Design and Development of an Efficient Network Intrusion Detection System using Ensemble Machine Learning Techniques for Wifi Environments

Abhijit Das
Research Scholar, VTU-Belagavi, CSE, PESITM, Shivamogga
Asst. Prof., Dept. of CSE, BNMIT, Bangalore, India

Pramod
Associate Professor, Dept. of ISE
PES Institute of Technology & Management
Affiliated to VTU, Shivamogga, India

Abstract—Intrusion Detection Systems (IDS) are vital for computer networks as they protect against attacks that lead to privacy breaches and data leaks. Over the years, researchers have formulated IDS using machine learning (ML) and/or deep learning (DL) to detect network anomalies and identify attacks. Network Intrusion Detection Systems (NIDS) within corporate networks is a form of security that detects and generates an alarm for any cyberattacks. In both academia and industry, the concept of deploying a NIDS has been studied and adopted. The majority of NIDS research, on the other hand, has focused on detecting threats that emerge from outside of a wired connection. In addition, the NIDSs recognize Wi-Fi and wired networks alike. The Wi-Fi network’s accessible connectivity distinguishes this from the wired network. A wired connection is highly resistant to many insider threats that could occur on a Wi-Fi router. A conventional view to developing NIDSs may miss malicious activities. This paper aims to design a multi-level NIDS for Wi-Fi predominant networks to identify both organizational Wi-Fi networks malicious activity and standard network malicious activity. Wi-Fi devices are common on campuses and businesses, and they are incorporated into the fixed wired network at the gateway. Wi-Fi networks are the primary target for this implementation; however, they are also designed to function in wired environments. For the Multi-Level NIDS, the proposed model used an ensemble learning method that pools the strengths of multiple weak learners into a single strong learner.

Keywords—Machine learning; ensemble learning; intrusion detection system; Wi-Fi security

I. INTRODUCTION

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

The 2022 Global Digital suite has reported that Internet users have reached 4.6 billion, as shown in Figure 1 [1]. At the beginning of the COVID-19 pandemic, one of the most talked-about stories was how much the world relied on the internet, especially as countries went into lockdown. Although the fluctuations in restrictions on movement over the last two years, the most recent data show that people are actually spending more time than ever before using connected technology. 5.3 billion people will be online by 2023, according to a report cited by the source. Over the period of 2018 to 2023, compound annual growth is 6%. A growth rate of 7.7 % from 2018 is expected in 2019 when there will be an additional 300 million internet users. It is also revealed that 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. 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.

The Internet Stack: The Internet protocol stack needs to be exposed to demonstrate the risks involved in blindly putting the trust in modern data transmission and management, which depicts the layers that describe the phases through which the data traverses between the communicating parties. Each layer fulfils its purpose and integrates with the others to provide the data transmission service. In Fig. 2, each layer could be explored as to its own world; nevertheless, this work aims to expose their main role in exchanging information, their characteristics, and the weaknesses that attackers commonly target at the highest level of the stack, application layer, which

Fig. 1. Internet and Social Media Accessibility Statis [1].
interacts directly with the users and programs. Right below appears the Transport layer that offers services to network applications. The network layer makes communication possible by facilitating Network addressing and routing. The Data Link layer is responsible for data transmission between devices belonging to the same Network. Finally, the physical layer refers to the actual hardware employed for the transmission. Knowing these layers’ functionality 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, the focus was to set, Network, and Data Link layers. Studying, thoroughly these layers help 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.

| Layer # | Layer Name   | Protocol | Protocol Data Unit | Addressing |
|--------|--------------|----------|--------------------|------------|
| 5      | Application  | HTTP, SMTP, etc. | Messages          | n/a        |
| 4      | TCP/UDP      |          | Segments/          | 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. 2. A Depiction of the 5 Layers of the Internet Stack Described by [2].](image)

**Classes of Network Attacks:** The objective of flooding attacks is usually to generate a denial of service effect on the victim. Flooding attacks overwhelm the communication service with a considerable amount of traffic, which most of the time results in the collapse of an already established communication between client and server. Another network attack taken into account is the 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 an 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 [3]. Providing details regarding these attacks, their mechanism, and how they relate to the characteristics of the link layer in wireless networks is key in discovering the effective implementation of an IDS. Enlightening research work regarding 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 [4]. The background research made as part of the generation of this dataset provided basic network attack information and specifications for types and classes of attacks, which are very popular among network attackers.

