)	8097 (1.4%)
Impersonation	48522 (2.7%)	20079 (3.5%)
Injection	65379 (3.6%)	16682 (2.9%)
Total	1795575 (100%)	575643 (100%)

3.2.2 Data Preparation

As mentioned in the description of the AWID dataset, some values of the features in each record of the dataset may represent noise for the intrusion detection task or may have missing values. For these reason, it is essential to pre-process the dataset before using it. Initially, each record has 154 features and one target class (155 features) in the dataset. We removed the zero-variance features, i.e. features having same value for all the records. We found 27 features with zero-variance. After that, we removed features that have more than 50% missing values. After performing these steps, the number of features reduced to 36 listed in Table 3.2. For a feature with still missing values, we replaced the missing values with the most frequently occurred value in the feature. Features 29, 30, 31, 32, and 33 are MAC addresses with hexadecimal values associated with the Wi-Fi adapters of senders, access points, and receivers. They are device-dependent and vary with them. We replaced the values in these features with 1 if there was a MAC address, and 0 if there was a missing value. Features 0, 1, 2, 3, and 13 were eliminated as they are time-specific and our implementation is not based on time-series. The implementation treats each record individually for prediction. Feature 19 is a combination of Features 20 and 21;

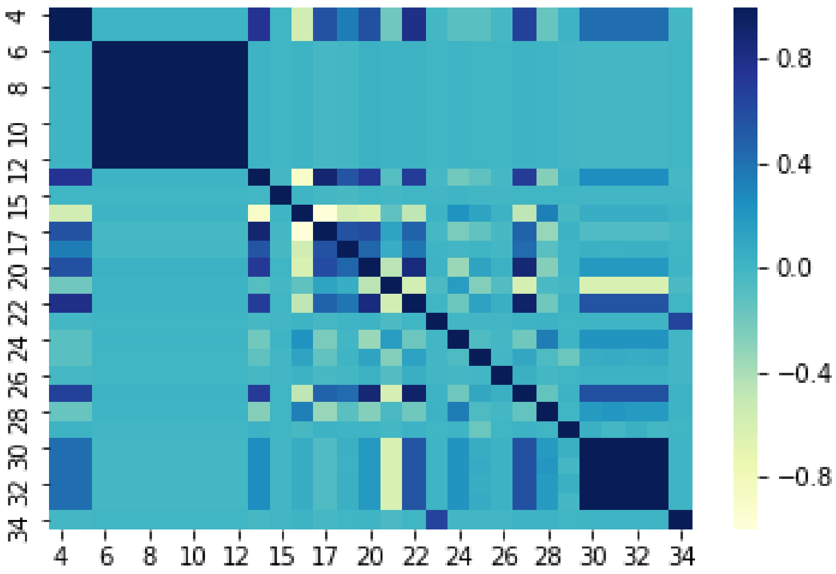


Fig. 3.4. Features correlation heat map

therefore, we eliminated it. Also, Feature 35 was discarded as it denotes the sequence number.

After doing the above steps, we computed the correlation between the features. We found that a few of the features are strongly correlated to others. Figure 3.4 shows the correlation heat map of the features. It can be seen from the figure that features {4, 5}, {6, 7, 8, 9, 10, 11, 12}, {30, 31, 32, 33} form strong positively correlated groups as the correlation value corresponding to them is 1. Similarly, features {16, 17} form strong negatively correlated group as the correlation value is -1. We selected one feature from each group and discarded the remaining features from the corresponding group. As a result, Features 4, 6, 30, and 16 were selected from the groups. After going through all these steps, we derived a new features list consists of 18 features. These features are shown in *italic* font in Table 3.2.

Table 3.2.
Features remained in the AWID dataset after pre-processing

#	**Attribute name**	**Description**
0	frame.time_epoch	Epoch Time
1	frame.time_delta	Time delta from previous captured frame
2	frame.time_delta_displayed	Time delta from previous displayed frame
3	frame.time_relative	Time since reference or first frame
4	*frame.len*	*Frame length*
5	frame.cap_len	Frame length stored into the capture file
6	*radiotap.length*	*Header length*
7	radiotap.present.tsft	TSFT
8	radiotap.present.flags	Flags
9	radiotap.present.channel	Channel
10	radiotap.present.dbm_antsignal	dB Antenna Signal
11	radiotap.present.antenna	Antenna
12	radiotap.present.rxflags	RX flags
13	radiotap.mactime	MAC timestamp
14	*radiotap.datarate*	*Data rate (Mb/s)*
15	*radiotap.channel.freq*	*Channel frequency*
16	*radiotap.channel.type.cck*	*Complementary Code Keying*
17	radiotap.channel.type.ofdm	Orthogonal Frequency-Division Multiplexing
18	*radiotap.dbm_antsignal*	*Antenna signal*
19	wlan.fc.type_subtype	Type/Sub-type
20	*wlan.fc.type*	*Type*
21	*wlan.fc.subtype*	*Sub-type*
22	*wlan.fc.ds*	*DS status*
23	*wlan.fc.frag*	*More Fragments*
24	*wlan.fc.retry*	*Retry*
25	*wlan.fc.pwrmgt*	*Power Management*
26	*wlan.fc.moredata*	*More Data*
27	*wlan.fc.protected*	*Protected flag*
28	*wlan.duration*	*Duration*
29	*wlan.ra*	*Receiver address*
30	*wlan.da*	*Destination address*
31	wlan.ta	Transmitter address
32	wlan.sa	Source address
33	wlan.bssid	BSS Id
34	*wlan.frag*	*Fragment number*
35	wlan.seq	Sequence number

3.2.3 Ensemble Algorithms

Ensemble learning algorithm builds a prediction model for a machine learning problem by combining the strengths of various base learners. It helps to get a more accurate solution compared to a single base learner. We considered following ensemble algorithms for our work:

Bagging

In this method, random subsets of records with replacement are created from a dataset. An ML algorithm, individually may be prone to over-fitting such as decision tree, is applied to each subset. The final model comprises models trained from each subset. When a new record is given for prediction, the model classifies the record with a label predicted by the majority of the component models [28].

Random Forest (RF)

It is an improved version of bagging that uses decision trees. The learning capabilities of a decision tree give RF the ability to extract precise information considering all features. This in turn results in an accurate performance. However, it can also result in excessive variance and produce over-fitting. Combining this approach with the principles of the bagging algorithm, the issue could be acceptably corrected. The role of bagging in RF is generating random subsets of data. After randomly selecting the samples, RF verifies that the samples from one subset have less correlation relative to the other subsets. Decision trees are fitted for each subset. Each decision tree is used to classify data outside its training subset. Finally, from the results of every decision tree, the most common occurrence is chosen as the overall classification output. One additional feature of RF that improves its performance over bagging is that it randomly samples a subset of features from the features set for each decision tree, thus tackling high-variance more efficiently [28].

Extra Trees

This method shares characteristics of the RF algorithm. The clearest similarity is the use of decision trees at the core. The difference is that it adds further randomization compared to RF. In RF, random features are sampled for each decision tree, then locally optimal feature/split combination is computed from those features. In contrast to RF, ET selects a random value for the split. This approach makes the algorithm more generalized and less-prone to over-fit [29].

XGBoost

It is an efficient implementation of the Gradient Boosting (GB) algorithm. In the GB, several weak learners are used in sequence. During the training process, each learner focuses in those samples which were misclassified by the previous learner in the sequence. The GB consists in minimizing a cost function. The cost function describes the difference between an actual value and the approximation corresponding to the actual value. The minimization problem is tacked with derivatives, and the objective is to find the fastest descent in the difference between the actual value and the approximation. One of the main concerns in the use of GB is time. Even though this algorithm is recognized by its learning effectiveness, the performance comes with a burden of expensive computation. These issues became the motivation to provide a set of tools as XGBoost to bear the computational burden of GB [30].

