SAKARYA UNIVERSITY
INSTITUTE OF SCIENCE AND TECHNOLOGY

ANOMALY BASED DETECTION OF DDoS ATTACK USING DISCRETE TRANSFORM AND MACHINE LEARNING TECHNIQUES

M.Sc. THESIS

Mohammed S. M. SALIM

Department	**:**	**COMPUTER AND INFORMATION ENGINEERING**
Field of Science	**:**	**COMPUTER ENGINEERING**
Supervisor	**:**	**Assist. Prof. Dr. Seçkin ARI**

Oct. 2018

SAKARYA UNIVERSITY
INSTITUTE OF SCIENCE AND TECHNOLOGY

ANOMALY BASED DETECTION OF DDoS ATTACK USING DISCRETE TRANSFORM AND MACHINE LEARNING TECHNIQUES

M.Sc. THESIS

Mohammed S. M. SALIM

Department	:	**COMPUTER AND INFORMATION ENGINEERING**
Field of Science	:	**COMPUTER ENGINEERING**
Supervisor	:	**Assist. Prof. Dr. Seçkin ARI**

This thesis has been accepted unanimously / with majority of votes by the examination committee on 16.01.2013

.....	.....	.....
Head of Jury	**Jury Member**	**Jury Member**

DECLERATION

I declare that all the data in this thesis was obtained by myself in academic rules, all visual and written information and results were presented in accordance with academic and ethical rules, there is no distortion in the presented data, in case of utilizing other people's works they were refereed properly to scientific norms, the data presented in this thesis has not been used in any other thesis in this university or in any other university.

Mohammed SALIM

25.10.2018

PREFACE

This master thesis focuses on a behavioral approach for building an anomaly-based network intrusion detection system (NIDS). The behavioral-based processing technique is one of the state-of-the-art methods used for developing NIDS. In this technique, a set of behavioral features are extracted and analyzed to discriminate the network traffic. The potential in this approach is the subtle behavioral dissimilarity that can be found by analyzing the specified behavioral features for both network profiles. Moreover, the behavioral approach is particularly useful when an attacker's traffic cannot be discriminated using the signature-based detection techniques.

I would first thank Allah for his bounty and his many Blessings that let me accomplish this work. He has praise first and last.

I would also like to express my sincere gratitude to all those who helped me to accomplish this work, led by my advisor Dr. Seçkin ARI, who has spared no effort to help me during the preparation of this work. Without his appreciated participation and wide knowledge in many disciplines in computer engineering, this work could not have been completed. Also, my gratitude is expressed to the Turkish Government and in particular to YTB for giving me the opportunity to complete my graduate study. I would not be able to reach this academic achievement without their generous support during preparation for this degree.

Finally, I have to express my profound gratitude to my parents and to my spouse for their passionate support. Their encouragements were always appreciated in difficult times. without whom I would never have enjoyed so many opportunities.

TABLE OF CONTENTS

LIST OF SYMBOLS AND ABBREVIATIONS

ANN	: Artificial Neural Networks
CI	: Computational Intelligence
CRS	: Core Rule Set
DT	: Decision Tree
DWT	: Discrete Wavelet Transform
HIDS	: Host-based Intrusion Detection System
HTTP	: HyperText Transfer Protocol
ICMP	: Internet Control Message Protocol
IDS	: Intrusion Detection System
IP	: Internet Protocol
IPS	: Intrusion prevention System
ML	: Machine Learning
NIDS	: Network intrusion detection system
NVD	: National Vulnerability Database
OWASP	: Open Web Application Security Project
SMTP	: Simple Mail Transfer Protocol
SOMs	: Self-organizing Maps
SVM	: Support Vector Machine
TCP	: Transmission Control Protocol
UDP	: User Datagram Protocol
VPN	: Virtual Private Network
WAF	: Web application firewall
XSS	: Cross-site Scripting

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Keywords: Anomaly based detection, DDoS attack, HTTP GET Request, behavioural features, Request Rate, URI Diversity, Machine Learning, SVM

Distributed Denial of Service (DDoS) attacks is a serious threat to any online service on the internet. In contrast to other traditional threats, DDoS HTTP GET flood attack can exploit legitimate HTTP request mechanism to effectively deny any online service by flooding the victim with an overwhelming amount of unused network traffic. This paper introduces a new anomaly-based technique for discriminating between DDoS HTTP GET requests and legitimate requests using a combination of behavioural features. The main selected features are the diversity of the requested objects, requesting rates for all the requested objects, and request rate for the requested object with the most frequency. These parameters are selected as the proposed features that will be used together for effective discrimination within the proposed system.

The proposed mechanism is evaluated using sub set of the UNB ISCX IDS 2012 evaluation dataset representing attack traffic, in addition to another sub set extracted from the 98 world cup dataset for legitimate traffic. Performance evaluation shows that the proposed mechanism does effective detection due to the subtle behavioural dissimilarity between non-recursive attack and legitimate requests traffic.

There are many behavioural parameters that can be extracted from the network traces that could be helpful for discriminating or detecting other types of attacks in an effective way. There are many fields within the well-known HTTP protocol which can be studied and analysed to extract new detection or discrimination parameters that can be used to detect more advance and subtle attacks. Then, these extracted parameters could be used to implement a new detection mechanisms or systems.

AYRIK DÖNÜŞÜM VE MAKINE ÖĞRENME TEKNIKLERI KULLANILARAK DDoS SALDIRISININ ANOMALI TESPIT EDILMESI

ÖZET

Anahtar kelimeler: Anomali tabanlı tespiti, DDoS saldırısı, HTTP GET İsteği, davranışsal özellikler, İstek Hızı, URI Çeşitliliği, Makine Öğrenmesi, SVM algoritması

Dağıtılmış Hizmet Reddi (DDoS) saldırıları, internetteki herhangi bir çevrimiçi hizmet için ciddi bir tehdittir. Diğer geleneksel tehditlerin aksine, DDoS HTTP GET sel saldırısı, kurbanını kullanılmayan ağ trafiğiyle sel tarafından herhangi bir çevrimiçi hizmetin etkin bir şekilde inkar edilmesi için meşru HTTP istek mekanizmasını kullanabilir. Bu yazıda DDoS HTTP GET isteklerini ve yasal isteklerin davranışsal özelliklerin bir kombinasyonunu kullanarak ayırt edilmesine yönelik yeni bir anomaliye dayanan teknik tanıtılmaktadır. Seçilen ana özellikler, istenen nesnelerin çeşitliliği, istenen tüm nesneler için oranlar talep edilmesi ve istenilen nesne için en yüksek frekansla talep oranıdır. Bu parametreler önerilen sistemde etkin ayrımcılık için birlikte kullanılacak önerilen özellikler olarak seçilmiştir.

Önerilen mekanizma, meşru trafik için 98 dünya kupası veri kümesinden çıkarılan başka bir alt gruba ek olarak, saldırı trafiğini temsil eden UNB ISCX IDS 2012 değerlendirme veri kümesinin alt kümesi kullanılarak değerlendirilir. Performans değerlendirmesi, önerilen mekanizmanın, özyinelemeyen saldırı ve meşru istek trafiği arasındaki ince davranışsal farkdan dolayı etkili bir algılama yaptığını göstermektedir.

Ağ izlerinden çıkarılan ve diğer saldırı tiplerini etkili bir şekilde ayırt etmek veya saptamak için yararlı olabilecek birçok davranış parametresi vardır. Bilinen HTTP protokolü içinde, yeni tespit veya ayrımcılık parametreleri çıkarmak için incelenebilecek ve analiz edilebilecek birçok alan vardır. Ardından, bu ayıklanan parametreler yeni bir tespit mekanizmaları veya sistemleri uygulamak için kullanılabilir.

CHAPTER 1. INTRODUCTION

1.1. Background

The Internet network becomes more popular as a huge number of clients use the internet connection to access global services, knowledge, and information. Online operations like shopping, trading, banking and payment make services friendly and easier for customers, however, these online services are more vulnerable to malicious attacks. Securing these services against such threats and intruders becomes harder for network administrators and even for security experts.

Many websites provide detailed information and instructions about security software and hacking tools. Intruders, and even new users, can exploit these details to initiate an attack against other clients or against online eservices. Therefore, internet users, websites, and information infrastructures are significantly vulnerable to deliberate attacks preventing users from accessing their private or confidential information. For instance, to send a site offline, unused traffic can be directed to overload that website. Anyone and anywhere in the world can use bots maliciously to launch such attack which is known as Distributed Denial of Service or DDOS attack.

To initiate the DDoS attack, an attacker has to build a bot. A bot is constructed by controlling a set of infected computer machines. Attackers take control of the computer machines by broadcasting malicious software over emails, websites and social media. The attacker can then command the bot to direct the volumetric traffic to overwhelm the destination service provider. The target will not be able to handle that huge amount of traffic and as a consequence it’s service will be down. Moreover, attackers have not to construct their own bots by themselves as bots can also be sold or rent to attackers intending to initiate a targeted attack. Anyone anywhere in this world can initiate a

DDoS attack in a cheap and easy way that can take almost any unsecured site down no matter its size. Unsecured websites are vulnerable to DDoS attacks as they do not have enough resources or security infrastructure to defend themselves against such attack. Intruders take advantage of this vulnerability by using the DDoS attack to exploit internet users and influence their political or financial or any other events.

Attackers can easily cause a huge damage to any online service or site without being susceptible to any responsibility. To recompense for security vulnerabilities in the whole network infrastructure, security tools and software such as intrusion detection system, anti-virus and firewalls have been developed by security experts. The majority of service providers and normal internet clients rely on these tools to secure their internet connections and deny any anomalous traffic. Legitimate users are protected by these tools from illegitimate or anomalous users who attempt to exploit all possible vulnerabilities in the whole information infrastructure to steal confidential information or personal data hosted online or even locally. Unfortunately, these anomalous users or hackers can take advantages over network administrators and security experts. One of these advantages is that the attackers can totally mimic the legitimate profile. In this condition, passive network tools (e.g. routers and firewalls) are not aware enough to handle such subtle attacks. Therefore, Network Intrusion Detection Systems or (NIDS) were introduced to deal with such situations using different subtle techniques.

Intrusion detection systems (IDSs) are classified as security tools that have the ability to monitor and analyze actions and traffic initiated by all users in any site or system to discriminate any anomalous profile. Throughout this work, an anomaly-based IDS is proposed and developed using the anomalous methodology to deny any piece of network traffic related to the predetermined network attack.

The anomaly-based detection method attempts to predict any malicious behavior (e.g. abnormal network traffic or strange network session) that differ to some extent from the expected Profile (1). In anomaly-based intrusion detection, malicious patterns are also called anomalous patterns which refer to any pattern deviated from the normal pattern in somehow. Deviation from the expected profile is the main factor in the

anomaly-based detection mechanism (2). The training and monitoring phases are the most common phases in any anomaly-based detection system. The normal profile representing the expected behavior during the runtime is built and performed during the training phase. In monitoring phase, all activities and transmitted network traffic are analyzed and processed (e.g. by using set of online extracted key features) to validate the inspected behavior or data against the expected profile. Various analysis techniques can be utilized to implement the detection engine like machine learning and statistical analysis techniques. Based on this analysis the current behavior (i.e. session or traffic) can be classified as a normal or anomalous profile (3). Anomaly-based intrusion detection technique is a popular method that is broadly applied in many fields including fraud detection and intrusion detection (4).