**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 IDS, which is ML. Nowadays, ML has become a very popular technology used in several fields, including 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 [5]. 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 to design learning models in an ensemble fashion as the combination of several ML models causes every model to help each other in correcting weaknesses. Therefore, the ensemble model will present an improved performance compared to the performance of each model considered alone. The outcome of including ensemble learning in network security is 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 [6] in any instance where the situation may happen.

It is critical as well to highlight that the technology of ML provides a precise way in which the effectiveness of a model can be measured. Most ML packages normally provide accuracy, which is considered in classifying if the network traffic in a normal state or an attack is present, which becomes of great importance. Nevertheless, several other metrics can present deeper information about how effectively the model detects intruders. For example, logarithmic cost, F1 score, mean absolute error or mean squared error, among many others. These 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 other class samples [7]. 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 the 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 behaviour incorrectly were attacks. This result is clearly misleading and strongly suggests that in the case of IDS development, there needs to be included as many precision metrics as possible.

During last year, considering up to the 10 most disrupting events regarding cyberattacks, they compromised 5 million records with sensitive credit information in the mildest of these events up to breaching 1 billion records in the worst case [8]. 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 personal network setup does not restrict the findings to benefit these types of networks. 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 one type in an organization as they are traditionally in a personal network. Therefore, it is a more significant benefit to study the development of an IDS capable of scanning the behaviour of a network with a structure that combines wired and wireless networks. Even though the data, which was previously exposed, is comprehensive and provides a considerable amount of data to develop an effective model, it lacks the combination of different types of networks that, as mentioned before, are more common in the organization. An experimental setup that includes these requirements is necessary to simulate the performance of an IDS against network threats presented in this network architecture. The result of the combination of these two different types of networks is the motivation for the
development of an IDS capable of scanning and analyzing not only wireless traffic but also Ethernet traffic. The inclusion of wired networks allows this research to include attacks in this type of network. 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 than 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 a frame for wireless connection. A different way of analyzing the traffic for the layers above the Link layer is employed. A unit flow is used, a condensed structure that describes network traffic characteristics 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 are 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 wired network attacks or wireless attacks. Moreover, another component should focus its analysis on link-layer attacks and perform its analysis and attack detection on wireless frames.

Network attacks that are performed either in the 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 network’s structure, whether it is a home network set up or enterprise network set up; or it can stress more the Internet layer that the intruder aims 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 74% of security incidents resulted from the extended enterprise level. Of this 74%, 42% represent actual inadvertent employees. A total of 74% is reached considering, besides employees, customers, and suppliers, entities that are known to the company. The remaining 26% is attributed to parties unknown to the organization. Insider attacks are not a current issue. Another study refers to a survey made to United States security personnel, where insider incidents were cited 59% of the time [9].

II. RELATED WORK

This section briefly reviews the literature for the current work, where researchers present the design and implementation of NIDS.

A. Wi-Fi Focused Research

Koliag et al. [4] extensively studied the attacks against Wi-Fi networks and categorized them. The contribution that proves to be the highlight of this research is the 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 ML algorithms. There are several characteristics that made this work stand out from similar research made in the field. The authors provided a wide overview of attacks being performed in the 802.11 standards 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 behaviour and behaviour that describes a network attack. For the study of Wi-Fi IDSs, 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 a 96.20% detection rate with all the features in the dataset and 96.26% with 20 features. Alotaibi et al. [10] attempted to improve the accuracy by applying the majority voting technique in which several ML 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 improves performance and is used for feature selection. The proposed solution was based on the combination of several machine-learning algorithms to learn patterns for different network behaviours. The initial procedure where patterns are constructed is called the 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 leave only those that contribute to the classification. This work used the similar ensemble algorithms that have been used in this work, the combination of machine-learning algorithms, with a different approach and features. The participating algorithms used were Bagging, Random Forest(RF), 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 Koliag et al. and reported an accuracy of 96.32%.