3.2.4 Results

We used the above-mentioned ensemble to develop the models on AWID training dataset using 18 features. The ML framework used for model development is scikit-learn [31], written in Python. We performed 10-fold cross-validation to develop the models for each algorithm.Each algorithm was run with different random states and the mean accuracy was used to select the best model. Table 3.3 shows the classification accuracy for each algorithm. The RF model achieved the highest accuracy. We chose it as the final model for the WNIDS and evaluated it on the test dataset. The performance of the final model is reported using accuracy, precision, recall, and f1-score. These can be derived from the confusion matrix retrieved from the prediction output. These metrics along with the confusion matrix are defined as follows:

Table 3.3.
Classification accuracy of ensemble algorithms on AWID training dataset

Algorithm	**Accuracy (in %)**
Bagging	98.957 (±0.0471)
ExtraTrees	99.017 (±0.0407)
Random Forest	**99.096 (±0.0485)**
XGBoost	98.943 (±0.0000)

- **Confusion Matrix (CM)**: an $n \times n$-dimensions, where n is the number of class labels in the dataset. A cell location (i, j) denotes the number of predicted value for the column j class label corresponding to true value of the row i class label. A cell location (i, i) denotes true-positive for class label i. All cell locations $(i, j \mid i \neq j, 1 \leq j \leq n)$ denotes false-positive (FP) for class label j. All cell locations $(i, j \mid i \neq j, 1 \leq i \leq n)$ denotes false-negative (FN) for class label i.

- **Accuracy (in %)**: $\frac{\#\ of\ correctly\ predicted\ records}{\#\ of\ total\ records} \times 100$
$= \frac{\sum_{i=1}^{n} CM_{(i,i)}}{\sum_{i=1}^{n} \sum_{j=1, i \neq j}^{n} CM_{(i,j)}} \times 100$

- **Precision**: For class label i, $P_i = \frac{CM_{(i,i)}}{\sum_{j=1}^{n} CM_{(i,j)}}$
Average Precision: $P_{avg} = \frac{\sum_{i=1}^{n} N_i \times P_i}{\sum_{i=1}^{n} N_i}$, $N_i = \#$ of records for class i

- **Recall**: For class label i, $R_i = \frac{CM_{(i,i)}}{\sum_{j=1}^{n} CM_{(j,i)}}$
Average Recall: $R_{avg} = \frac{\sum_{i=1}^{n} N_i \times R_i}{\sum_{i=1}^{n} N_i}$, $N_i = \#$ of records for class i

- **F1-score**: A useful metric to evaluate the models for imbalanced datasets. A high value of f1-score is considered good for the classification task of the imbalanced dataset. It is derived using precision and recall as follows:
  For class label i, $F_i = \frac{2 \times P_i \times R_i}{P_i + R_i}$
  Average F1-score: $F_{avg} = \frac{\sum_{i=1}^{n} N_i \times F_i}{\sum_{i=1}^{n} N_i}$, $N_i = \#$ of records for class i

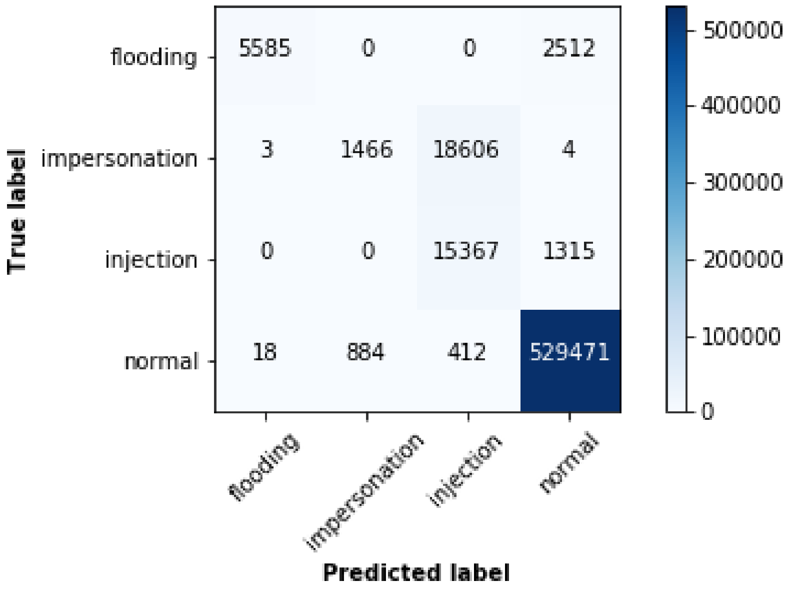


Fig. 3.5. Confusion Matrix for the 4-class classification model

The RF model achieved an accuracy of **95.873**% to identify the records with their classes. Figure 3.5 shows the confusion matrix for the model. Figure 3.6 shows the precision, recall, and f1-score values for all the traffic classes along with average values. We noticed that the performance of the RF model is low for the impersonation attack as its f1-score is 0.13. Most of the records for the impersonation attack are classified as an injection attack as seen in Figure 3.5. The average precision, recall, and f1-score of the model are 0.96, 0.96, and 0.95, respectively. Although we achieved a slightly lesser accuracy than [6] and [13], we found that they classified most of the impersonation attack records into normal. Our RF model performs better in this aspect compared

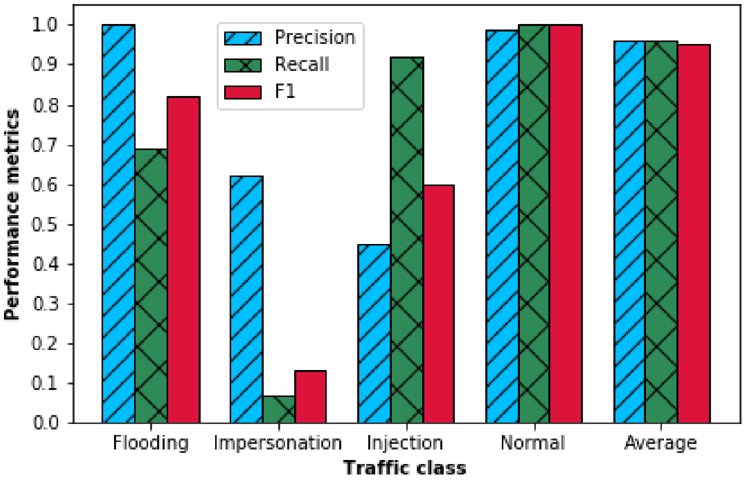


Fig. 3.6. Performance metrics for each class

to them as we identify the impersonation attack into another attack class instead of normal. Also, the f1-score value reported by [6] is lower than ours. Considering

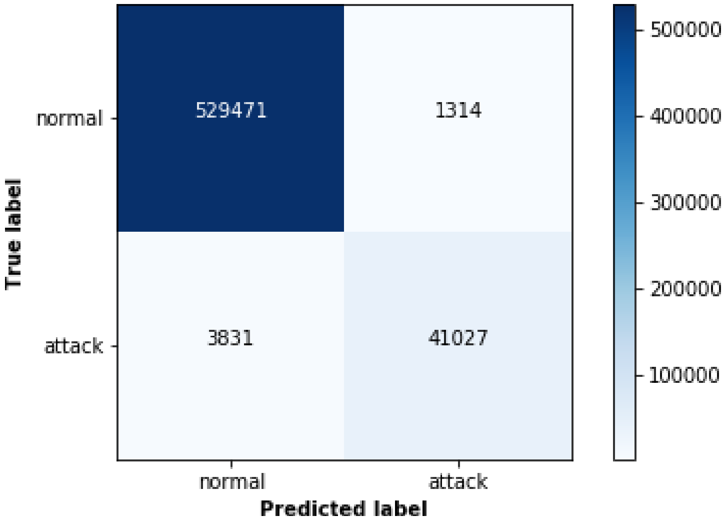


Fig. 3.7. The 4-class confusion matrix converted into the 2-class confusion matrix

flooding, impersonation, and injection attacks as attack only class, then the 4-class confusion matrix shown in Figure 3.5 can be mapped into the confusion matrix of Figure 3.7. The model achieved an accuracy of **99.106%** of identifying whether a record is an attack or normal. In comparison, [6] and [13] achieved accuracy of 96.28% and 96.32%, respectively in this context. The results of this work were published in [32].