In recent years, the research community in intrusion detection field focuses on the anomaly-based approach. A comprehensive survey on anomaly-based network intrusion detection is presented in (4). The common techniques, tools and datasets used for developing anomaly-based detection system are discussed and classified in this survey. Also, challenges and recommendations are concluded. Another extensive survey on anomaly-based network intrusion detection is provided in (5). The survey focuses on the classification of the IP traffic using machine learning classifiers.

The robustness of this approach is coming from dealing with behavior rather than signatures in anomalous detection. Combining the behavioral features extracted from the raw data with artificial intelligence techniques (e.g. SVM and decision tree algorithms) is a new trend in intrusion detection research to build an effective and robust IDSs that can perfectly predict and discriminate the anomalous network traffic.

In this work, three features are selected for building anomaly-based network IDS that assign each network traffic sample to a relevant profile. In the monitoring stage, the proposed features are fed to the classifier within each sampling period. The proposed system's output will be delivered by the end of each 60 consecutive sampling periods. DWT (Discrete wavelet transform) technique is used to assign the previous 60 consecutive sampling periods to only one profile rather than 60 profiles. Summarizing

the predicted output can increase the robustness and the efficiency of the proposed IDS by reducing the analysis process for the output by administrators as well as the security experts.

CHAPTER 2. SECURITY

2.1. Intrusion Detection System

Malicious activities targeted at computing resources or network environment can be detected by dynamically monitoring and analyzing all the transmitted traffic in addition to all the generated activities using pieces of software or other tools that called Intrusion detection systems (IDSs) (6) (7). Moreover, IDSs are defined by The National Institute of Standards and Technology as "the process of monitoring the events occurring in a computer system or network and analyzing them for signs of intrusions, defined as attempts to compromise the confidentiality, integrity, availability, or to bypass the security mechanisms of a computer or network" (8). IDSs detect unauthorized or anomalous users attempting to generate anomalous activities or traffic (e.g. accessing private information in computer systems without the right privilege, transmitting volumetric unwanted traffic) using the predefined normal profiles that compared against all new activities and traffic. The predefined profiles are dynamically built and updated for each profile type from the audited access logs.

IDS is an integral part of any comprehensive information security architecture within any computer environment. IDSs perform as the logical complement to passive network firewall technology deployed at the network border (9). Therefore, IDSs play key roles for securing and controlling any site or information infrastructure. IDSs are necessary tools in any security infrastructure. These tools or devices can be used to analyze huge amount data (e.g. generated activities, network traffic traversing network) every second for possible security breaches whether these breaches were originated from inside or outside the organization (10).

Attackers attempt to exploit any vulnerability or defect within any part of the whole infrastructure, e.g. operating systems, network protocols, software applications, etc.,

to launch their unauthorized activities. Vulnerability is defined by the National Vulnerability Database (NVD) as “A weakness in the computational logic (e.g., code) found in software and hardware components that, when exploited, results in a negative impact to confidentiality, integrity, or availability“ (11). Growth of number of vulnerabilities found in software over time is shown in Figure 2.1. this number has doubled in this decade, but during 2017 the number has increased horribly.

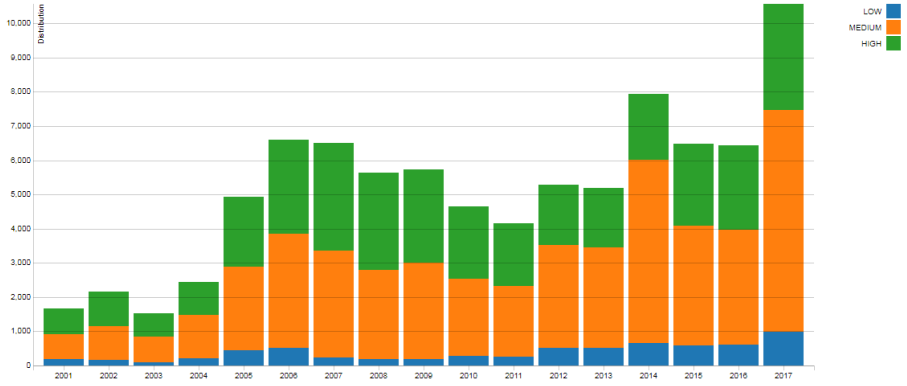

Figure 2.1. The Growth of number of vulnerabilities in software over time (12)

These vulnerabilities are real threats because they allow other parties to easily access personal and sensitive information e.g. banking information, passwords even when hosted by well-known IT companies or banks. Furthermore, internet, social media, cloud computing and mobile devices constitute the new challenges of security experts and researchers. With these real threats and challenges and despite the intensive research conducted on the field of intrusion detection, more efforts are still needed in this area to achieve necessary improvements in various aspects including performance and accuracy of the detection (4). In this work, IDSs are studied as a new subtle layer of security within any modern security infrastructures. IDSs improve the performance of a security infrastructure by adding a new subtle security layer on top of the existing.

2.1.1. IDS’s evaluation criteria

IDSs evaluation process is a crucial on enhancing the information security. By evaluating the way an IDSs monitor, analyze traffic, and detect intrusion, researchers and developers in this field can improve IDSs. Also, the evaluation results and

conclusions enable developers and administrators to discover IDS's capabilities and limitations (13). For developing a robust and efficient IDS, a set of standard measurements and metrics should be achieved (14) (7). In this section, these evaluation standards and factors used to evaluate IDSs are discussed.

Accuracy: The accuracy property measures the correctness of the detection achieved by a particular IDS. It shows the percentage of true detection for normal and anomalous profiles (14) (15). The accuracy measurement is used to ensures that IDSs can classify the actual behaviors correctly.

Performance: IDS's performance is a key measurement in the evaluation process of an IDS. Performance factor measure the ability of an IDS to process traffic online in a high-speed link (e.g. 10Gbps) with minimum packet loss. The performance property depends on other external factors (e.g. hardware platform, operating system) (14).

Efficient IDS should consume less time and resources while detecting the anomalous traffic. Consuming more time or resources by the IDS services may badly affect other end user services e.g. web pages forwarding, e-mails, banking, etc. that will suffer from a reduction in the quality of service. For instance, in 10G bps Ethernet network, an IDS should be able to process between 812,740 and 14,880,960 number of received and/or transferred packets per second (16). Moreover, there are other popular and common performance measurement options particularly for evaluating IDSs that operate in the application layers. For example, connection (TCP connection) per second (c/s) and maximum concurrent connection per second (mcc) are common metric for evaluating the performance of the IDSs performance. For instance, the Cisco ACE supports 4 million concurrent connections with connection rate of 325,000 c/s (i.e. 325,000 new connections can be created each second and up to four million concurrent connections which can be satisfied during 12 seconds) (16). If the IDS require unrestricted time or resources for inspecting and analyzing this huge amount of connections every second, this means that the infrastructure or the site will suffer from bad quality of service or a security breach. For instance, in the worst case the IDS will inspect and analyze up to four million connections each second for anomalous

connections. This means that the IDS should handle each connection in less than 250 nanosecond seconds, otherwise the traffic will suffer from the delay during this stage which may lead to other breaches. Inspecting concurrent connections (sessions) individually is presented in the literature of intrusion detection. In this work, the proposed mechanism depends on extracting behavioral features from the network traffic rather than inspecting the concurrent connections/packets each second. The extraction should be completed in a significant time to ensure real-time operations.

Completeness: The completeness factor represents the range of the detectable attacks or threats that can be handled by an IDS. Achieving this measurement is not practical and very expensive because having a thorough knowledge about universal attacks and threats or anomalous use is impossible. However, the completeness measurement for an IDS can be evaluated against known attacks. For instance, network intrusion detection, XSS (cross site script) detection. A fully aware IDS should be able to handle all known vulnerabilities and threats. Moreover, such IDS need to be able to handle new unseen attacks by employing novel and subtle detection (14).

Fault Tolerance: This property shows the ability of an IDS to provide the detection service under any attacking circumstances. An IDS should be able to withstand and continue operating no matter what the type of the current attacks. Also, IDSs under attack need to ensure ongoing network and application services (14). For instance, attackers can facilitate more advance malicious activities by targeting the IDS first and sending it down. Targeting an IDS component by attacks could make the entire network intrusion-detection system ineffective in addition to make the entire infrastructure susceptible to countless number of security breach (17). However, attacks against IDSs is not considered in this work.

2.1.2. Binary classification

In intrusion classification, there are two main profiles. An anomalous behavior is classified to the POSITIVE category, while the NEGATIVE is labeled as a legitimate. Moreover, an instance can be classified by an IDS to a predicted category that can be

either correct (TRUE) or incorrect (FALSE). Therefore, there are four possibilities for the result of a binary classifier defined as follows: True Positive, True Negative, False Positive, and False Negative. As listed in table 2.1, True Positive indicates that an anomalous behavior is correctly predicted by an IDS, whereas a False Positive indicates that an IDS predict a legitimate behavior as being an anomalous. Likewise, a True Negative occurs whenever a normal behavior is correctly predicted by an IDS as legitimate, while a False Negative occurs when an anomalous behavior is incorrectly predicted as a normal behavior. (i.e. False Negative indicates the worst case when an IDS fails to detect the attack) (14).

Table 2.1. Intrusion detection's binary classification

	Actual label	Predicted label
True Positive (TP)	Intrusion	Intrusion
False Positive (FP)	Legitimate	Intrusion
True Negative (TN)	Legitimate	Legitimate
False Negative (FN)	Intrusion	Legitimate

There are no other possibilities for the result of a binary classifier other that the four combinations (i.e. TP, FP, TN, FN). Thus, theses four variables are used as the key factors to indicate the accuracy of the IDSs and other evaluations metrics. Consequently, a real time and efficient IDSs are expected to make true decisions or predictions for the majority of the inspected instances (i.e. predicting TP and TN decisions as many as possible with no FP and FN predictions as possible (14).

2.1.2.1. Accuracy

Accuracy is a property that shows to what extend does a system or a method works correctly. Accuracy is measuring the rate of the correctly predicted instances in addition to the prediction failure rate that a system is producing (14) (18). For example, An IDS with accuracy of 95% correctly assigns 95 instances or observations out of 100 to their actual classes while failing to assign the remaining observations to their actual classes.

$$accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (2.1)$$

However, accuracy measurement alone makes the evaluation prune to error and deceptive (19). For instance, with an unbalanced dataset that contains 900 negative observations and 100 positive observations, when IDSs misclassify all the positive instances, the accuracy will be equal to 90% while the model completely fails to classify all the positive instances to their actual class. Therefore, it is recommended to measure the classifier accuracy while excluding the correctly classified negative instances. The following three factors have accomplished particular distinction in evaluating the accuracy and efficiency of the IDSs. The introduced factors called precision, recall, and F measure (18).