The literature review also found studies that make use of the DL approach. In [11], 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 to learn what features were
the most useful to determine patterns. These patterns aided in classifying network traffic behaviour 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 DL 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% was obtained after the mentioned techniques were applied. However, information in relation to data preparation was very limited. More than half of the features in the AWID dataset contain missing values. It’s also worth noting that there are a variety of features in the dataset, including hexadecimal characters and numerical values. Because of this, the dataset pre-processing is essential before using it in a model. The lack of attention to data preparation has been noted in the literature. On the other hand, this study shows that the data preparation for this work was meticulous.

B. Research Focused on Ensemble Models and Multi-level Intrusion Detection

Zaman et al. [12] proposed a more detailed perspective of network traffic analysis for intrusion detection. In this work, the authors focused on the different layers of the Internet stack. They proposed four different solutions that correspond to attacks analyzed from the perspective of the 4 upper layers in the Internet Stack. The IDSS 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 99.84% for the Transport layer IDS, which used Neural Networks for classification. Considering all layers, the best accuracy was 99.41% using a Neural Network classifier.

Zainal et al. [13] proposed an approach where classifiers with different learning paradigms are combined into a single Ensemble model. The paradigms employed are Linear Genetic Programming, Adaptive Neural Fuzzy Inference System, and RF. The authors’ objective is to enhance the precision and lower the false alarm rate in IDS. Two principal steps were exposed in this work. First, select relevant features. Second, developing an ensemble model composed of classifiers with different learning paradigms. They demonstrated that the performance of the ensemble model surpassed the performance of the three models used separately for all types of attacks.

Li et al. [14] presented the use of Rough Set Theory 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. They used rough Set Theory to implement a rough meta-learning classification strategy. This strategy combined multiple rough learning methods. After the experiment, the author demonstrated that the 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. [15] propose 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, the authors mentioned that the KDDcup99 dataset was used. One of the main objectives of this work was tackling the unbalanced nature of this dataset using ensemble learning. The model is evaluated using receiver operating characteristic (ROC) curves. The authors computed the area under the ROC curves (AUC) for more specific results. In the results, it was found that the ensemble model outperforms the single based models.

In [16], Nenekazi et al. uncover 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, the weak ensemble classifier, 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 range of 0.045651 and 0.25615 when accuracy will reach as much as 90%. A gradient boosted machine was used by Tama and Rhee [17] to increase the detection accuracy of anomaly-based IDS. Gradient boosted machine’s best results are achieved by a grid search of input parameters. Rezvy et al. [18] used the AWID dataset that they used in [19]. They combined autoencoder frameworks with feed-forward neural networks. 99% of the time, their model can classify everything correctly. Faik Kerem Ors et al. [20] have developed a ML-based Wi-Fi IDS to protect IoT devices better. A single multi-class classifier operating on encrypted data from the Wireless Data Link Layer demonstrates that the benign traffic and six types of IoT attacks can be identified with an overall accuracy of 96%.

III. METHODOLOGY

The proposed installation of NIDS for the Wi-Fi network is shown in Fig. 3. The NIDS for the Wi-Fi network will receive Wi-Fi frames from the NIDS sensors for intrusion detection. As the NIDS is implemented using a ML approach, the model needs to train and validate it before deployment using the existing Wi-Fi frames dataset; it is the AWID dataset in this case. An overview of the various ensemble algorithms employed in the model development process is also offered here. Next, the algorithms will be applied to training data to identify the best one, and their evaluation of test data has been discussed in detail. Fig. 3 depicts the NIDS implementation process.

A. Datasets

The AWID Wi-Fi intrusion data was gathered in a typical Wi-Fi network environment. All of the devices connected to the Wi-Fi network were connected to a single access point for the Internet connection. A single intruder machine launched the attacks. Wi-Fi frames were captured using a computer in monitor mode. It was equipped with high processing capabilities to be able to capture a large number of frames at a
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. Both threat-class and threat-specific versions of the AWID dataset are available. Every record in the dataset is classified as either flooding, impersonation, injection attacks or normal. The identical information is labelled with threat specific class for 17 different Wi-Fi threats or benign. Many devices can use this available enormous or small version of this data format. In total, there are four AWID datasets available. The threat-class version of the dataset is used in this work. Table I shows the distribution of the records in the dataset. The AWID dataset has an imbalanced allocation, with more than 90% of the records in train and test sets being normal. As it can be seen, there is a significant difference in features and labels between large and small sets. The smaller dataset was collected separately from the larger dataset using various methods and techniques. As a result, there is no connection between them other than the sheer number of features and labels. Each record in the AWID dataset is a Wi-Fi frame with 155 features. 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. Control, management, and data frames are also included in Wi-Fi frames. Therefore, not every feature will apply to every frame. Therefore, some records may have missing values for those features.