4. MULTI-LEVEL INTRUSION DETECTION SYSTEM FOR WI-FI DOMINANT NETWORKS

The previous chapter provided an analysis about obtaining critical information from Wi-Fi traffic to determine the presence of a network attack. The analysis in the previous chapter considered information pertaining to the Wi-Fi specific link layer of the TCP/IP Stack. In this chapter, we desire to expand the analysis to consider information pertaining to the network layer and aboce. Furthermore, we decided to expand the architecture of the network from strictly Wi-Fi to a combination of Wi-Fi and Wired networks. We concentrate our study in the consideration of the most popular networks in general: a home local area network (WLAN) and Wi-Fi dominant organizational network.

4.1 Home Wi-Fi (WLAN) Setup

Over the last decades, the preference of Wi-Fi networks over Ethernet networks has become stronger and stronger. A research conducted by Parks Associates in 2017 explained that more than 70% of households have Wi-Fi or Apple AirPort access [33]. Moreover, these households had an average of 30% more computing devices than those without Wi-Fi. This statistics shows that wireless connectivity is predominant at homes, and the trend is very likely to increase the popularity gap between wireless and wired networks at homes. Having this technology in almost every household brings the concern of security. Attackers are aware of this trend as well, and they perform their malicious research to develop new mechanisms to compromise wireless networks. Finding security deficiencies in one type of wireless network will most likely be useful to find security deficiencies in other types of wireless networks.

4.1.1 Home Wi-Fi (WLAN) Setup Implementation

To replicate a WLAN network setup, we considered basic elements required to create an emulated environment, such as Wi-Fi access point, and a few desktop machines equipped with Wi-Fi adapters as WLAN clients. This part of the work can be considered the continuation of the previous chapter since the primary focus is the elaboration of a strictly wireless environment. Moreover, the analysis will be centered around processing the data taking into account the characteristics pertaining to the link layer. It is similar to the information (attributes) considered in the previous chapter. To accomplish this, the set of elements that used for set-up the WLAN environment were as follows:

- **Wi-Fi clients**: We used three Dell Inspiron 3668 Intel Core i5 with 2.4 GHz clock speed and 8 GB memory. The machines were equipped with Wi-Fi adapter to enable wireless connectivity. We installed 64-bit Kali Linux that has powerful network penetration testing tools such as Metasploit, Aircrack-ng, and so on.

- **Access Point**: We used Tp-Link AC 1750 router model that supports 802.11ac standard with dual band Gigabit connections: 2.4 GHz 450Mbps and 6GHz 1300 Mbps.

We placed the three machines in the environment with different objectives. The objective of the first desktop machine was to monitor the network traffic. When network traffic was desired to be collected, we set the network interface card to monitor mode as depicted in Figure 4.1. We collected and processed the data in this machine. The second computer was in charge of making use of the tools available in Kali to perform the malicious network interventions. The third desktop was placed in the environment to play the role of the victim, so it would be the one affected by the attack of the second desktop.

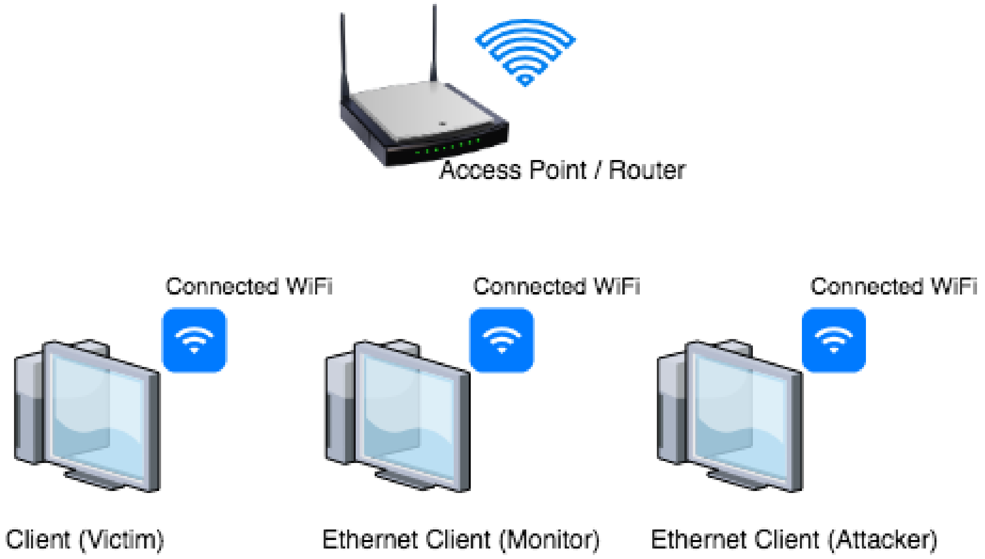


Fig. 4.1. A common wireless based home network environment set-up.

4.2 Organizational Network Infrastructure

Organizational networks are much more appealing for attackers since the potential profit is far greater than that from a home local area network. It is difficult to replicate a large organizational network for penetration testing purposes in a lab environment. However, for our research objectives, we defined a few key characteristics extracted from the this type of network infrastructure. The characteristics mainly focus on the hardware required to expand an organizational network. We observe that client machines with Ethernet and Wi-Fi adapters, routers, access points, and switches are the core elements to establish and expand a network infrastructure.

4.2.1 Environment set-up for an organization Network

The objective at this level in the experiment is to upgrade the environment to not just operate as a wireless network, but to emulate an organization network environment where Ethernet connections are considered as an important component. The additions involve the use of a switch to deploy a network heavily dependent on wired connections. We needed to collect packets for two specific reasons. First, gathering information for the NIDS implementation. Second, for the actual system to be able to capture the packets in real time. To achieve these two objectives, it was necessary to set-up a specific system to analyze and process the data. We selected NetGear ProSAFE 5 Port Gigabit Ethernet Plus Switch that supports port mirroring. It is important to mention that for Ethernet connections we planned to analyze generic network attacks.

The setup for this environment was very similar to the one implemented for only wireless network with the addition of the mentioned switch. Besides the switch, there were two new laptops introduced as well. These laptops were added as Wi-Fi clients and previously used desktop machines were connected through the switch to the router via Ethernet cards. These clients would maintain a wireless connection in the environment in the same manner as laptops usually connect in organizational networks. Following the same line, desktop clients would be connected to the network through Ethernet cards as normally they do in organizational networks. A representation of this environment is shown in Figure 4.2.

4.3 Launching Network Attacks

This section exposes the attacks that were performed in the infrastructure mentioned in the previous section. Link layer attacks were performed for wireless network. Generic network based attacks were launched for the combined Ethernet and Wi-Fi networks.