2.1.2.2. Precision, recall, and f-measure

In measuring accuracy and efficiency using precision, recall, and F-Measure metrics, the key factors are the indicators for the number of correct and incorrect anomalous predictions (i.e. TP, FP and FN). TN indicator is not important in this evaluation criterion, so it will be excluding while modeling this specific type of accuracy.

Precision: precision indicates the percent of the number of correctly predicted positive instances (i.e. anomalous instances) out of all the predictions. Efficient and practical IDSs should achieve a high precision rate which ensure minimizing the false positive rate also called false alarm. False positive cases do not represent a security breach but they produce an administration overload that will (18).

$$precision = \frac{TP}{TP+FP} \; where \; precision \; \epsilon \; [0, 1] \quad (2.2)$$

Recall: recall is the percent of the number of correctly predicted positive instances (i.e. anomalous instances) out of all the actual positive cases. Efficient and practical IDSs should achieve a very high recall rate which ensure minimizing the false negative rate. Moreover, any false negative case should be eliminated completely in IDSs. In false

negative case, an anomalous instance is handled like a normal behavior, Therefor, false negative represents a security breach that should be prevented (14).

$$recall = \frac{TP}{TP + FN} \quad where\ recal\ \epsilon\ [0,1] \tag{2.3}$$

F-Measure: both of precision and recall accuracy measurements monitor the accuracy from single different view. Therefore, both of these measurements cannot thoroughly model the accuracy of an IDS alone. So. A new accuracy measurement is introduced by mixing the properties of the precision and recall measurements to model more appropriate single evaluation metric called F-Measure.

$$\mathrm{F-Measure} = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} \quad where\ \mathrm{F-Measure}\ \epsilon\ [0,1] \tag{2.4}$$

Rather than using two evaluation metrics for monitoring the accuracy of the classifier, F-Measure can be used as single evaluation criterion for the accuracy. Subtle and efficient IDSs should accomplish high rate for the F-Measure ensuring low rate for the false alarm while minimizing the possibility for any failure of attack detection. Thus, IDS's F-Measure property is highly recommended to be very high as much as possible.

2.1.3. Detection methods

An IDS's engine or classifier is constructed mainly using one of the popular detection methods. In object detection, there are two common approaches for recognizing an object. The first approach is known as anomaly-based method. In this approach the detection engine is built to be familiar with the "normal" behavior only and any deviation (deviation to some threshold) from that behavior is considered as an anomalous or attack. In the other approach called misuse-based, an IDS's engine is provided with the required knowledge representing the abnormal or the attack behavior (6). During this work, only topics related to learning-based methods are discussed. The resources in (8), (20) and (6) are valuable materials discussing more details related to all techniques used for developing IDSs.

2.1.3.1. Anomaly-based IDSs

Anomaly-based IDS's engine depends on normal behavior (i.e., normal user activities, normal network traffic, etc.). Normal profile or behavior can be modeled using audited network traces and logs that constructed only from normal and legitimate traffic and activities. Then, the detection engine utilizes this normality model for detecting any anomalous behavior that deviates from this model with a specified threshold (6) (20). Although anomaly-based detection engines can detect unseen anomalous behaviors effectively when compared to other types of engines, they are prone to high rate false positive which decrease their accuracy. (20) However, in practice, it is impossible to define a general normal profile for this type of IDSs. Therefore, anomaly-based IDSs are prone to high rate false alarms due to considering any relatively deviated unseen normal profile as an anomalous profile mistakenly. In other words, any deviated normal profile or behavior which exceed the deviation threshold could be labeled incorrectly as anomalous. It is hard to cover all possible normal behaviors or profiles for any dedicated environment. Clearly, collecting such a dataset is infeasible.

Many anomaly-based IDSs have been proposed in the literature (20). For instance, in (21), a classification-based anomaly IDS is proposed. SVM-based classifier is used to model the anomaly-based detection engine for new unseen anomalies. The core normality model is built using only normal labeled instances. A novel kernel function is developed by the authors that construct a new higher dimension data space based on the classified instances to improve classification results. The developed kernel function considers properties of Netflow data and enables determination of similarity between two windows of IP flow records. Their proposed IDS achieve and average accuracy of 92% with the all attack samples. In this work, like (21), a classification-based anomaly IDS is proposed for detecting anomalous traffic generated by a specific type of DDoS attack. Binary SVM and decision tree learning algorithms are used to model the normal traffic based on a set of selected features during the training phase.

Also, clustering-based anomaly IDS is introduced in the literature. Unlike classification-based anomaly detection engine, clustering-based anomaly detection engine detects anomalous behavior without relying on any base or core profile whether

it was normal nor anomalous. Thus, there is no required knowledge for building that core model in clustering base detection engine. For instance, MINDS (Minnesota Intrusion Detection System) (22) is a data mining-based detection engine for detecting network anomalous behavior where each network trace is assigned a score as indicator to the severity degree of this network trace. First, the required features are extracted by analyzing and filtering the network traffic. Then, time window technique is used to summarized the extracted features. Next, known attacks are removed and excluded. Finally, the clustering-based anomaly detection engine labels each network instance using a score indicating whether it is an anomalous or normal instance.

In (2), PAYL is introduced as another payload-based anomaly IDS. The mean and variance of the byte values distribution is calculated to build a learning model during training phase. During detection, payload distribution is compared for each incoming payload for anomalous detection. PHAD and NETAD are also payload anomaly-based IDSs. They can detect instances deviated from the core model that is a learning-based model for normal network traffic. PHAD (23) (Packet Header Anomaly Detection) monitors 33 attributes extracted from the Ethernet packet header fields, IP, TCP, UDP and ICMP packets. These fields are used to build a learning-based engine for anomalous packet detection. The same as PHAD, ALAD (23) models the incoming TCP request by assigning a labeling score to each TCP connection instance. ALAD examines and model only the first 1000 bytes. NETAD (24) (Network Traffic Anomaly Detector) is similar to PHAD; the detection engine is a classification-based engine depending on feature extraction from the raw data. Another proposed classification-based anomaly IDS is ADAM (25) (Automated Data Analysis and Mining). ADAM utilize a set of classification methods and association rule mining techniques to detect attacks.

Machine learning techniques as a detection method can be used to develop anomaly-based IDS. In learning based, the IDS's model or engine is built by training an appropriate algorithm using a dataset containing both of legitimate and anomalous behavior information. During the real-time detection, the developed model detects or

labels each arriving profile or activity whether it is positive or negative. In intrusion detection, the normal profile is the negative while the anomalous is the positive (14).

Machine learning techniques can be divided into four categories: learning models, advanced statistical models, rule-based models, biological models, and signal processing techniques-based models. During this work, only learning based approach is considered. The works (20) and (6) discuss more details related to other techniques used for developing anomaly-based approach.

2.1.3.2. Misuse-based IDSs

Misuse-based IDSs rely on a knowledge base containing a set of modelled attacks or attacks signatures. The IDS's detection engine compare any inspected instance against these signatures stored within its knowledge base and an alert is fired when the signature of the inspected instance is identical to one of the known attack signatures. The misuse-based IDSs consider traffic bearing a signature differing from all the know attack signatures as a normal traffic, otherwise when there is a match, the traffic is considered as an attack. One of the main advantages of this type of IDSs is that the false positive rate is very low due to the signature-matching restriction in considering an instance as an attack. The known attacks can be modeled with full details and description using any description language within the signature knowledge base. Therefore, the attack signature match only the destined attack and it becomes hard to match any legitimate traffic. This feature classifies this type of IDSs to be free of the false positive alarms, i.e. misuse-based IDSs are not prone to the false positive.

However, misuse-based approach has many limitations and weaknesses as well. One of the main limitations is broadcasting and updating the knowledge base in misuse-based IDSs. Misuse-based IDSs cannot correctly handle new unseen attacks which is a weakness case for this approach. Therefore, providing an up-to-date knowledge base by continuously updating the internal knowledge with all possible signatures for each new attack release is a critical requirement to provide high accuracy. Otherwise the information system or infrastructure is susceptible to suffer from security breaches (6). Moreover, for known attacks, modeling a signature that covers all releases of the attack

is hard. Any mistakes in the modeling of these signatures will let the IDSs to be prone to false alarm and thus decrease the effectiveness of the detection technique (20). Even for the known attacks, attackers can alter part of the properties of the attack to ensure mismatching the known attack signature stored int the IDS's internal knowledge base. Therefore, the attackers avoid being detected by the misused-based IDS which considered another weakness of this detection approach. These limitations and weakness should be handled carefully to ensure zero security breaches and maximizing the rate of the F-measure accuracy factor.

2.1.4. IDSs types

IDSs can be classified based on the monitored and analyzed activities and events by these systems. Network-based IDSs, host-based IDSs, and application-based IDSs are common groups or classes for classifying the IDSs (6).

2.1.4.1. Host-based IDS

The host-based IDS (HIDS) is characterized by the property that its installed or located on a single hosting computer. The host-based IDSs attempt to detect anomalous traffic or attacks against the hosting computer environment by inspecting all activities (e.g. system calls) and data (e.g. system logs and network traffic) generated in the hosting environment (e.g. operating system). Based on that inspection and analysis, host-based IDSs can classify the currently monitored behavior or activities as anomalous or normal behavior (7).

The host-based IDSs monitor the dynamic behavior and the state of a computer system. Besides such activities, the host-based IDSs monitor accessing resources by the active utilities or software, for example, a software attempts inexplicably to access or alter a restricted system resources like credential database or paying information. Furthermore, the host-based IDS inspect the state of a system, the stored information, whether this information was stored permanently in files like log files, or it was dynamic information like that stored in the system's RAM. The contents of these data

repositories are inspected and analyzed to ensure that the behavior of the generated data and activities are as expected (26).

The main advantage of the host-based IDS is its ability to access (i.e. inspect and analyze) the content of the network packets even if the traffic is carried over secured communication line. The deployment location at the end of the connection line gives this type of IDS that ability to access and monitor the full payload. At the destination host all packets must have valid fields since the security devices at the network borders or interfaces will drop any packet containing any defected fields. Therefore, the remaining part, payload, can be analyzed efficiently to detect any anomalous pattern.

In contrast, the host-based IDSs have drawbacks as well, one of the limitations that host-based IDSs suffer from is its inability to detect a distributed attack on the entire network as the host-based IDSs only has access to content located on its hosting machine. This can limit the host-based IDS ability to detect complex attacks (e.g. DDoS) destined to any information infrastructure or site.

2.1.4.2. Network-based IDS

The network- based IDS (NIDS) analyzes the network traffic exchanged on a network link to detect the anomalous traffic destined to the site. The transmitted network packets can be inspected with various techniques, e.g. applying pattern matching on the header or the payload of a packet. Advanced analysis supports more subtle analysis of the packet content but requires more resources (6).

First advantage of the Network-based IDS, since the Network based IDS monitors network traffic destined for all users and devices inside the network without the deployment of separate IDSs at different hosts or segments in the network. This means that network-based IDSs reduce management overhead. Second, the feasibility to install network based IDSs as there is no dependencies or restrictions related to other systems or infrastructure including switched networks as the network-based IDS can be deployed within the network devices e.g. Cisco ASA (27). The network-based IDSs are compatible with all the operating systems (10) (6).