TABLE I. TRAFFIC RECORDS DISTRIBUTION FOR THE TRAINING AND TEST SETS IN THE REDUCED AWID DATASET [10]

|                | Training % | Testing % |
|----------------|------------|-----------|
| Normal         | 1633190 (91%) | 530785 (92.2%) |
| Flooding       | 48483 (2.6%) | 8096 (1.3%) |
| Injection      | 65377 (3.5%) | 16684 (2.8%) |
| Impersonation  | 48521 (2.6%) | 20078 (3.1%) |
| Total          | 1795565 (100%) | 575643 (100%) |

B. Preparation of Data

In the AWID dataset description, some values of the features in each record may represent noise for the intrusion detection task or maybe missing values. As a result, preprocessing the dataset is absolutely necessary. The dataset contains 154 features for each record and one target class (a total of 155 features). Features with zero variance, i.e., features whose value was the same across all records, have been eliminated. The AWID dataset description contained 27 features, some of which may represent noise or missing values for the intrusion detection task; because of this, preparing the dataset is a must. Each record in the dataset has 154 features and a single target class (a total of 155 features). A feature that had the same value in all data was eliminated because it had zero variance. Statistically, 27 characteristics have been found that had no statistical variation. Attributes with less than half of their values were removed. Following these steps, there are now 36 features in Table II. It was necessary to replace an attribute’s incomplete data with its most common value. Access points and receivers have MAC addresses of 29, 30, 31, 32, and 33 for Wi-Fi adapters. They constantly adapt to the changing needs of the machine. The values of all of these features were changed to indicate whether or not the device had an address. Features 0, 1, 2, 3, and 13 were omitted because this deployment does not make use of time series. In order to achieve success, prediction is handled on an independent record basis. Considering that this feature was a combination of Features 20 and 21, it was omitted from the final product.
Feature 35 was also left out due to the fact that it indicates the sequence identifier.

**TABLE II. AFTER PRE-PROCESSING FEATURES REMAINED IN THE AWID DATASETS**

| No. | Features name | Definitions |
|-----|---------------|-------------|
| 0   | frame.time.epoch | Time of epoch |
| 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.tsf | TSFT |
| 8   | radiotap.present.channel | Channel |
| 9   | radiotap.present.dbm.antsignal | dB Antenna Signal |
| 10  | radiotap.present.antenna | Antenna |
| 11  | radiotap.present.rxflags | RX flags |
| 12  | radiotap.mactime | MAC timestamp |
| 13  | radiotap.datarate | Data rate (Mbps) |
| 14  | radiotap.channel.freq | Channel frequency |
| 15  | radiotap.channel.type.cck | Complementary Code Keying |
| 16  | radiotap.channel.type.ofdm | Orthogonal Frequency-Division Multiplexing |
| 17  | radiotap.dbm.antsignal | dB Antenna Signal |
| 18  | wlan.fc.type.subtype | Type/Sub-type |
| 19  | wlan.fc.type | Type |
| 20  | wlan.fcsubtype | Sub-type |
| 21  | wlan.fc.ds | DS status |
| 22  | wlan.fc.frag | More Fragments |
| 23  | wlan.fc.retry | Retry |
| 24  | wlan.fc.powermgnt | Power Management |
| 25  | wlan.fc.moredata | More Data |
| 26  | wlan.fc.protected | Protected flag |
| 27  | wlan.duration | Duration |
| 28  | wlan.rx | Receiver address |
| 29  | wlan.da | Destination address |
| 30  | wlan.seq | Sequence id |
| 31  | wlan.sa | Source address |
| 32  | wlan.bssid | BSSID Id |
| 33  | wlan.fcfrag | Fragment number |
| 34  | wlan.fcfrag | Fragment number |
| 35  | wlan.ta | Transmitter address |

Finally, the correlation between the attributes has been calculated after following these steps. It has been discovered that a few of the characteristics are strongly linked to each other. Fig. 5 depicts the feature correlation heat map. A strong positive correlation can be seen in the figure between the features numbered 4, 5, 6, 7, 8, 9, 10, 11, and 12, as well as 30, 31, 32, and 33, where the corresponding correlation value is 1. A strong negative correlation exists between features 16 and 17, with a correlation value of -1. One feature has been chosen from that group and discarded the rest for each group.