Overview of Implemented Attacks

The purpose of the data link layer is to transfer data in and out of the physical link in the network between systems and routers [34]. Generally, attacks in this layer have objectives such as Man-in-the-middle (MITM), Address Resolution Protocol (ARP) spoofing, or Beacon flooding [35]. There could be situations where attackers

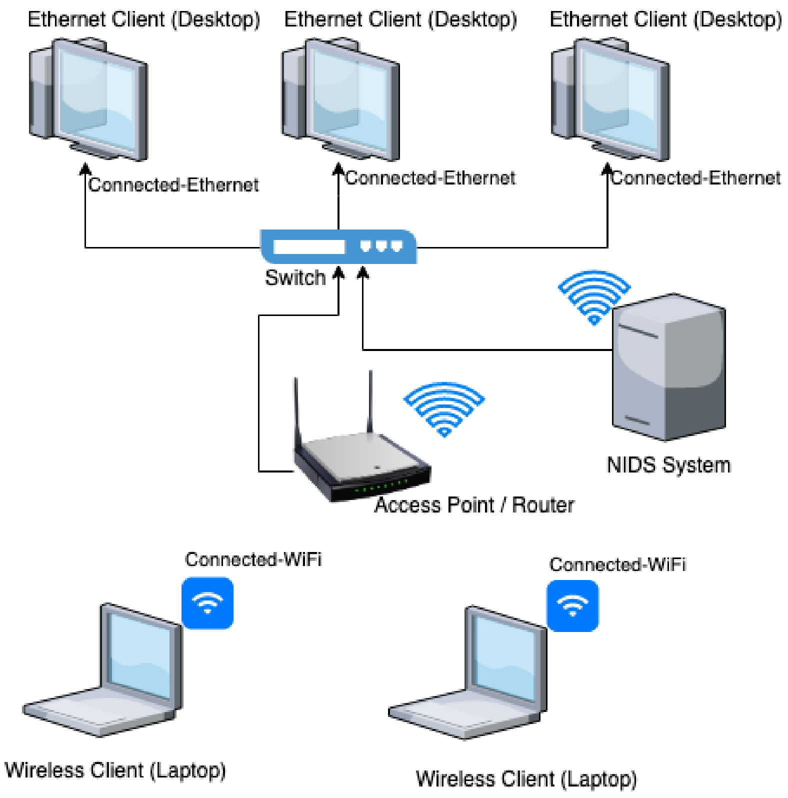


Fig. 4.2. An schematic of an organizational network.

disguise themselves to cause the victim to disconnect from the network by triggering the deauthentication process maliciously sending the deauthentication packet [36]. There are attacks that target the network layer and layers above it causing denial of service (DoS), or spoofing [37]. We collected the network traffic using `tcpdump` through the monitoring system. By collecting these packets, we can analyze activity between the attacker and the victim systems.

- **Evil Twin**: The attacker can create a fake access point (AP) or another AP with the same name in the network. When the user connects to the fake AP the attacker can redirect the packets to a legitimate network connection; it will provide the appearance of a regular AP while enabled to view packets passing through [38].

- **ARP Spoofing**: ARP is used for mapping IP and MAC addresses. The attacker sends a spoofed ARP response to the victim asking the router's MAC address by using the router's IP address and linking it to its own MAC address. When the victim sends a packet to the "router", it is actually directed to the attacker. The attacker can deceive the router in a similar fashion by pretending to be associated to the victim's IP address with its own MAC address [39]. Once the routing tables for both the victim and the router are updated, they will respond to the attacker without realizing it.

- **Beacon Flooding**: It is a simple attack, but can cause considerable damage to the victim. The attacker would create a large amount of APs with different SSID at extremely fast rates. As the wireless clients try to identify each AP, they get overloaded and their system crash [40].

- **Deauthentication**: It is the procedure followed by an AP and Wi-Fi client to disconnect the client from the network. By sending a deauthentication packet from the attacker's adapter, the targeted victim is disconnected from the network, which can only be annoying or can provide enough time for the attacker to prepare MITM attack [36].

- **IP Spoofing**: The attacker spoofs the IP address of another machine and sends packets with the spoofed IP address to the victim. The victim's response goes to the machine associated with that IP address instead of the attacker. It can be compared to forging a person's signature [41].

- **Ping Flood (ICMP Flood)**: In this DoS attack, the intruder overwhelms the victim's network with large amounts of Ping request packets. Since there has to be an equal number of Ping response packets, this causes an overflow [42]. With both the incoming and outgoing channels being overflowed with packets, it ultimately results in a DoS.

4.4 Implementation of Multi-Level Network Intrusion Detection System (ML-NIDS)

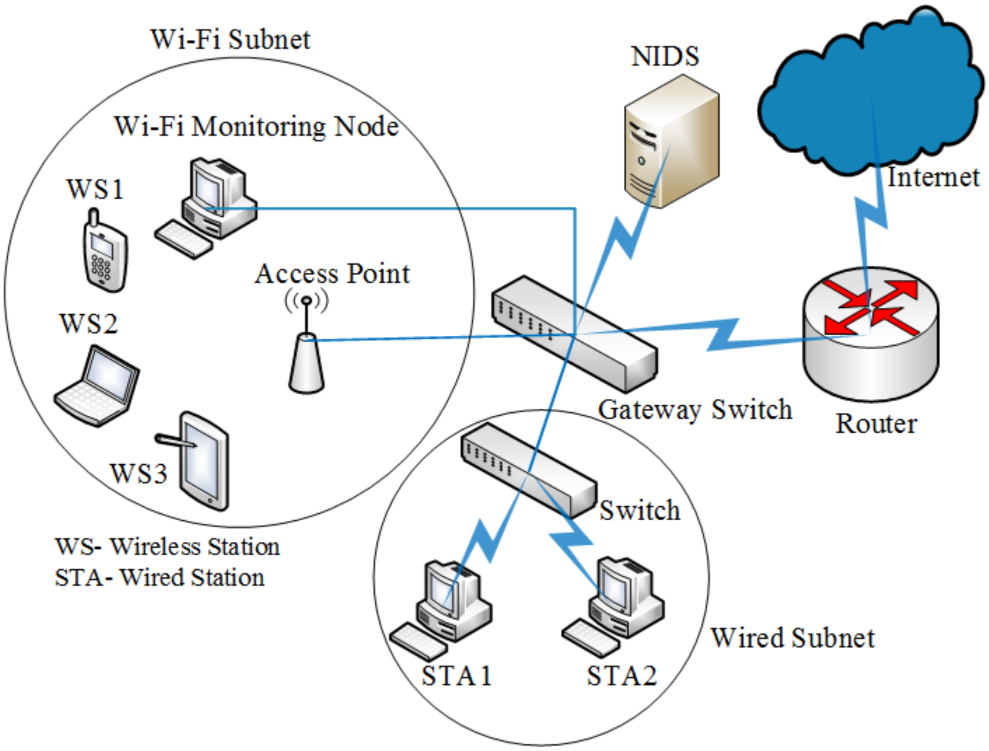


Fig. 4.3. A prototype for Multi-Level NIDS deployment in Wi-Fi dominant networks

Figure 4.3 shows a Wi-Fi dominant campus network prototype for our ML-NIDS implementation. We connect the NIDS system with a Gateway switch. The Gateway switch has several Ethernet ports. Using these ports, we connect Wi-Fi access points (AP) and other switches for local sub-networks. A monitoring node (installed with IDS in the figure) captures Wi-Fi frames for the traffic outgoing from or incoming to the wireless stations within a wireless network. The NIDS receives the packets for wired stations as well. Besides data, a network packet or frame consists of several headers corresponding to the different layers implemented in the network software stack of a wired or wireless station. These layers include application, transport, network, and data-link layers arranged in a top-down manner in the TCP/IP stack.

The NIDS processes data-link layer header fields in case of a Wi-Fi frame and extracts features to detect any internal attacks specific to Wi-Fi. The NIDS, then, processes header fields of network, transport, and application layers of Wi-Fi frame or wired packet and extracts features to detect generic network attacks. The functionality of the ML-NIDS is shown as a flow chart depicted in Figure 4.4.