However, this approach has drawbacks as well. NIDS, Unlike HIDS, is unable to analyze encrypted packet payloads for anomalous traffic detection i.e. encryption of the network connection completely hides the content from the NIDSs. In the other hand, providing the NIDS with the required private kay to decrypt the traffic to be able to analyze it for anomalous detection overloads the network. Also, traffic decryption is required after the analysis process is complete. This decryption-analysis-encryption process overloads the network and may add significant delay to network packets in addition to the security threats represented in exchanging and storing the required private key (6) (10).

Despite these limitations, this thesis focuses on network-based IDS because it is the most appropriate type of IDS when monitoring network traffic. However, handling these limitations and weakness carefully could improve the performance and efficiency of NIDSs particularly for the payload-based NIDSs which depend on inspecting the content of the network packets.

2.1.5. IDS vs firewall

Legacy firewall devices are passive defense systems that can allow or deny traffic based on defined rules. Legacy security devices are not so subtle to analyze the transmitted traffic for anomalies or attack detection. They usually permit or deny packets based on a preconfigured filtering criterion. For example, denying/permitting all network packets related to a specific protocol or service (e.g. ping service, HTTP web service, etc.). Network administrators are responsible to configure and update these manually depending on current network security policy. Therefore, firewall's performance and efficiency rely on the configured rules and filtering criterion. Furthermore, these rules require continuous evaluation and updates by any change or update in any security policy. In contrast, IDSs monitor and analyze each packet or connection traversing the network to ensure normal or expected behavior. Unlike firewalls, IDSs generate an alert when an attack or anomalous traffic is sensed. IDSs

can also label instances as either normal or attack and store results in labeled audit logs for future development or use.

Advanced firewalls mimic the behavior of the IDSs in the detection of unknown attacks by analyzing the traffic rather than depending on the administrative rules. Moreover, The Next Generation Firewall (NGFW) is the state-of-the-art in security hardware appliances. The new generation provide security features e.g. firewall, IDS, etc. in a single package or device. The NGFW can be configured to perform as both a traditional Firewall in addition to an IDS in the same time to benefit from the advantages of the two defense systems. For example, Cisco ASA is a high-performance multifunction hardware security appliance that offer next generation firewall, IPS, and VPN services. The Cisco ASA deliver these features through improved network integration, resiliency, and scalability (27).

Another example, web application firewall (WAF) is an application firewall that can take control over HTTP traffic by Applying a set of filtering criterion or rules on HTTP traffic. Generally, these filtering criterion and rules handle popular attacks such as cross-site scripting (XSS) and SQL injection attacks. WAFs can be deployed in various ways including hardware appliance installation, also customization as software is applicable (28). The OWASP ModSecurity Core Rule Set (CRS) is detection system based on a set of generic attack detection rules which is well-suited to web application firewalls. The CRS provides protection to web applications against many common attack categories, including SQL Injection, Cross Site Scripting, Locale File Inclusion, etc. with a minimum of false alerts (29). The official website of the OWASP ModSecurity Core Rule Set project can be found at (30). In other words, WAFs can be considered as specialized IDSs for web-applications or application layer within the OSI model. WAFs can be deployed in front of web servers to ensure attack-free traffic.

2.1.6. IDS architecture

There is a common IDS's architecture shared by most implementations. The basic components constructing this common architecture are Data collection unit, Analysis

unit, and Storage unit. (i) Data collection unit collects specific parameters from raw data that could help in predicting possible attack. (ii) Analysis unit processes the collected key parameters to label a profile as one of the two classes, i.e. normal or attack. (iii) Storage unit for logging the detection results including the labeled profiles with the profile definitions for future use and development. IDSs can be deployed in both software and hardware form. For instance, IDSs can be installed on web server machines as a software packages to protect them against possible attacks. Several different components are deployed in software and/or hardware form to install and run an IDS (31). A common IDS architecture is illustrated in Figure 2.2.

Data collection unit collecting the required parameters and metrics (e.g. system logs, system calls, network audit, user activities, etc.) from raw data which will be used later as evidence and provides it to the next stage within the overall architecture to decide whether a specific profile or behavior is anomalous or not. The main duty carried out by the data collection module is the preprocessing of raw data where these massive amounts of data can be represented by a set of key parameters or measurements rather than overwhelming the analysis stage with a huge amount of unwanted data. The reduction of the audit or raw data step facilitates the analysis of a particular profile for final decision (31). For example, Packet level traffic capturing is an important module for developing network based IDSs. Wireshark, open source network analyzer, can capture the required packet level traffic and then preprocess the audited traffic before sending the selected parameters to the detection engine in the next stage.

Analysis component processes the provided parameters collected by data collection unit to predict whether the current profile or traffic is anomalous or not. It is the principal unit in an IDS. IDSs are categorized according to the approach used to implement analysis unit. Several analysis and detection approaches are being proposed including statistical analysis, signature matching, and machine learning, methods as described in section 2.1.3. Analysis module helps in automated the prediction and detection of anomalous data while reducing human intervention in real time.

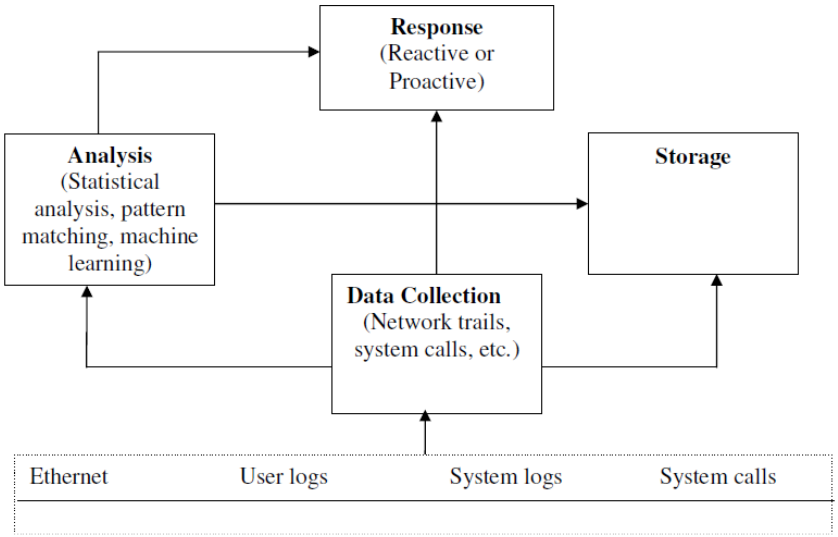

Figure 2.2. Generic IDS's Architecture (31)

Generic IDS's architecture is shown in Figure 2.2. Raw data (e.g. user activities, logs, network traces, etc.) are collected by data collection unit. Then these complex data are fed to the data preprocessing stage to extract only the required information from raw data using different tools. In this work, Wireshark, open source network packet analyzer, is used to collect and extract the required information alongside Cygwin commands simulating Unix based commands under MS windows environment. The extracted parameters or metrics is sent to the analysis module which attempts to predict the class or the status of the current instance profile. The prediction result is sent to the response model which generates an alarm on receiving a positive result.

2.2. DDoS Attack

Distributed Denial of Service is a security breach where net bots are used by the attackers to overwhelm computing resources including network resources and memory resource with volumetric unwanted data to make these resources too busy to handle requests and operations, thus denying legitimate users access to services. DDoS attacks can be initiated by exploiting legitimate services and features. DDoS attacks are initiated against wide range of services and features (e.g. application level services, and network level protocols, etc.). This common attack is classified based on the attacked services which will be unavailable to legitimate users during the attack. (31)

Early DDoS attacks were destined for the low-level protocol particularly against Layers 3 and layer 4. Nowadays, these types of low-level DDoS attack do not constitute any threat to information infrastructure as they can be denied literately by simple security appliances like firewalls or even routers. Moreover, DDoS attacks is continuously evolving by utilizing advance and subtle techniques to mimic the normal behavior. Thus, Attackers can overcome the detection techniques and strike more sensitive services and high-level protocols e.g. DNS and HTTP.

Main motives for initiating DDoS attacks tend to be related either to political or financial motives (32). Moreover, these types of attacks have become available as a commercial service that can be purchased online from anonymous entities or groups. Figure 2.3 is showing an ad for a DDoS commercial service provider. Anyone in this world can request this attacking service from such providers to take down a site as much time as the malicious client like. Malicious clients only have to pay for the requested orders while the DDoS attack providers completely carry out the task (33). Subsequently, online services like DNS, web and email servers should be secured and protected through a powerful defense wall by deploying the modern security appliances and software packages to ensure attack-free traffic and activities.

2.2.1.1. Network DDoS attack

Simple network attack is one of the simplest attack types in DDoS classes that destined to network protocols and services. Layer 4 DDoS attack targets network layer protocols by a huge amount of network traffic. The victim service or device will be overloaded by the unused traffic, so legitimate users cannot anymore access this service. By involving more exploited computer machines or net bots, DDoS attack can be more serious. For instance, The SYN flood and connection flood are main examples for the layer 4 DDoS attacks. These basic attacks cannot saturate the targeted connection of the victim site particularly for the modern network links with high throughput like 10Gbps access lines. Therefore, these types of threats are no more considered in information security research and development. In (32), low-level DDoS attack classes are discussed with details.

Figure 2.3. Any DDOS services 24/7 (33)

2.2.1.2. Application DDoS attack

The application layer of the OSI model is the interface between the end-user software and the underlying layers constructing the internet network. This layer facilitates providing services such as web services, email services. End-user data (e.g., SMTP or HTTP) is transmitted between end-user applications over this layer without any awareness to any other layers. The underlying protocols are abstracted in this layer to simplify creating and developing client or end-user applications. Application Layer or layer 7 DDoS attack is a DDoS-based attack where end-user services are targeted. This application-layer DDoS attack is more subtle and advanced than network attack.

Unlike network DDoS attack, Application layer DDoS attack relies on legitimate requests rather than overwhelming the victim server with bogus requests. Moreover, in this type of attack, the compromised machines used as attacking computers must successfully create a full TCP connection using a genuine IP address. In the attack-free traffic, the attacking activities seem legitimate since it is originated from genuine IP address and behaving normally, but while the attack is going the overall behavior of the traffic is deviated from the regularity. Furthermore, layer 7 DDoS attack does not thoroughly rely on volumetric traffic to successfully run the attack and achieve the malicious results. It is a more sophisticated attack that can exploit vulnerabilities in application layer. (34) Unlike simple DDoS attacks, application layer DDoS attack attempts to exhaust the victim resources like memory and CPU resources rather than overwhelming the bandwidth of the victim link (33).