There were four features that were chosen from the groups: Features 4, 6, 30, and 16. Thus the model came up with a new list of 18 features after going through all of these steps. Table II displays these features in italic type.

**C. Ensemble Algorithms**

An ensemble model is a technique that utilizes various algorithms or combines them to enhance the strength and efficiency of any of the components’ algorithms. Syafri et al. (2012) [21] proposed that the benefit of ensemble methods is that they can be adjusted more appropriately than single models in any modifications in the controlled data stream.

1) **Bagging:** Unlike Boosting, the Bagging algorithm can be used for both classification and regression. In 1996, Leo Breiman developed the Bagging algorithm, which is also known as Bootstrap aggregating (Breiman, 1996) [22]. It was developed to improve the algorithm accuracy by producing diverse models and then combining them to produce the final model. Breiman’s experiments indicated that Bagging could improve the efficiency of inconsistent learners but reduce the efficiency of a steady one when it was first established (Breiman, 1996) [22]. Changing the training data in the unstable learner can significantly impact the hypothesis it generates (Dietterich, 2002) [23]. The experimental results reveal that Bagging works better than Boosting and Randomization in the presence of noise (Dietterich, 2000b) [24].

2) **Random Forest:** It is a decision tree-based variation of bagging. A decision tree’s learning ability allows RF to extract precise information considering all aspects. This results in a precise performance. But it can also lead to excessive variation and over-fitting. Together with the bagging algorithm’s principles, this strategy could fix the issue. Bagging in RF creates random subsets of data. RF validates that samples from one subset have less correlation than samples from other subsets. Each subgroup has a decision tree. A decision tree’s training subset is used to classify data. Finally, the most common occurrence from each decision tree is picked as the overall classification output. The fact that RF randomly chooses a portion of features from each decision tree’s feature set increases performance over bagging.

3) **Extra Trees:** The RF algorithm has a lot in common with this technique. Decision trees are at the heart of both approaches. When comparing it to RF, the main distinction is that it adds more randomness. Random features are scoured for each decision tree, and the locally best feature/split combination is computed. A random number is chosen for the split in ET, unlike RF. Over-fitting is less likely with this approach, which broadens the application of the algorithm.

4) **XGBoost:** The Gradient Boosting (GB) technique is effectively implemented in XGBoost. Multiple weak learners are sequentially used in the GB. Each learner focuses on those samples that the previous learner misclassified in the sequence during the training process. 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 tackled 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 well-known for its ability to learn, the performance comes at the expense of considerable computational resources. These problems led to the creation of a set of tools called XGBoost that can help with the computational load of GB.

D. Implementation of Multi-Level NIDS

Fig. 6 shows a Wi-Fi dominant campus network prototype for the proposed ML-NIDS implementation with a connected Gateway switch. The Gateway switch has several Ethernet ports. Wi-Fi access points (AP) and other switches for local sub-networks can be connected using these ports. 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.

Fig. 6. Proposed IDS Model for Wi-Fi.

The NIDS processes data-link layer header fields in the case of a Wi-Fi frame and extract 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 Fig. 7.

Fig. 7. Flowchart Demonstrating the Functioning of Multi-Level NIDS.

E. Obtaining Attack Traffic for ML-NIDS Implementation

A publicly available dataset has been selected, which contains captures of network traffic through an Ethernet connection for attacks and normal network behaviour. CTU University in the Czech Republic provides this dataset. Overall, it consists of 13 captures of normal traffic networks mixed with attacks. It is also available as separate captures, normal and attacks traffic behaviour. One particular CTU dataset has been selected that captured network traffic during a malware attack launched using an Ethernet connection. The model also used another dataset from [25] for normal traffic and DoS attacks. From the experiment mentioned, network traffic has been captured using tcpdump during the time when the attacks were launched. Since the length of the time captured was not too long, it has been decided that these captures were not useful for model training purposes. Therefore, these captures have been kept for testing. In this manner, it is possible to determine how the model would perform if an unrelated dataset is used for testing. Data Preparation and Features Extraction