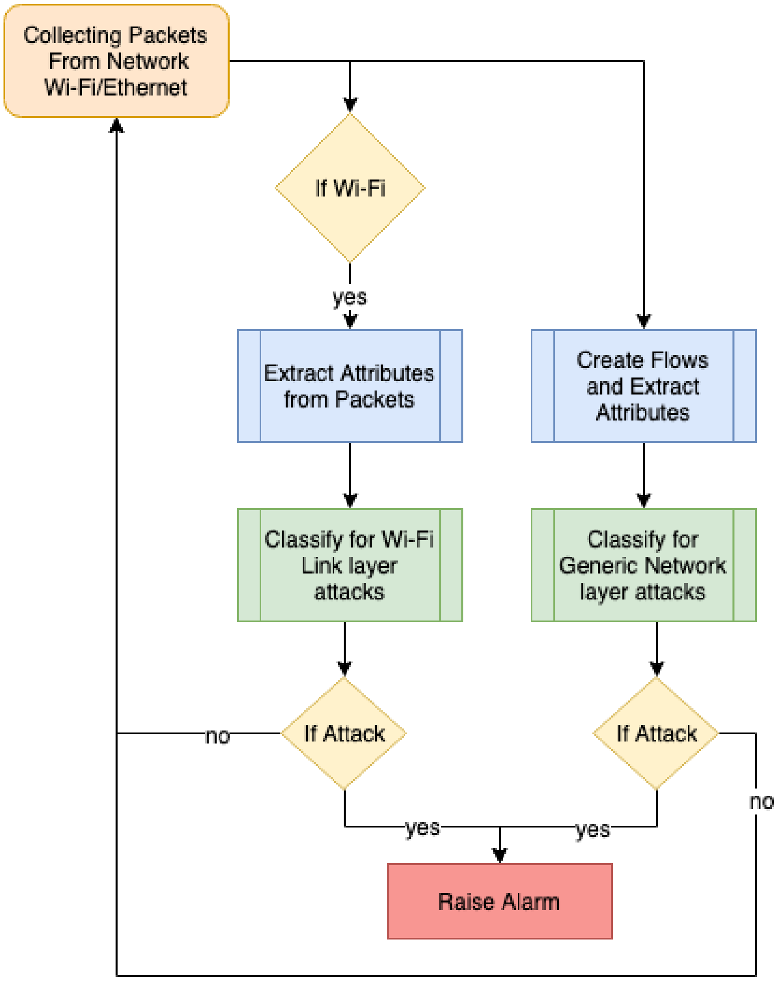

Collecting Packets From Network Wi-Fi/Ethernet
If Wi-Fi
yes
Extract Attributes from Packets
Create Flows and Extract Attributes
Classify for Wi-Fi Link layer attacks
Classify for Generic Network layer attacks
no
If Attack
If Attack
no
yes
yes
Raise Alarm

Fig. 4.4. Flowchart demonstrating the functioning of Multi-Level NIDS.

This section covers the implementation of the ML-NIDS. We start with a brief description of the data that we used. We, then, explain the process of feature extraction and data processing. Finally, discuss the use of the ensemble ML models considered in the study.

4.4.1 Obtaining Attack Traffic for ML-NIDS Implementation

We selected a publicly available dataset, which contains captures of network traffic through an Ethernet connection for attacks and normal network behavior. This dataset is provided by CTU University in Czech Republic. Overall, it consists of 13 captures of normal traffic network mixed with attacks. It is also available as separate captures, normal and attack traffic behavior [43]. We selected one particular CTU dataset that captured network traffic during a malware attack launched using an Ethernet connection. We used another dataset from [44] for normal traffic and DoS attacks as well.

From the experiment mentioned, we captured network traffic using `tcpdump` during the time when the attacks were launched. Since the length of the time we captured was not too long, we decided that these captures were not useful for model training purposes. Therefore, we kept these captures for testing. In this manner, we could also determine how the model would perform if an unrelated dataset is used for testing.

4.4.2 Data Preparation and Features Extraction

In order to use the data for the ML-NIDS development, we needed to prepare the raw data obtained. All raw data manipulated in this study is in the PCAP format. To extract information for these files, we used a tool called "scapy". Scapy is library that permits manipulation of packets to extract their information in the python programming language [45].

The manner in which information is extracted from the captured file to develop an intrusion detection model varies from the way proposed for wireless network in this same study. In this section, it will be referred to the concept of flow to present a procedure that focuses more in finding meaningful information from the dataset used. A flow is a sequence of packets that share similar characteristics in their headers [46]. Flows are commonly used to resolve performance issues, and because of the great amount of information they contain, using flows is the primarily manner in which traffic network is represented for analysis [47]. We used features from [44] listed in Appendix A. These features were derived by expressing statistical values describing characteristics such as direction (incoming/outgoing), size, flags, among many others.

After we used `scapy` to extract the features, we continued to further process the data to see if there was any opportunity reduce the number of features. To determine what features were redundant, we calculated the correlation between them and removed those highly correlated. The heat map shown in Figure 4.5 depicts the correlation status of all 68 features.

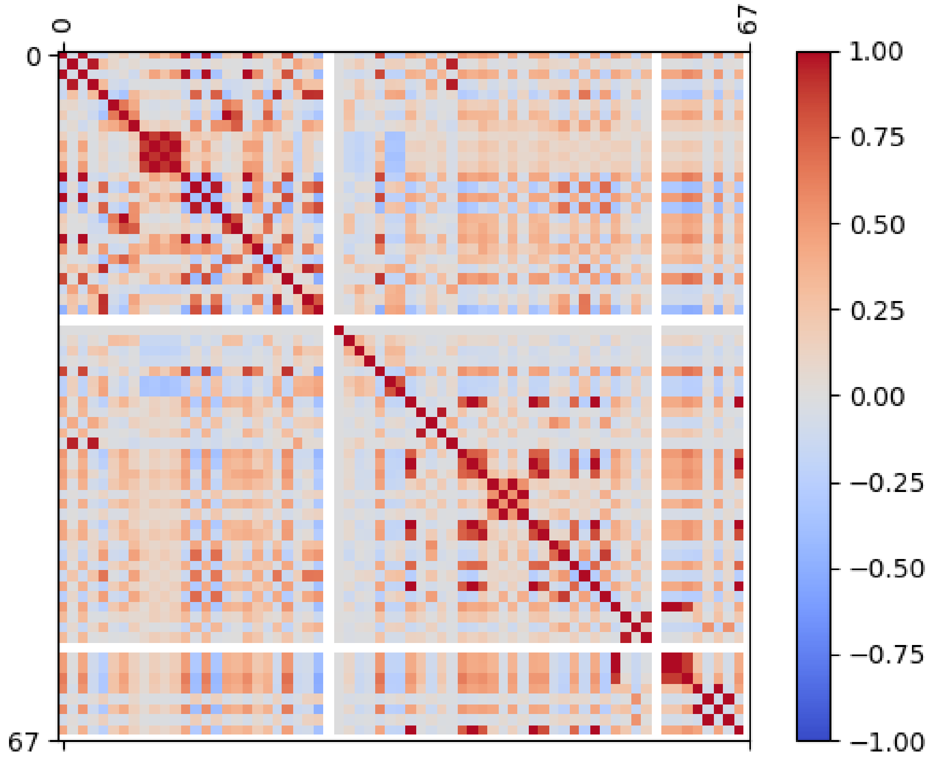


Fig. 4.5. Heat Map of Correlation between Original Features.

We set a threshold of 95%, so from a group of features 95% correlated, we only kept one feature. This process reduced the number of features from 68 to 37. A similar correlation heat map shown in figure 4.6 displays the correlation status of the 37 remaining features.