2.2.2. High level DDoS methods

These type of DDoS attacks are advanced attacks that mimic the normal client's behavior like obeying to network protocols and completing the three-way TCP handshake. Therefore, these attacks seem to be like normal traffic and bypass protection against layer four DDoS attacks. On the other hand, these high-level DDoS attacks apply different methodologies depending on the exploited vulnerabilities or weakness found in application layer's protocols and services to implement the aggression. Limitations found in application layer protocols include properties like connection time out and connection rate which can be exploited to implement a particular high level DDOS attack. However, software companies that provide end-user applications regularly release new versions with updates and fixes that handle all detected vulnerabilities and limitations in the old releases. Despite this attention, attackers intensively investigate everywhere for new possible breaches to initiate a new attack. Some examples of HTTP attacks (34).

2.2.2.1. HTTP GET flood DDoS

This approach utilizes HTTP application protocol to apply denial of service for a target victim. HTTP GET flood attack overwhelm the victim with Volumatic unwanted HTTP requests to exceed the victim throughput by saturating all available resources, thus make their services unavailable during a particular period of time. Simplicity in implementing this type of DDoS attack makes it more common. According to (35), HTTP floods DDoS attack is the most popular which forming more than 80 percent of modern DDoS attacks. Like other DDoS method, HTTP GET flood attack can be initiated by starting a distributed malicious script running remotely from the distributed compromised machines or a prepaid botnet (36). The malicious script utilizes their compromised machine resources and start sending HTTP requests to the victim site. After a period of time and according to the attack intensity, the victim will not be able to respond to any new legitimate request as all its resources are exhausted.

HTTP GET flood attack is one of the serious network attacks because it is totally compliant with the HTTP protocol. HTTP GET flood attack perfectly looks like real

HTTP traffic. Attacker thoroughly mimic legitimate http request to send flood attack. Therefore, signature-based intrusion detection systems may not be able to distinguish this number of anomalous requests from the legitimate requests.

HTTP GET flood attack can by divided into two main classes based on the requested content (37). First HTTP GET flood attacking class called simple HTTP GET flood attack is a basic and widespread application layer attack repeating a static set of URI addresses over and over. In the other hand, Recursive GET flood is a more advanced version of the HTTP GET flood attack that firstly iterate through the website to retrieve, fetch or parse every URI address that can be requested and then start flooding requests using the parsed URI addresses. Unlike simple HTTP GET floods, recursive HTTP GET floods require doing some homework to retrieve all or part of the victim URI addresses. Networks security infrastructures may apply specific polices violating or mitigating URI crawling which make parsing URI addresses more complicated. Also, HTTP GET flooding attack can request random generated URI addresses. In this work, Simple HTTP GET flooding attack will be discussed only due to the limitation found in the available datasets.

In 2010, OWASP provided the public with a free tool OWASP Switchblade (38). This tool provides three various classes of DDoS attacks that can be initiated locally. This tool can be used to make the OWASP Community aware of the DDoS attacks that can exist with Layer7. OWASP Switchblade with default configuration can start an HTTP GET attack. Also, it can be utilized to start a targeted DDoS attack by running and commanding this tool from a distributed mastered machines or bots. Reports at (33) and (34) illustrate extensively a set of tools that can be used to simulate layer 7 DDoS attack. These tools allow the network administrators, security experts and even researches to evaluate and ensure the preferred security level within any site.

2.2.2.2. Low-bandwidth DDoS

This method works by opening connections with the victim and then sending just the required amount of data in an HTTP header that can keep the connections open. After

a period, the destined victim connection space will be filled. Also, low-bandwidth DDoS can be implemented using HTTP POST requests where the request traffic is sent very slowly. They prevent connection from termination.

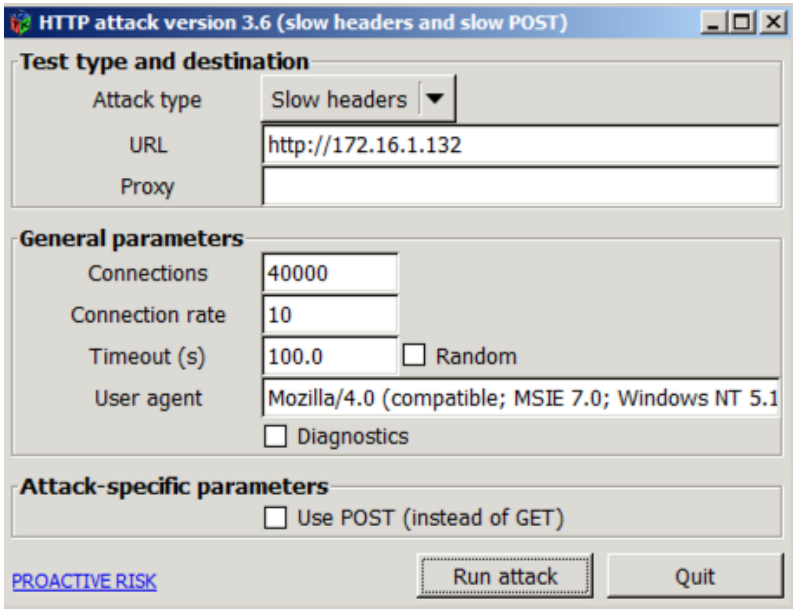

Figure 2.4. A screenshot of switchblader DDoS tool.

There is a main common property that can be found in all DDoS methods. The traffic generated from these methods is deviated from the normal traffic in somehow. Therefore, the IDS can detect these anomalous traffic or attacks based on that deviation. However, there still a need to discover the way that the IDS engine can use to predict that anomalous behavior generated by these attacks. In this work, the traffic behavior will be analyzed by extracting a set of metrics that will be used by the IDS to predict whether the traffic behavior is legitimate or deviated.

CHAPTER 3. ARTIFICIAL INTELLIGENCE BASED TECHNIQUES FOR IDS

In the literature, there are various methods inspired by different disciplines and fields that could be utilized to develop and implement efficient IDSs. For instance, misused/signature and Artificial Intelligence (AI) techniques are popular in this area. They are utilized to build a real-time IDSs. Furthermore, the main two problems with any detection approach are accuracy and performance. High accuracy rate can be achieved by detecting all actual attacks online without failures. In the other hand, enhancing an IDS's performance requires real-time detection without incurring additional delay to ongoing services and application.

AI-based IDSs rely on modeling normal behavior. The normal model is the core model built by using any combination of AI-based techniques (e.g. statistical or Machine Learning (ML) algorithms, etc.) The core model is used to inspect various types of profiles or behaviors (e.g. network traffic, user activities). Detection of unseen aggressions can be more efficient using AI-based approaches as deviation from the core model is considered a threat. Unlike traditional IDSs, AI-based IDSs can detect new attacks on their own. This means that in traditional IDSs the security experts must configure the system for each new unseen attack with the right pattern or signature. In AI-based IDSs, the designed model can learn new unseen patterns without any human intervention. Common learning paradigms are discussed in the next section. However, they may suffer from high false positive as discussed in subsection 2.1.3.1. Intrusion detection process can be automated by utilizing various AI-based techniques. One of the advantages of these techniques is dispensing need for human interaction during real-time detection. Many AI-based methods can be used for developing IDSs. For instance, Machine learning, Fuzzy Logic, Artificial Neural Networks and Data Mining are AI-based techniques that can be used to develop IDSs.

3.1. Learning Methods

In intrusion detection, anomaly detection approach can operate in two common methods according to label attribute. These two common methods are called Supervised and unsupervised learning paradigms. The label attribute indicates the actual class a particular instance is related to. The label attribute is associated with each instance within data space and should assigned one particular class value among a specific set of classes, for instance, in intrusion detection this attribute can be assigned one of two classes called normal or attack. In this case, it is called a binary labeling as there are only two categories and a particular binary learning technique or binary classifier is applied. In the other case, when there are more than two categories (e.g. DDoS, XSS (cross-site scripting), Probing), a multiclass learning technique is applied. Building accurate dataset containing labeled instances of all categories is considered a high-priced challenge. In intrusion detection, creating labeled evaluation data set requires significant efforts and preparations. Researchers and security experts often conduct these efforts and preparations manually to ensure error-free and actual labeling for every instance. Evaluation data sets are substantial for developing IDSs as the core behaviors learned by the detection engine are modeled according to the provided data set. As a consequent any breach or crack within the data set is reflected on the accuracy of the developed IDS (39) (40). However, new behaviors are released dynamically, i.e. new anomalous or even normal types may be developed after building and IDS. Therefore, any IDS should be provided with the new labeled instances that will be used to update the current detection engine and its core profiles.

The instances within a dataset is often represented by a matrix of dimension m by n, also can be called as a variable “D”, where variable m is representing number of observations in a particular dataset, also called instances. Variable n is indicating number of attributes excluding target or class attribute. Each column represents an attribute e.g. in intrusion detection, attributes can include features like source IP address, destination port number, total sent bytes, etc. each attribute can be written as a_j representing the j^{th} attribute in all observations in the dataset “D”. Each tuple represents an observation of the application. The entire dataset represented by the variable “D” can be written as the equation 3.1 below.

In supervised learning and within a particular dataset, each observation should have a specific attribute called “class”. This attribute should be labeled with a particular class name indicating the specific group this observation is belonging to. The set of labels is represented by the variable l_i. In a particular m by n dataset, values of all “class” attributes in all observations are constructing a one-dimensional vector of dimension m, the number of observations in the dataset.

3.1.1. Supervised learning

In supervised learning, the prediction results are defined in the “class” attribute in any particular dataset (5). Supervised learning, also called classification, is a machine learning technique that builds a predictive model that hopefully could be able to correctly predict the value of the “class” attribute for any particular new unseen observation. The predictive model is developed based on the labeled dataset which is constructed from a set of labeled instances. A particular supervised learning-based classifier is modeled by analyzing all these labeled instances. This classifier could be constructed as an algorithm, decision tree, rules, etc., that can be used later to predict value of the “class” attribute for a particular unseen instance (5).

$$D = \begin{pmatrix} a_1^1 & a_2^1 & a_3^1 & \cdots & a_n^1 & [l_1\,] \\ a_1^2 & a_2^2 & a_3^2 & & a_n^2 & [l_2] \\ a_1^3 & a_2^3 & a_3^3 & & a_n^3 & [l_3] \\ . & . & . & \cdots & . & . \\ a_1^i & a_2^i & a_3^i & & a_n^i & ? \\ . & . & . & & . & . \\ a_1^m & a_2^m & a_3^m & & a_n^m & [l_n] \end{pmatrix} \tag{3.1}$$

There are two main stages in any supervised learning system. In the first stage, called training, the provided pre-labeled samples are used to construct the predictive model. Then, in the other stage, this model is used to predict value of the “class” attribute for new unseen instances. The prediction result for these new observations should be evaluated to ensure accurate classification result. Assuming a binary classification

system, each observation in the data (whether seen or unseen) must be labeled as "P" or "N". Moreover, each observation can be defined in term of a set of features as defined in equation 3.1 above. For instance, the ith object in the matrix "D" is defined by the n-attributes or $d_i = (a_1^i, a_2^i, a_3^i, \ldots, a_j^i, \ldots, a_n^i)$. The classification or the supervised learning attempts to find a function like *f(d)* that best predicts the class "l" for any unseen observation defined by the vector d. In binary classification assumption, the predicted values are represented by a discrete set containing only two different values as ("P", "N") where each pre-defined sample belongs to only one class. The generalized function is the key component constructing this type of classifier. The simplest supervised learning is the linear classification where the modeled function is a line that separate between the two classes. Section 3.3.1 describe the linear SVM supervised learning algorithm. Furthermore, more advanced classifiers generalize more complex classification boundary as shown in figure 3.1. Various supervised learning-based algorithms have been utilized for developing anomaly-base detection engines. These algorithms include support vector machine, decision tree, and k-nearest neighbor algorithms.