In order to use the data for the ML-NIDS development, the raw data obtained is needed to prepare. All raw data manipulated in this study is in the PCAP format. To extract information for these files, "scapy" tool has been used. Scapy is a library that permits the manipulation of packets to extract their information in the python programming language. The manner in which information is extracted from the captured file to develop an intrusion detection model varies from the way proposed for a wireless network in this same study. This section will refer to the concept of flow to present a procedure that focuses more on finding meaningful information from the dataset used. A flow is a sequence of packets that share similar characteristics in their headers. Flows are commonly used to resolve performance issues. Because of the great amount of information they contain, using flows is the primary manner in which traffic network is represented for analysis. The features were derived by expressing statistical values describing characteristics such as direction (incoming/outgoing), size, flags, among many others. After that scapy was used to extract the features, it continued to process the data further to see if there was an opportunity to reduce the number of features. To determine what redundant features were redundant, their correlation and removed those highly correlated features. The heat map shown in Fig. 8 depicts the correlation status of all 68 features.

A threshold of 95% has been set, so from a group of features 95% correlated, only one feature has been kept. This process reduced the number of features from 68 to 37. A similar correlation heat map shown in Fig. 9 displays the correlation status of the 37 remaining features.
Fig. 8. Heat Map of Correlation between Original Features.

Fig. 9. Heat Map of Correlation between Features after Reduction of Features.

F. Ensemble Machine Learning Model Development

After the datasets were processed and converted into a format that can be used to train the NIDS model, the ML model was implemented. RF, Bagging, Extra Trees, and XGBoost were used as ensemble ML classification algorithms [27]. Training the models has been started with the dataset containing all 68 original features. Afterwards, the model was trained with the reduced 37 feature dataset. In the training process has been performed with 10 cross-validations and calculated the metrics: Accuracy, F1-score, Precision, and Recall.

It has been decided to include these metrics due to the nature of network behaviour. In a typical network, attacks are not something that happens very often. Therefore, it captures where a network attack was present; the number of 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 better understand the performance of a model trained and tested with very unbalanced datasets. This implementation corresponds to the second module of the Multi-Level NIDS. Therefore, the functionality of this system is organized in the flow chart shown in Fig. 7. The monitor node collects the data, 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.

IV. Result and Discussion

Models for the AWID training dataset were developed using the ensemble method depicted in Fig. 4. Models are built using sci-kit-learn, a Python-based ML tool. The models for each algorithm were developed using a 10-fold cross-validation process. The best model was chosen by averaging the accuracy of all of the algorithms, each using a distinct set of random states. Each algorithm’s classification accuracy is shown in Table III. The most accurate model was the RF one. This NIDS model for the Wi-Fi network has been used as the last model for testing the dataset.

| Ensemble Technique | Accu% |
|--------------------|-------|
| Bagging            | 98.967|
| ExtraTrees         | 99.018|
| RF                 | 99.097|
| XGBoost            | 98.944|

This work evaluated the prediction results of different experiments using the test scores produced from the confusion matrix [26]. The confusion matrix is a two-dimensional matrix that represents the correlation of true conditions and predictive results shown in Table IV. TP describes the number of abnormal samples being accurately classified. TN defines the number of normal samples being accurately classified. FP specifies the number of normal samples being falsely classified as abnormal samples. FN specifies the number of abnormal samples being falsely classified as normal samples. Various test scores were calculated using the confusion matrix in this work: Accuracy, Precision, Recall, and F1 Score.

\[
\text{Accuracy(Acu)} = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)
\]

\[
\text{Precision} = \frac{TP}{TP + FP} \quad (2)
\]

\[
\text{Recall} = \frac{TP}{TP + FN} \quad (3)
\]

\[
\text{F1Score} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} \quad (4)
\]