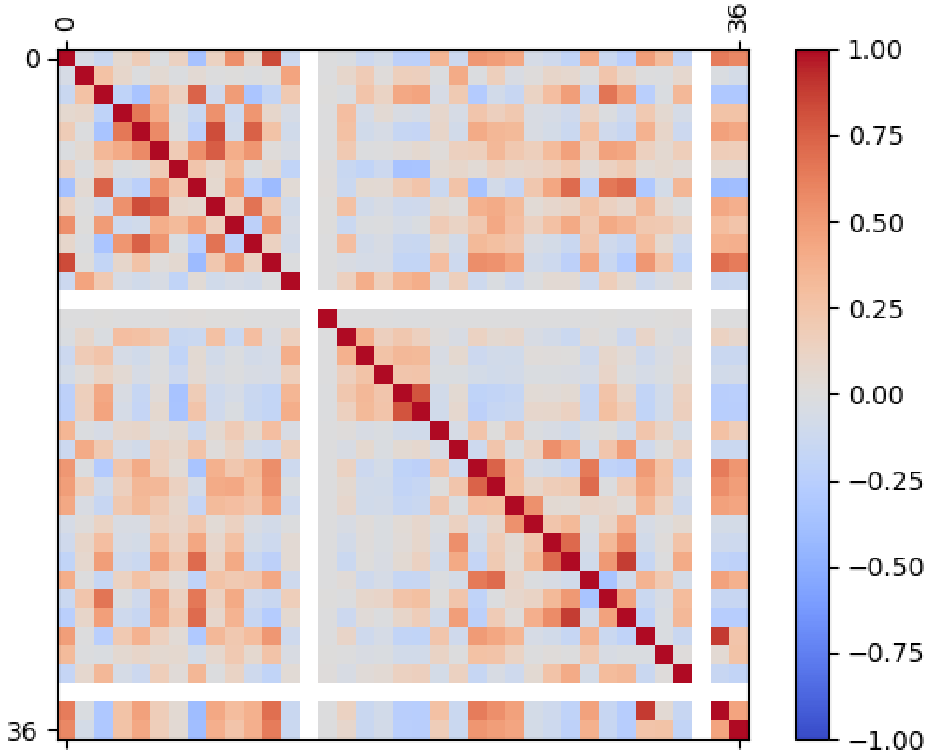


Fig. 4.6. Heat Map of Correlation between Features after Reduction of Features.

4.4.3 Ensemble Machine Learning (ML) Model Development

After the datasets were processed and converted into a format that can be used for training the NIDS model, we proceeded to implement the ML model. The same selection of ensemble ML classification algorithms were used: Random Forest, Bagging, Extra Trees, and XGBoost. We started training the models with the dataset containing all 68 original features. Afterwards, we trained the models with the reduced 37 feature dataset. In the training process, we performed 10 cross validation and calculated the metrics: Accuracy, F1-score, Precision, and Recall. We decided to include these metrics due to the nature of network behavior. In a typical network, attacks are not something that happen very often. Therefore, in captures where a network attack was present, the number packets involved in the attack will be considerably less than the number of packets involved in normal network activity. For this reason, datasets containing network attacks are very unbalanced. In these cases, accuracy can be a deceiving metric to measure the performance of a NIDS. The other considered metrics can help us have a better idea of the performance of a model trained, and tested with very unbalanced datasets.

This implementation corresponds to the second module of the Multi-Level NIDS. The first module being the implementation covered in Chapter 3. Therefore, the functionality of this system is organized in the flow chart shown in Figure 4.4. The data is collected by the monitor node, and if the packets come from a Wi-Fi connection, they are first analyzed to determine if there is a Link Layer attack present. Regardless of the type of connection, Wi-Fi or Ethernet, all packets are analyzed to determine if an attack at the network layer or above is present.

4.4.4 Results

After training the four mentioned ensemble models with the datasets before reducing the number of features, we recorded the performance metrics. These are exposed in Table 4.1. As shown, Random Forest leads all 4 algorithms in all 4 metrics with an accuracy of 99.96%, F1-Score of 99.90%, precision of 99.92, and recall of 99.89%.

Table 4.1.
Classification accuracy on the training dataset

Algorithm	**Accuracy**	**F1 Score**	**Precision**	**Recall**
Random Forest	99.96	99.90	99.92	99.89
Bagging	99.94	99.87	99.89	99.87
Extra Trees	99.94	99.87	99.90	99.85
XGBoost	99.91	99.85	99.90	99.80

From the efficiency point of view, the training times Random forest is the second fastest model with 24.86 seconds after Extra Trees with training time of 14.39 seconds. Bagging took longer to train with 59.12 seconds, and the second slowest was XGBoost with 31.83 seconds.

After decreasing the number of features, the same models were trained and the metrics recorded are shown in figure 4.2. Here we can see that Bagging and XGBoost performed very similarly. Both with accuracy of 99.96%, F1 Score of 99.90%, and precision of 99.85%. Bagging lightly surpasses XGBoost in recall with a 99.85% versus 99.83%. The rest of the models score similar, but slightly lower metrics.

Table 4.2.
Classification accuracy on the training dataset after feature reduction

Algorithm	**Accuracy**	**F1 Score**	**Precision**	**Recall**
Random Forest	99.95	99.89	99.98	99.85
Bagging	99.96	99.90	99.98	99.85
Extra Trees	99.95	99.89	99.97	99.83
XGBoost	99.96	99.90	99.98	99.83

In training time, the fastest model once again was Extra Trees with 13.03 seconds of training time, followed by XGBoost with 18.89 seconds. Random Forest took 20.44 seconds for training, and the Bagging model trained in 36.33 seconds.

The performance metrics improved after the feature reduction and model efficiency increased. The Bagging model improved significantly in performance and efficiency. Therefore, we decided to select it to perform the classification task for the NIDS. We decided to test this model with against 3 different test sets. The first one was generated with the training set, but separated before the training process. The second one was obtained from CTU University. This set is not related in any form to the CTU dataset used to create the training set. The third is network attack capture obtained from our experiment performed in the isolated network previously introduced.

We started with the test set separated from the training set before training process. After running the bagging model for classification, we found that for all 4 metrics this model obtained a perfect performance score. This exceptional result could have been caused due to the fact that this test set was generated together with the set used to train the model. Even though they were separated before the training process, they

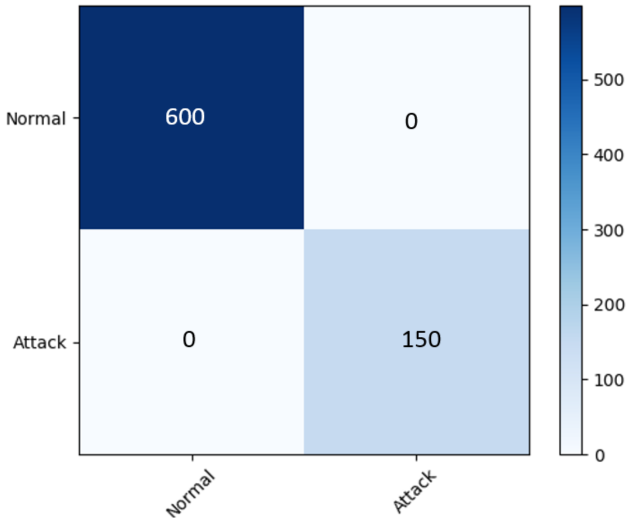


Fig. 4.7. Confusion Matrix for 1st Test Set.

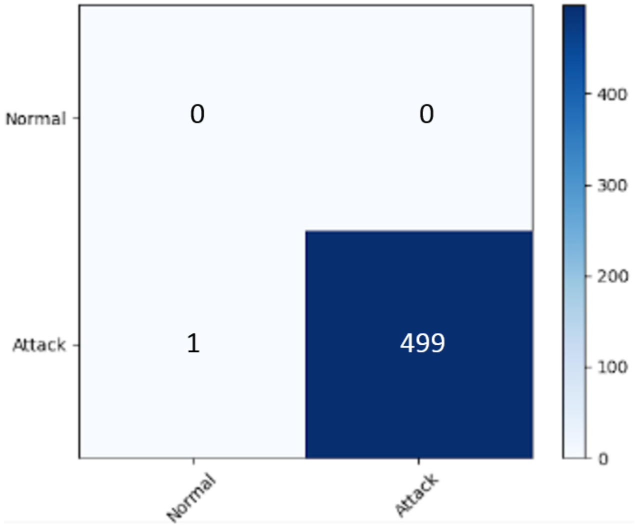


Fig. 4.8. Confusion Matrix for 2nd Test Set.

still share very similar characteristics. The confusion matrix shown in Figure 4.7 exposes classification output of the model when tested with the mentioned set.