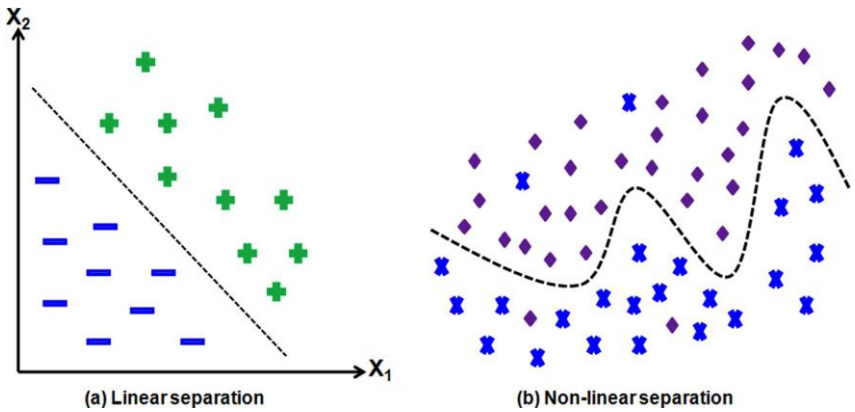

Figure 3.1. Simple and advanced classifiers (4)

One of the principal drawbacks of supervised learning is the need of the pre-labeled instances. This is due to the expensive cost of generating accurate pre-labeled samples in terms of time and budget. Moreover, new pre-labeled dataset may be needed for each new operating environment. IDS research community suffer from the lack of such evaluation datasets. In this work, supervised learning approach is investigated and studied to construct the prediction engine. The objects within the dataset is defined by

three features and pre-labeled to the actual classes during the first stage. Both of SVM and DT algorithms are used during the development of the proposed IDS.

3.1.2. Unsupervised learning

Unsupervised learning algorithms do not need pre-labeled data. Clustering is one of the popular unsupervised learning technique. In this technique, a set of instances are divided to a predefined set of groups known as clusters where all instances belong to the same cluster share common behavior that differ from all behaviors of those in other clusters. (4) In contrast to classification, there is no need for labeled dataset for clustering. Heuristic methods are used to build clusters instead. For instance, instances sharing common features like Euclidean space distance measurement are grouped to the same cluster (5).

K-Means clustering algorithm is a popular clustering technique due to its simplicity and efficiency. K-Means attempts to divide a set of observations into k distinct clusters based on heuristic features such that the variation within each particular cluster is reduced as possible. Typically, the Euclidean Distance measurement is used as a heuristic metric for finding the most appropriate cluster for a particular instance. Variable k as the number of output clusters should be determined first manually before starting this algorithm. As a consequence, defining the number of clusters by human experts is considered as a disadvantage of this unsupervised learning method because it violates the automation process (41). Another drawback of this clustering technique is that the algorithm is only applicable if the mean is defined. In other words, all attributes (e.g. a_j^i in equation 3.1) defining the observations in the dataset must be numerical values. Categorical attributes should be normalized for robust clustering.

3.2. Computational Intelligence Techniques

Computational intelligence (CI) is a part of the well-known artificial intelligence field. CI is concerned with the development of intelligent agents. Such agents can learn from the gained experiences and knowledge to make inferencing, decision or prediction in

an intelligent way like human (42). CI is a more specific research area branched from artificial intelligence. In AI, representative knowledge and symbolic languages are commonly used for developing reasoning and cognitive systems, while CI analysis numerical information and reasons from numerical representations (43) (44).

3.2.1. Artificial neural networks (ANN)

Neural networks (NNs) is a very astonishing technique that is applied to a wide range of different fields including the well-known problem of intrusion detection. One of the promises side in this technique is the generalization feature. Generalization allow the modeled behaviors to represent every piece of information gained from any previous knowledge or experience. This property can be utilized to build an in-depth detection engine that can detect unseen attacks efficiently. Moreover, ANN have the ability to distinguish and detect patterns that can be applied to other parts or issues of IDSs like classification of attacks (45).

Artificial Neural Networks (ANN) are sets of mathematical functions that is inspired by biological neural networks (nerve cells). ANN often just called Neural Networks (NN). ANNs are constructed from a network of huge number of processing units that cooperate together to find the most appropriate solution to a specific problem. A particular processing unit is called a neuron which is doing some computational operations. Each neuron is connected to an activation function. A particular ANN is constructed from several layers where each layer is built from a set of neurons. Output of a particular layer is sent as input to all neurons found in the advanced layer. Outputs are calculated based on the weights of the connections related to these neurons. These weights are the result of the learning process which is an optimization process. In the learning process, the best set of weights for solving a particular problem are found and assigned to each connection (45) (43) (44).

In (44), authors reviewed all available alternatives for ANNs as shown in Figure 3.2 . The unsupervised ANNs called self-organizing maps (SOMs) uses the Euclidian distance to cluster a set of observations to a predefined number of clusters. SOMs are

popular class of NN and very common in anomaly-based IDSs. For example, SOMs are used to learn and model patterns of normal behaviors or activities. As drawbacks ANN have long training time. Moreover, in the case of unsupervised learning, the number of layers, the number of neurons per layer as well as the number of clusters must be chosen manually (44).

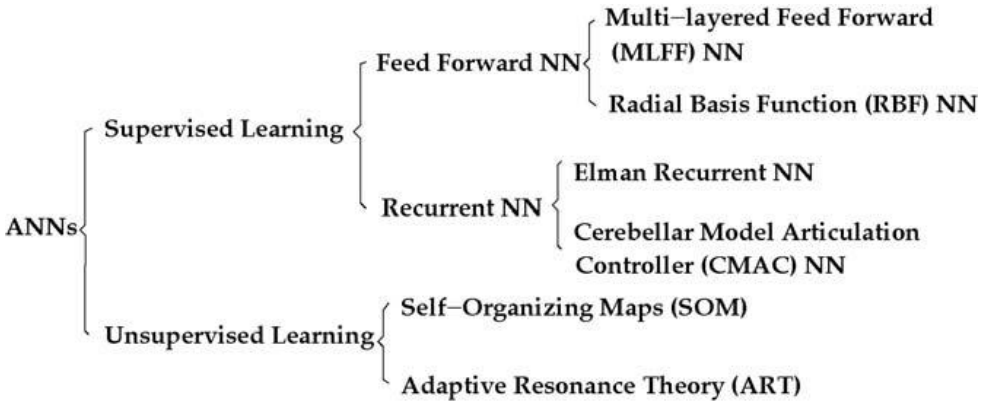

Figure 3.2. Types of reviewed ANNs (44)

The main common in the works done in (45), (44), and (43) is the evaluation dataset. They depend on The KDD cup 1998 evaluation dataset to assess their proposed approaches. For instance, the author of (45) evaluated the proposed method which is ANN-based using this evaluation dataset. In this work, a detection mechanism of the DDoS is proposed. DDoS is considered as an application-layer attack. Therefore, The KDD cup evaluation dataset is not applicable for the development of such IDS since it contains only lower layer (e.g. transport layer, network layer, etc.) based attack samples. However, different evaluation datasets are used in the development phases as explained in section 4.1. Moreover, AI-based approaches and particularly ML and DWT techniques are used to develop the proposed IDS.

3.2.2. The fuzzy logic

Fuzzy logic is another computational technique depending on fuzziness of facts. The degree of fuzziness is used instead of the variables itself. For example, rather than the traditional true or false Boolean values, truth degree is used. Moreover, fuzzy logic-based approaches assign each observation in the data space to various classes at the

same time with different degree of truth for each class. Fuzzy logic increases the robustness of IDSs since the boundary between the normal and the anomaly cannot be modeled in a well-defined model. In other words, fuzzy logic approach is applicable in computer security field since the computer security field contains fuzziness areas. For instance, in non-fuzzy logic approaches, deviation from the normal model even slightly may cause a false detection which make the system prune to high false positive rate. In the other hand, fuzzy logic-based approach can overcome tiny amount of deviation to ensure minimal rate of false positive which increase the IDSs robustness. The work conducted in (46) is a fuzzy based IDS where the rate of false positive in the proposed anomaly-based IDS is minimized to neglectable rate as in misuse-based IDS. Fuzzy logic techniques attempt to model the fuzziness and vagueness found in a set of observations or other samples to construct flexible patterns for detection hopefully greatly increases the robustness of detection systems (44).

Authors in (47) show that while applying fuzzy logic approach, the rate of false positive for an anomaly-based NIDS decreases noticeably. They model a fuzzy-based model using a set of fuzzy-based for discrimination between normal and anomalous behaviors. The genetic algorithm was utilized to build fuzzy-based model or classifier, which is a set of fuzzy-based rules. These rules have a common structure that is built using a set of weighted “if then” rules. A particular rule is molded by one or more fuzzy expressions related to each other by one or more fuzzy logic operators (e.g. fuzzy AND, fuzzy OR, and fuzzy NOT). Weights of rules are real numbers in [0; 1] that show the confidence of rules (43).

The most common evaluation dataset, KDD cup 1999, was used while conducting experiments in (46) to classify the anomalous attacks within the dataset space to 4 classes. Authors use fuzzy association rules for developing a novel NIDS. Both behaviors of network traffic are modeled relying on a set of fuzzy-based association rules. These fuzzy-based rules for both profiles are generated and modeled during the training phase. However, the proposed algorithm fails to achieve the misuse overall true positive rate while it mange to achieve low rate of the false positive rate which

was less than 3%. Moreover, various core methods of computational intelligence are explained in section 4 in (43) and section (44).

3.3. Machine Learning Techniques

The Machine learning (ML) is powerful techniques for building reasoning and knowledge discovery systems. ML techniques attempts to generalize observations by extracting and building common patterns. A learning machine can improve its internal patterns and knowledge base through dynamic learning. (5) ML techniques are applied to a wide range of fields in various life aspect like object recognition, prediction and search engines, medical diagnosis, and so on. ML allows machines to inference correctly so they can take the right decisions. In intrusion detection applications, a particular ML-based model, should be able to decide whether a particular behavior or activity is normal or anomalous based on the gained knowledge. ML-based, also called anomaly-based, IDSs attempt to discriminate between various behaviors based on deviations from expected behaviors. In other words, depending on modeling of normal activities or behaviors, they report deviations from that profile as an attack (48).