The RF model was 95.873% accurate in classifying the records. Fig. 10 depicts the model’s confusion matrix. Each of the traffic classes has its own set of average values for precision, recall, and f1-score has shown in Fig. 11. Due to its f1-score of 0.13, the RF model’s impersonation attack performance was poor. Fig. 10 shows that most impersonation threats are classified as injection threats. The model has an f1-score of 0.95 and intermediate accuracy, recall, and F1-score of...
The majority of the impersonation assault records were classed as "normal" even though the accuracy was slightly lower than that of [4] and [10]. In this regard, the proposed RF model outperforms theirs because it classifies the impersonation attack as a different type of attack. In addition, [4] reported a lower f1-score than the proposed model in this work. The model has an f1-score of 0.95 and intermediate accuracy, recall, and F1-score of 0.96. If flooding, impersonation, and injection threats are considered the sole attacks, it can map the confusion matrix of Fig. 10 to Fig. 12. When it comes to determining if the information is either malicious or benign, the model obtained 99.996% accuracy. To put it another way, in this context, [4] and [10] were both 96.28% and 96.28% accurate when it came to this question.

![Fig. 10. Confusion Matrix for the 4-class Classification Model.](image1)

The performance metrics were recorded after training the four mentioned ensemble models with the datasets before reducing the number of features. These are exposed in Table V. As shown, RF leads all 4 algorithms in all 4 metrics with an accuracy of 99.96%, F1 Score of 99.90%, the precision of 99.92, and recall of 98.99%

![Fig. 11. Performance Metrics for Each Class.](image2)

| Algorithm   | Accuracy | F1 Score | Precision | Recall |
|-------------|----------|----------|-----------|--------|
| Random Forest | 99.96    | 99.89    | 99.92     | 99.89  |
| Bagging     | 99.96    | 99.90    | 99.98     | 99.85  |
| Extra Trees | 99.95    | 99.90    | 99.98     | 99.85  |
| XGBoost     | 99.96    | 99.90    | 99.98     | 99.98  |

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. RF 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, it has been decided to select it to perform the classification task for the NIDS. This model has been tested against three 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. The proposed model started with the test set separated from the training set before the training process. After running the bagging model for classification, it was found that for all four 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 still share very similar characteristics. The confusion matrix shown in Fig. 13 exposes the classification output of the model when tested with the mentioned set.

![Fig. 12. 4-Class to 2-Class Confusion Matrix Conversion.](image3)

![Fig. 13. Classification Accuracy on the Training Dataset After Feature Reduction.](image4)
Then the model was tested with the additional malware capture obtained from CTU. After testing the model with this dataset, the results were as follows: 99.79% for accuracy, f1-score of 99.89%, perfect precision score, and 99.79% for recall. The entire dataset was a malware attack, so it can be observed that very few records were misclassified. In the confusion matrix in Fig. 14, the classification report has been presented for this set. In the same confusion matrix, it can also see 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, the model was tested against a dataset generated during the experiment mentioned previously. The duration of the attacks launched was not very long and did not generate as many records as desired. Therefore, this network attack capture has been combined with normal traffic and balanced it to be composed of 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 Fig. 15.

![Fig. 13. Confusion Matrix for 1st Test Set.](image1)

![Fig. 14. Confusion Matrix for 2nd Test Set.](image2)

**V. Conclusion**

A NIDS has been implemented to detect generic network attacks and Wi-Fi specific attacks. The most useful features selection techniques have been applied to select the features from the datasets to achieve precise and efficient performances. Ensemble ML models have been applied to classify network traffic as either normal traffic or malicious network traffic. The Ensemble ML models have been implemented separately to analyse Wi-Fi specific attacks and generic network attacks. RF for Link Layer Attacks and Bagging for Network Layer attacks were the best performing models. It is worth highlighting that the models performed well accuracy, performance measure, F1 score, precision, and recall. As mentioned, these additional metrics gave 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%. A perfect score has been obtained for Network Layer attacks for two test sets. 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. This work demonstrated that careful attribute selection improves efficiency and speeds up the training and classification process. Attribute selection also helps the model consider useful information that directly affects how records are classified. Thus, ignoring as much noise as possible, the model performs more precisely. The objective of this work is not just to operate a wireless network but to emulate an organization network environment where Ethernet connections are considered an important component. It is important to mention that this work planned to analyze generic network attacks for Ethernet connections. The packets must collect for two specific reasons; first, to gather information for the NIDS implementation. Second, for the existing system to capture the packets in real-time.