We then tested the model with the additional malware capture obtained from CTU. The results after testing the model with this dataset where as follows: 99.79% for accuracy, f1-score of 99.89%, perfect precision score, and 99.79% for recall. The

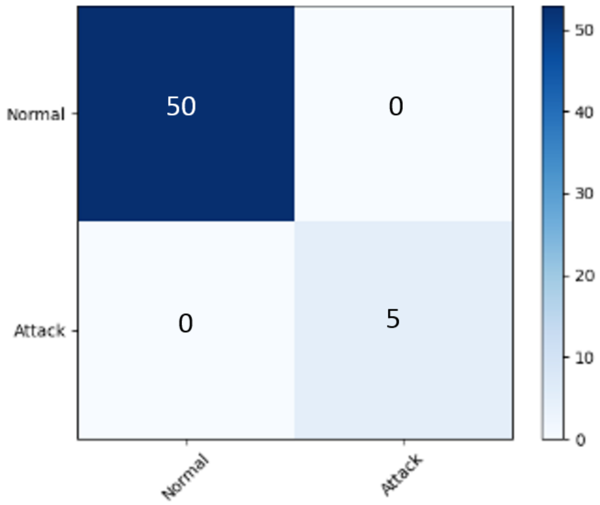


Fig. 4.9. Confusion Matrix for Test set obtained from experiment.

entire dataset was a malware attack, so we can see that very few records were miss-classified. In the confusion matrix in Figure 4.8, we present the classification report for this set. We can also see in the same confusion matrix that almost all records were classified as attacks. This outcome was expected since the entire test set was a capture of a malware attack. Since the miss-classified records were very few, it is hard to identify them in the confusion matrix.

At last, we tested the model against a dataset generated during the experiment mentioned previously. The duration of the attacks launched was not very long and did not generate as much records as desired. Therefore, we combined this network attack capture with normal traffic and balanced to be composed 10% network attack traffic and 90% normal network traffic.

The results reflected perfect performance for this small test set for all metrics considered. Performance with this test is displayed in the confusion matrix shown in Figure 4.9

5. FUTURE WORK AND CONCLUSION

5.1 Future Work

The ML-NIDS has many branches through which it can be expanded and improved. The following are a few considerations for upgrading this work to improve its functionality:

- One of the greatest challenges faced in this study was generating a larger original dataset for training. Even though we were able to set up the desired infrastructure and were able to launch several network attacks, we were not able to collect massive network captures of a wider variety of attacks. Therefore, we consider obtaining more penetration tools to be able to launch a wider variety of network attacks as future work. These tools can help generate a dataset that is more trustworthy.

- Another consideration is to provide this ensemble based multi-level intrusion detection service from the cloud. It has become popular providing cloud based services that we consider this feature as one of the key upgrades. We consider this as a key upgrade because computational resources are expensive, and for data processing tasks hardware requirements are high. By providing this service as cloud based, it can become more accessible for anyone without the hardware resources needed for efficiently processing network data.

- An additional tool that should be considered as future work is the use of Big Data tools for implementation. Network data is considerably large and could be considered as the perfect scenario for the use of Big Data tools such as Apache Spark or Hadoop to provide a distributed approach for tasks such as reading, filtering, and so forth. These tools also provide Machine-Learning libraries that can be used for the implementation of the ensemble models.

5.2 Conclusion

We have been able to implement a NIDS that is capable of detecting generic network attacks and Wi-Fi specific attacks. We also applied techniques to select the most useful features from the datasets used to achieve precise and efficient performances. We also made use of Ensemble Machine-Learning (ML) models to classify network traffic as either normal traffic or malicious network traffic. We implemented the Ensemble ML models separately for the analysis of Wi-Fi specific attacks and generic network attacks.

We determined that the best performing models were Random Forest for Link Layer Attacks and Bagging for Network Layer attacks. It is worth highlighting that the models performed well not only in accuracy, which is the most popular performance measure, but also in F1 score, precision and recall. As mentioned, this additional metrics gave us a more trustworthy judgment of the performance of the models against several imbalanced test sets of different nature. For Link Layer attacks, the accuracy obtained for two class detection was 99.106%. For Network Layer attacks, we obtained perfect score for two of the test sets, and it performed well for the test separated from the training set before the training process with an accuracy of 99.79%, F1 score of 99.89%, perfect precision score, and 99.79% for recall.

In this work, we have also demonstrated that careful attribute selection not only improves efficiency by speeding up the training, and classification process. Attribute selection also helps the model consider useful information that affects directly how records are classified. Thus, ignoring as much noise as possible, the model performs more precisely.

REFERENCES

REFERENCES

[1] N. McDonald, "Digital in 2018: World's internet users pass the 4 billion mark," Jan 2018. [Online]. Available: https://wearesocial.com/blog/2018/01/global-digital-report-2018

[2] "Internet speed today: The evolution of home wi-fi." [Online]. Available: https://www.linksys.com/us/home-wifi-internet-speed-evolution/

[3] B. Marr, "How much data do we create every day? the mind-blowing stats everyone should read," Jul 2018. [Online]. Available: https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/#7707a29c60ba

[4] "Developer help." [Online]. Available: http://microchipdeveloper.com/tcpip:tcp-ip-five-layer-model

[5] I. Muscat, "What are injection attacks? - dzone security," Apr 2017. [Online]. Available: https://dzone.com/articles/what-are-injection-attacks

[6] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, "Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset," pp. 184–208, Firstquarter 2016.

[7] B. Mukherjee, L. T. Heberlein, and K. N. Levitt, "Network intrusion detection," pp. 26–41, May 1994.

[8] A. Mishra and A. Mishra, "Metrics to evaluate your machine learning algorithm," Feb 2018. [Online]. Available: https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234

[9] D. Bisson, "The 10 biggest data breaches of 2018... so far." [Online]. Available: https://blog.barkly.com/biggest-data-breaches-2018-so-far

[10] "Insider threat: 74% of security incidents come from the extended enterprise, not hacking groups." [Online]. Available: https://www.clearswift.com/about-us/pr/press-releases/insider-threat-74-security-incidents-come-extended-enterprise-not-hacking-groups

[11] M. B. Salem, S. Hershkop, and S. J. Stolfo, "A survey of insider attack detection research," Jan 1970. [Online]. Available: https://link.springer.com/chapter/10.1007/978-0-387-77322-3_5

[12] F. Donovan, "45,000 patient records exposed in nuance healthcare data breach," May 2018. [Online]. Available: https://healthitsecurity.com/news/45000-patient-records-exposed-in-nuance-healthcare-data-breach

[13] B. Alotaibi and K. Elleithy, "A majority voting technique for wireless intrusion detection systems," in *2016 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*, April 2016, pp. 1–6.

[14] V. L. L. Thing, "IEEE 802.11 Network Anomaly Detection and Attack Classification: A Deep Learning Approach," in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, March 2017, pp. 1–6.

[15] S. Zaman and F. Karray, "Tcp/ip model and intrusion detection systems," in *2009 International Conference on Advanced Information Networking and Applications Workshops*, May 2009, pp. 90–96.

[16] A. Zainal, M. A. Maarof, S. M. Shamsuddin, and A. Abraham, "Ensemble of one-class classifiers for network intrusion detection system," *2008 The Fourth International Conference on Information Assurance and Security*, 2008.

[17] L. Shen and L. Feng, "An efficient architecture for network intrusion detection based on ensemble rough classifiers," *2013 8th International Conference on Computer Science Education*, 2013.

[18] Y. Wang, Y. Shen, and G. Zhang, "Research on intrusion detection model using ensemble learning methods," *2016 7th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, 2016.

[19] N. N. P. Mkuzangwe, F. Nelwamondo, N. N. P. Mkuzangwe, and F. Nelwamondo, "Ensemble of classifiers based network intrusion detection system performance bound," *2017 4th International Conference on Systems and Informatics (ICSAI)*, 2017.

[20] J. Kurose and K. Ross, *Computer Networking: A Top-Down Approach*, 7th ed. Pearson, 2013.

[21] S. Parthipattu, "802.11 sniffer capture analysis - management frames and open auth," Sep 2017. [Online]. Available: https://community.cisco.com/t5/wireless-mobility-documents/802-11-sniffer-capture-analysis-management-frames-and-open-auth/ta-p/3120622

[22] "Aircrack-ng," https://www.aircrack-ng.org/.