3.3.1. Linear support vector machines (SVM)

SVM is a supervised ML technique used for classification as well as regression tasks. SVM was introduced by Cortes and Vapnik. SVM produce one or more model according to number of available classes in training data. Then, during evaluation stage, the target classes of the testing data are predicted using that modeled classifier. Testing data is constructed only from the testing attributes. In this algorithm, each feature's value is plotted as a point in n-dimensional space where variable n is number of generated features or parameters in each observation in both of training and testing data. Value of each feature represents the value of a particular coordinate. A learning-based classifier is modeled from the training attributes during the training phase. This modeling process is completed by finding the hyper-plane which best separates all samples by differentiating the two classes very well as illustrated in Figure 3.3.

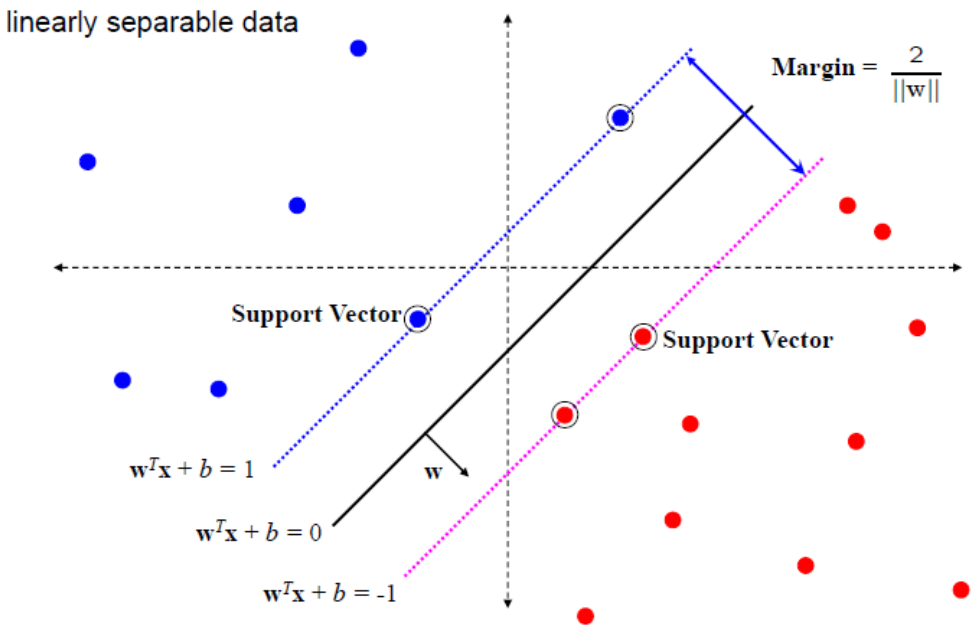

Figure 3.3. Linear hyperplane that maximizes the margins between binary classes (49)

3.3.2. IDS based decision tree (DT)

Decision Tree (DT) is a popular ML-based algorithm used for inferencing over supervised attributes (50). Despite its simplicity, DT algorithm can be used in many popular applications. For instance, DT can be used like a powerful decision-making tool in banking or marketing fields that can help managers to take the right decisions. As its name implies, a set of labelled observations can be modeled by a particular DT algorithm to constructs a classification-based model that looks like a tree structure. During making decision, starting from the root node each non-leaf node is an internal analysis or investigation on a single attribute leading to the final decision. Before making the final decision, the tree structure is traversed down to one of the leaf or decision nodes. The leaves nodes indicate the predicted value of the "class" attribute. Attributes of data space are split by the edges of the tree structure (50) (51).

3.4. Classification-based Proposed Model

During this work, SVM classification algorithm is utilized to build the DDoS discriminating model that can separate or predict new unseen network traffic samples to the actual profile (i.e. legitimate traffic or DDoS GET Flooding traffic). During

filtering step, removing unnecessary network traffic is executed. Also, the key features that represent each network traffic sample is extracted from traffic. Then, these extracted features are summarized based on time window. After feature preparation step, the DDoS anomaly-based attack detection model is used to assign each window to the predicted class. The main contribution of this work is a supervised anomaly IDS that perfectly predict or detect any anomalous DDoS traffic within network traffic samples. This is an in contrast to (22) where a rank showing the degree of being attack or normal is assigned to each instance of network activities. This score is assigned based on an unsupervised learning model. The work done in (21) is very close to the work in this thesis, the authors model only the normal samples using OCSVM (once class SVM) algorithm to build a classification anomaly-based classifier. Whereas in this work a binary SVM algorithm is used to build such classifier.

3.5. DWT Technique

The Discrete Wavelet Transform (DWT) is used to convert a signal from the time domain to the frequency domain and visa verse. DWT is powerful technique that can represent the time series data using less than 2% of the original time series data by decomposing and reconstructing the original signal. The capacity of wavelet representations to concentrate signal energy in few coefficients is the key of efficiency (52). In this work, DWT is mainly used to decompose and reconstruct the time series data within a predefined time window of 60 samples. This number is chosen without thought. Each window is composed of three time-series vectors representing the three selected features. By applying DWT technique, the complete window is represented only by one sample network traffic. In other word, the window containing 60 network traffic samples can be represented and classified only using one sample of network traffic that have three parameters. Rather than classifying 60 consecutive samples of network traffic, only one sample is processed during the classification phase by the anomaly-based classifier. This technique can increase the performance of the proposed IDS by reducing the time required to process all these samples by the classifier. The time required to re-represent the window using DWT is less than the time required to process all the samples by the anomaly-based classifier.

The efficiency of the proposed IDS is also improved while utilizing this technique, particularly reducing the false alarm rate which can consume the system administrator resources. Also, it can be useful when there it is required to analyze the offline logs generated by the proposed IDS (e.g. building training datasets). The amount of the logs can be reduced noticeably.

CHAPTER 4. EXPERIMENT

In this section, the technical description of developing the proposed IDS is described e.g. tools, datasets, etc. Starting by discussing the datasets and its characteristics and then moving to the tools or software which used to implement each step in this work. The extraction of the selected features is covered in detail. Moreover, Tools and other command lines used in the extraction steps are listed alongside their required arguments. These features represent the core of the proposed IDS as they are used to model its engine using learning-based techniques. The proposed IDS attempt to predict the status of the traffic during the final experiment. Finally, evaluation of the proposed IDS is discussed.

4.1. Datasets

In this sub section, the introduced datasets in this work are defined and presented in detail. Two separated datasets are used to accomplish this work. Both datasets are publicly available to all researchers. These datasets were chosen carefully so a set of behavioral features can be extracted from the original features of these datasets. Then, an attempt to use these extracted behavioral features to discriminate the actual traffic perfectly. In addition, all the steps and procedures that were followed to complete this work are shown and explained briefly. In this work, the proposed system is evaluated using two different network traces or datasets representing the normal and the anomalous profiles. The first, The UNB ISCX 2012 evaluation dataset is used to represent the anomalous traffic while the other, the 1998 FIFA World Cup traces is a legitimate or normal HTTP GET requests.

4.1.1. The UNB ISCX 2012 evaluation dataset

The UNB ISCX 2012 evaluation dataset was built by the Information Security Centre of Excellence (ISCX) at University of New Brunswick (UNB). It is available to public since 2012 (39). The UNB ISCX 2012 dataset includes 7 weeks of labeled network traces with full packet payloads captured in PCAP format from an actual and controlled network environment. Countless parameters or features, e.g. source address, total source bytes, protocol type (TCP, UDP, and ICMP), etc. can be extracted from each trace or tuple inside the dataset using a proper analysis software tool. Figure 4.1 shows a sample set from the traces captured during 15th of June capturing activity. The audited network traces are parsed using Wireshark network analyzer (any other network packets analysis tool may be used instead). Each connection is traced and analyzed separately to show the detailed information, e.g. the protocols hierarchy, timing information, etc., related to each connection in an organized format.

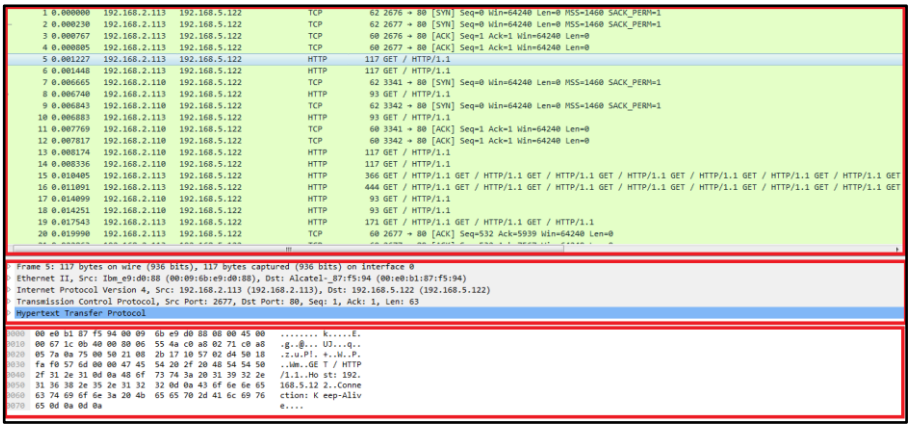

Figure 4.1. Network traces with their packet payload captured during the 15th of June auditing activity.

Using Wireshark analyzer, the analysis screen contains three sections by defaults. The upper part shows the captured network traces inside the PCAP repository file in separated tuples with a summarized information related to each trace. In the middle part, detailed information about the underlying protocols and their fields involved in initiating this connection. Protocols are displayed alongside their detailed information in a specified format where the lowest level protocol is listed at the top alongside all its fields with their corresponding values. Then, the next protocol is displayed

alongside its fields and their corresponding values and so on. At the bottom, the high level used protocol, HTTP protocol, is listed with all its fields alongside their corresponding values as illustrated in figure 4.1 above. At the last section, the payload or the packet content related to the selected network trace is displayed in bytes.

Moreover, The UNB ISCX 2012 evaluation dataset is represented in XML format. In this format, the relevant profile, labeling as either normal or anomalous, for each network trace can be found in the last tag called “Tag” inside the xml file available alongside each sub dataset file. The UNB ISCX IDS 2012 evaluation dataset is publicly available for researchers through contacting the author. Figure 4.2 shows a random anomalous network trace from the UNB ISCX IDS 2012 evaluation dataset where the assigned value for the last tag clearly indicates that this trace is an anomalous network trace or session.

Figure 4.2. XML representation for a random anomalous network trace with its relevant profile captured during the 15th of June auditing activity.

The UNB ISCX 2012 evaluation dataset consists of 7 sub datasets containing all network traffic including payload of normal and malicious activities generated during the individual days. The data audited during the fifth day, Tuesday, 15/6/2010, contains the traffic generated by the HTTP GET flood DDoS activity containing more than 23 gigabytes of captured data. The large size of the captured traffic is due to capturing the packet payload alongside their header, rather than only capturing the headers. The attack is started by the bots’ master and run for 60 minutes (39). Editcap, a part of Wireshark, is used to split the 24-gigabyte main PCAP testbed file to sub

PCAP files. Using Editcap tool simplify the analysis stage. Each split sub PCAP file contains only audited data during only one sampling interval. Then, tshark tool, also a part of Wireshark software, is used to analyze the split sub PCAP files and extract the chosen parameters for the given interval.