The proposed work has many branches through which it can be expanded and improved in future. The following are a few considerations for upgrading this work to enhance its functionality:

- One of the most significant challenges faced in this study was generating a larger original dataset for training. Even though it was possible to set up the desired infrastructure and launch several network attacks, the model was not able to collect massive network captures of a wider variety of attacks. Therefore, the model considers obtaining more penetration tools to launch a wider variety of network attacks as future work. These tools can help generate a more trustworthy dataset.
- Another consideration is to provide this ensemble-based multi-level intrusion detection service from the cloud. It has become popular in delivering cloud-based services that can be considered this feature as one of the key upgrades. This can be viewed as a key upgrade because computational resources are expensive, and for data processing tasks, hardware
requirements are high. By providing this cloud-based service, it can become more accessible for anyone without the hardware resources needed for efficiently processing network data.

- An additional tool that should be considered for future work is using Big Data tools for implementation. Network data is considerably large and could be viewed as the perfect scenario for using 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 offer Machine-Learning libraries that can be used to implement ensemble models.

**References**

[1] Simon Kemp, “Digital 2022: Global Overview Report,” Jan 2022. [Online]. Available: https://datareportal.com/reports/a-decade-in-digital?utm_source=Global_Digital_Reports&utm_medium=Partner_Article&utm_campaign=Digital_2022

[2] ©2021 Microchip Technology, Inc. “Developer help.” [Online]. Available: http://microchipdeveloper.com/tcpip/tcp-ip-five-layer-model

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

[4] 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.

[5] Abhijit Das, Pramod and Sunitha B S, “Anomaly-based Network Intrusion Detection using Ensemble Machine Learning Approach” International Journal of Advanced Computer Science and Applications(IJACSA), 13(2), 2022. http://dx.doi.org/10.14569/IJACSA.2022.0130275

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

[7] 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-f10baf638234

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

[9] 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/10.1007/978-1-4899-7829-3-5

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] B.A. Tama, K.-H. Rhee, “An in-depth experimental study of anomaly detection using gradient boosted machine, Neural Comput. Appl. 31 (4) (2019) 955–965.

[18] S. Rezvy, Y. Luo, M. Petridis, A. Lasebue, and T. Zebin. An efficient deep learning model for intrusion classification and prediction in 5g and iot networks. In 2019 53rd Annual Conference on Information Sciences and Systems (CISS), pages 1–6, March 2019

[19] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis. Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset. IEEE Communications Surveys Tutorials, 18(1):184–208, Firstquarter 2016.

[20] F. K. Örs, M. Aydn, A. Boyittarkan and A. Levi, “Scalable WiFi Intrusion Detection for IoT Systems,” 2021 11th IFIP International Conference on New Technologies, Mobility and Security (NTMS), 2021, pp. 1-6, doi: 10.1109/NTMS49979.2021.9432662.

[21] I. Syarif, E. Zaluska, A. Prugel-Bennett, G. Wills, “Application of bagging, boosting and stacking to intrusion detection, in: Machine Learning and Data Mining in Pattern Recognition, Springer, 2012, pp. 593–602.

[22] Breiman, L. (1996). Bagging predictors. Machine learning, 24(2), 123-140.

[23] Dietterich, T. G. (2002). Ensemble learning. The handbook of brain theory and neural networks, 405-408.

[24] Dietterich, T. G. (2000b). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. Machine learning, 40(2), 139-157.

[25] Q. Niyaz, W. Sun, and A. Y. Javaid, “A deep learning model for intrusion classification and prediction in 5g and iot networks. In 2020 9th International Conference on Software Defined Networking (SDN),” CoRR, vol. abs/2004.07257, 2020. [Online]. Available: http://arxiv.org/abs/1611.07400

[26] Abhijit Das, Pramod and Sunitha B S, “An Efficient Feature Selection Approach for Intrusion Detection System using Decision Tree” International Journal of Advanced Computer Science and Applications(IJACSA), 13(2), 2022. http://dx.doi.org/10.14569/IJACSA.2022.0130276

[27] Abhijit Das, Pramod and Sunitha B S, “Anomaly-based Network Intrusion Detection using Ensemble Machine Learning Approach” International Journal of Advanced Computer Science and Applications(IJACSA), 13(2), 2022. http://dx.doi.org/10.14569/IJACSA.2022.0130275