[23] "Metasploit," https://www.metasploit.com/.

[24] "HakTip 19 WiFi 101: Beacon Frames and Injection," https://www.hak5.org/episodes/haktip-19.

[25] "The Dangers of Evil Twin Wi-Fi Hotspots," https://lifewire.com/dangers-of-evil-twin-wi-fi-hotspots-2487659.

[26] "ARP Spoofing," https://en.wikipedia.org/wiki/ARP_spoofing.

[27] "List of Attributes- Wireless Security Dataset Project," http://icsdweb.aegean.gr/awid/attributes.html.

[28] "Ensemble Learning to Improve Machine Learning Results," https://blog.statsbot.co/ensemble-learning-d1dcd548e936.

[29] "ExtraTrees Classifier," https://en.wikipedia.org/wiki/Random_forest.

[30] "Extreme Gradient Boosting," https://xgboost.readthedocs.io/en/latest/ Accessed Sep 28, 2018.

[31] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[32] F. D. Vaca and Q. Niyaz, "An ensemble learning based wi-fi network intrusion detection system (wnids)," in *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*, Nov 2018, pp. 1–5.

[33] "71% of u.s. broadband households have wi-fi or apple airport access." [Online]. Available: http://www.parksassociates.com/blog/article/pr-01102017

[34] M. Rouse, "What is data-link layer? - definition from whatis.com." [Online]. Available: https://searchnetworking.techtarget.com/definition/Data-Link-layer

[35] Tutorialspoint.com, "Network security tutorial." [Online]. Available: http://www.tutorialspoint.com/network_security/index.htm

[36] B. Skerritt and B. Skerritt, "Forcing a device to disconnect from wifi using a deauthentication attack," Jun 2018. [Online]. Available: https://hackernoon.com/forcing-a-device-to-disconnect-from-wifi-using-a-deauthentication-attack-f664b9940142

[37] "Network mitigations." [Online]. Available: http://networkmitigations.blogspot.com/2011/01/layer-3-network-layer-attacks.html

[38] S. Chaudhary, "Evil twin tutorial," Jul 2014. [Online]. Available: http://www.kalitutorials.net/2014/07/evil-twin-tutorial.html

[39] "Home." [Online]. Available: http://www.hackingloops.com/penetration-testing-of-men-in-middle-attacks-using-arp-spoofing/

[40] R. Sankar, "Wifi stress testing using mdk3, beacon flooding deauthentication attack." Jun 2018. [Online]. Available: https://kalilinuxtutorials.com/mdk3/

[41] C. Hofer and R. Wampfler, "Ip spoofing - portal." [Online]. Available: http://rvs.unibe.ch/teaching/cn applets/IP_Spoofing/IP Spoofing.pdf

[42] "Ping flood (icmp flood)." [Online]. Available: https://www.incapsula.com/ddos/attack-glossary/ping-icmp-flood.html

[43] "Ctu-13 dataset." [Online]. Available: https://www.stratosphereips.org/datasets-ctu13/

[44] Q. Niyaz, W. Sun, and A. Y. Javaid, "A deep learning based ddos detection system in software-defined networking (SDN)," *CoRR*, vol. abs/1611.07400, 2016. [Online]. Available: http://arxiv.org/abs/1611.07400

[45] "Scapy." [Online]. Available: https://scapy.readthedocs.io/

[46] "Traffic flow (computer networking)." [Online]. Available: https://www.revolvy.com/page/Traffic-flow-(computer-networking)

[47] K. Conklin, "What is network flow monitoring?" Dec 2018. [Online]. Available: https://www.whatsupgold.com/blog/network-monitoring/what-is-network-flow-monitoring/

APPENDIX

A. ATTRIBUTES EXTRACTED FOR HIGHER LAYER ATTACKS DETECTION

Table A.1.
Features for TCP flows [44]

#	**Attribute name**
0	Number of Incoming TCP Flows
1	Ratio of TCP Flows over Incoming Flows
2	Number of Outgoing TCP Flows
3	Ratio of TCP Flows over Outgoing Flows
4	Ratio of Symmetric Incoming TCP Flows
5	Ratio of Asymmetric Incoming TCP Flows
6	Number of Distinct Source IP in Incoming TCP Flows
7	Entropy of Source IP Incoming TCP Flows
8	Bytes for Incoming TCP Flow
9	Bytes for Outgoing TCP Flow
10	Number of Packets Incoming TCP Flow
11	Number of Packets Outgoing TCP Flow
12	Number of Distinct Window Size Incoming TCP Flows
13	Entropy of Window Size Incoming TCP Flows
14	Number of Distinct TTL values Incoming TCP Flows
15	Entropy of TTL values Incoming TCP Flows
16	Number of Distinct Source Ports Incoming TCP Flows
17	Entropy of Source Ports Incoming TCP Flows
18	Number of Distinct Dest Ports Incoming TCP Flows
19	Entropy of Dest Ports Incoming TCP Flows
20	Ratio of Dest Ports $\leq$ 1024 Incoming TCP Flows
21	Ratio of Dest Ports $>$ 1024 Incoming TCP Flows
22	Ratio of Incoming TCP Flows SYN Flag Set
23	Ratio of Outgoing TCP Flows SYN FlagSet
24	Ratio of Incoming TCP Flows ACK FlagSet
25	Ratio of Outgoing TCP Flows ACK FlagSet
26	Ratio of Incoming TCP Flows URG Flag Set
27	Ratio of Outgoing TCP Flows URG FlagSet
28	Ratio of Incoming TCP Flows FINF lagSet
29	Ratio of Outgoing TCP Flows FIN Flag Set
30	Ratio of Incoming TCP Flows RST Flag Set
31	Ratio of Outgoing TCP Flows RST Flag Set
32	Ratio of Incoming TCP Flows PUSH Flag Set
33	Ratio of Outgoing TCP Flows PUSH FlagSet

Table A.2.
Features for UDP flows [44]

#	**Attribute name**
0	Number of Incoming UDP Flows
1	Ratio UDP Flows over Incoming Flows
2	Ratio UDP Flows over Incoming Flows
3	Number of Outgoing UDP Flows
4	Ratio UDP Flows Over Outgoing Flows
5	Ratio of Symetric Incoming UDP Flows
6	Ratio of Asymmetric Incoming UDP Flow s
7	Number Distinct Source IP Incoming UDP FLows
8	Entropy Source IP Incoming UDP Flows
9	Bytes Incoming UDP Flow
10	Bytes Outgoing UDP Flow
11	Number Packets Incoming UDP Flow
12	Number Packets Outgoing TCP Flow
13	Number Distinct Source Ports Incoming UDP Flows
14	Entropy Source Ports Incoming UDP Flows
15	Number Distinct Dest Ports Incoming UDP Flows
16	Entropy Dest Ports Incoming UDPFlows
17	Ratio Dest Ports $\leq$ 1024 Incoming UDP Flows
18	Ratio Dest Ports $>$ 1024 Incoming UDP Flows
19	Number Distinct TTL values Incoming UDP Flows
20	Entropy TTL values Incoming UDP Flows

Table A.3.
Features for ICMP flows [44]

#	**Attribute name**
0	Number of Incoming ICMP Flows
1	Ratio ICMP Flows Over Incoming Flows
2	Number of Out going ICMP Flows
3	Ratio ICMP Flows Over Outgoing Flows
4	Ratio of Symetric Incoming ICMP Flows
5	Number of Asymmetric Incoming ICMP Flows
6	Number Distinct Source IP Incoming ICMP FLows
7	Entropy Source IP Incoming ICMP Flows
8	Bytes Incoming ICMP Flow
9	Bytes Outgoing ICMP Flow
10	Number Packets Incoming ICMP FLow
11	Number Packets Outgoing ICMP Flow
12	Number Distinct TTL values Incoming UDP Flows
13	Entropy TTL values Incoming ICMP Flows