4.1.2. The 1998 FIFA World Cup traces dataset

In the other hand, for a real-world normal HTTP GET requests, the 1998 FIFA World Cup Dataset (40) provided by the Internet Traffic Archive is used. The 98 world cup traces are legitimate HTTP GET requests containing all the HTTP requests made to the 1998 FIFA World Cup site during 92 days from April 30, 1998 to July 26, 198. During this period, HTTP requests were logged to common log format files by the site logging system where each HTTP Get request logged to a separate record alongside its attributes. Request's attributes set contains client source IP, internal requested URI address, request time formatted to GMT in addition to other fields. Private data like source IP addresses and user names in the network traces have been removed and only IP addresses replaced by unique integer identifiers which are preserved throughout the dataset. To ensure privacy, the mapping file for the source IP address pseudo nomination is not publicly available. Logging files are organized by a day based numbering system. The 1998 FIFA World Cup Dataset is composed of the access logs collected from different servers used in the World Cup Web Site. Each access is an entry or a tuple containing seven values or fields explained in table 4.1.

4.2. The Proposed Model

Figure 4.3 shows the architecture of the proposed IDS. It is a common architecture consisting of two main parts, data collection and analysis model as described in 2.1.6. First, the raw network traces are sent to the processing stage regularly each sampling period. The relevant features can be extracted and calculated by analyzing different fields from different levels inside the captured network packets or traces. Then, the extracted features are dispatched to SVM based classifier which assign it to a relevant profile. The classifier is trained previously with a training set of network traces from

both profiles with the same selected features set explained in section 4.3. ML-based classifier is a binary classifier which labels a negative profile with “0” and a positive profile with “1”. Positive and negative profiles are discussed in section 2.1.2.

Table 4.1. The 1998 FIFA World Cup Dataset’s Fields.

Field	**Description**
clientID	Client identifier.
Timestamp	GMT time of receiving the request by the site.
Method	request method.
Requested object	The URL address of the requested object (e.g., image, HTML file, etc.) inside the hosting server.
Status	HTTP version in addition to response status code.
Size in bytes	Number of bytes in response.

The analysis model, the core of the proposed IDS, produces a decision regularly every sampling period based on the analyzed network traffic. The sampling period is equal to one second. The output of this sampling period can be considered as the final decision where the system admin is responsible for monitoring and managing such huge amount of decisions. To avoid that situation, the produced outputs or decisions during the next n sampling periods are stored locally for advanced analysis to produce a summarized decision each n sampling periods.

Within the advanced analysis, instead of generating the prediction regularly every second, it will be generated regularly every minute. DWT technique will be applied to reduce and summarize the 60 records. The summarized output representing one minute will be fed to the prediction engine for the final prediction. The prediction result labels the traces during the previous period as positive or negative. The period is equal to 60 seconds, but it could be any period.

4.3. The Selected Features

Anomaly based detection method implies that the proposed mechanism will concentrate on the traffic behavior rather than the signature or the structure of the

traffic generated by the http requests. This is due to the complete similarity between malicious and legitimate HTTP GET requests from structural prospect. Therefore, to differentiate between legitimate and illegitimate http traffic, the HTTP GET requests behavior will be monitored and analyzed by extracting some relevant features from the inbound traffic generated by all http requests.

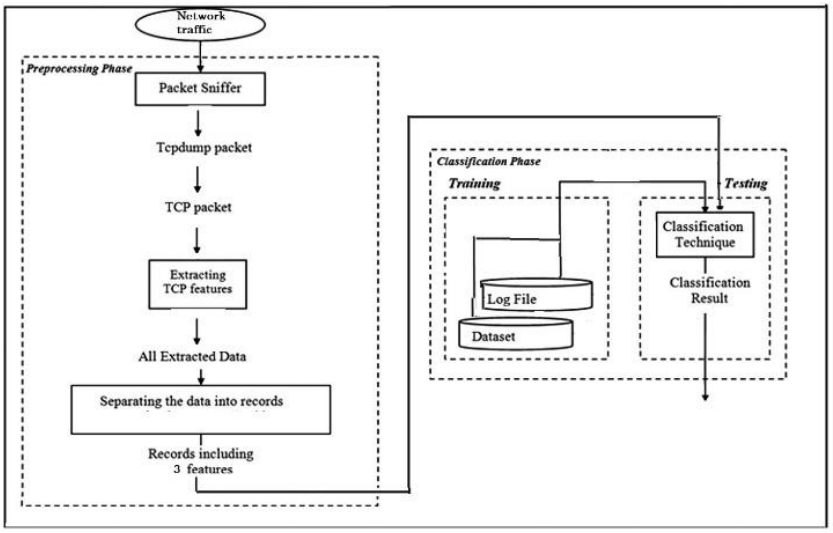

Figure 4.3. The structure of the proposed IDS

Request generated from HTTP GET flood attack are generated fundamentally by the malicious software which are usually the same for all the compromised machines that connected to a single botnet. As a result, enormous number of requests with high similarity and low diversity can be found while analyzing the fields related to host and URI addresses. The malicious software is uninspired, so they do not have the ability or the creativity to choose among the internal retrievable objects or URI addresses from the victim site. Therefore, the overwhelming traffic often converged to a small static group of URI address(s). On the other hand, the HTTP GET requests generated by the normal or human clients or even search engines are scattered to a large group of URI addresses.

The first and foremost feature is the URI diversity for the incoming requests. Since the malicious tools used to launch the attack from the compromised source machines behaves the same for all the machines mastered by the attacker, the request flows

originating from the malicious sources are similar. In contrast, the request flows from the legitimate human clients are distributed and mostly their requested URI addresses have large diversity among them. Second, the request rate for the requested URI address with the maximum frequency among all the requested URI addresses. Normal users tend to request any online resource for limited times during a short period of time, whereas the mastered bots send massive requests for the same objects over the same TCP connection or stream. This sophisticated parameter can be used to discriminate the anomalous request even having high diversity among the requested URI addresses. Third, request rate per second is also extracted for perfect discrimination.

4.4. Experiment

The two involved dataset files are split to sub files. Each split file the network traffic within different sampling periods. The next step was to process each split file to extract the selected features. The generated features or parameters are time series data represented with integer values. These new parameters represent a new dataset used as the main dataset in training and building ML-based classifier.

For traffic discrimination, Linear Support vector machine classifier is used to discriminate the traffic based on the selected attributes. The classifier is modeled firstly using the training part of the generated time series dataset. Then, the trained model can be used to classify the new or unseen dataset to a relevant profile in the evaluation part.

4.4.1. The development environment

The experiment is carried out using a laptop machine with Intel® Core™ i5 M430 CPU @ 2.27 GHz, and with 6.0 GB of 1333 MHz DDR3 RAM. The experiment is performed using several software tools. Each tool is used during a specified stage of the experiment to achieve a determined output at that stage. The following sub sections describe these tools and show they are used and commanded to achieve the desired

output during each step of the experiment. These tools will be listed according to their need in the experiment, the first used tool will be listed first and so on.

4.4.1.1. Wireshark network analyzer

Wireshark network analyzer is a software tool for analyzing network traffic. Wireshark versıon 2.2.5 is used during the early stage i.e. to implement the data collection model see 2.1.6. It is used to split the genuine PCAP file containing all the traffic captured during 15 of June auditing activity. Wireshark network analyzer is responsible for splitting and extracting the required network traffic from the whole PCAP file, then analyzing these split network traces to mine the determined set of statistics or information.

At the first step, a PCAP file is split using the tool called “editcap”, a part of Wireshark network analyzer. Edditcap tool extract and split only the traffic generated during the attack period which continued for about one hour (39) from the whole PCAP file to another temporary sub PCAP file as the output of the first step. The size of the whole PCAP file is about 24 GB, therefor, processing such massive size of network traffic requires a robust machine with an enough memory installed. The complete editcap command configured with the appropriate arguments is shown in figure 4.4

```
editcap -A "2010-06-15 21:04:45" -B "2010-06-15 22:04:44" testbed-15jun.pcap out.pcap
```

Figure 4.4. Editcap command line that used to split the main pcap file

In the next step, the anomalous traffic is collected by splitting the inbound HTTP traffic destined to the site’s main web server from the output file of the previous command. In this step, another tool called “tshark” is used to implement this step as shown figure 4.5.

```
tshark -r out.pcap -Y "ip.dst==192.168.5.122 and tcp.port==80" -w HTTPGETREQUESTS4DDoS.pcap
```

Figure 4.5. Tshaark command line that used to seperate the attack traffic

Again, editcap is used to split the last output PCAP file containing about one hour of the anomalous network traffic. Figure 4.6 shows the command that split that PCAP file to sub PCAP files representing a sampling period of one second.

```
editcap -i 1 HTTPGETREQUESTS4DDoS.pcap HTTPGETREQUESTS4DDoS.pcap
```

Figure 4.6. Editcap command line that used to extract the anomalous traffic regularly every second

The previous command, editcap, uses an indexing system to name the new generated PCAP files based on the name of the input PCAP file.

4.4.1.2. Cygwin

Cygwin is a large collection of software tools which provide functionality like a Linux distribution on Windows. Several Cygwin tools are used to process the output of the tshark tool. Tshark tool produce a lot of unimportant or unused statistics beside the required information. Therefore, a proper Cygwin tools or commands are used to process the output generated by each tshark tool to keep only the required bits and neglect any other data. For Example, the output of the tshark tool shown in figure 4.7 is fed directly to a set of Cygwin tools (e.g. uniq, sort, head, etc.) that will finally give an integer number representing the number of requested objects inside the processed sub PCAP file which represent the network traffic during 1 second sample period.

The previous command is executed with all the sub PCAP files representing the different one-second sampling periods. Therefore, it is run inside a loop where the symbol coming after the double percent sign represent a variable for the input sub PCAP file. This variable is defined within the outer loop which iterates all the one-second split sub PCAP files. The time series output will be directed to a CS- formatted file representing one of the three selected parameters described in section 4.3. .

```
tshark -r %%f -qz "io,stat,0,COUNT(http.request)http.request" | head -13 | tail -1 | gawk '{print $6}' >> out.csv
```

Figure 4.7. Generating time series information for first feature using tshark and cygwin commands

Like the previous tshark command, the following two commands generate two CSV formatted files. The same way, all the one-second split sub PCAP files are iterated and processed by both tools (i.e tshark and Cygwin). The processing output will be directed to a CSV formatted file containing the time series data related to the other two selected parameters described in section 4.3. above in this chapter.

```
tshark -r %%f -T fields -e http.request.uri | tr " " "\n"| sort | uniq | wc -l >> http_req_uri_div_ps4DDoS.csv

tshark -r %%f -qz http_req,tree | head -8 | tail -1 | gawk '{print $2}' >> max_requested_uri_ps4DDoS.csv
```

Figure 4.8. Generating the other two selected features using tshark and cygwin command lines

In figure 4.9, the time series data represents the anomalous traffic is plotted. These data were generated by processing the evaluation dataset during the previous steps. Three different graphs show the selected three features. In parallel, the other dataset is processed in the same way as coming in the next sub section.

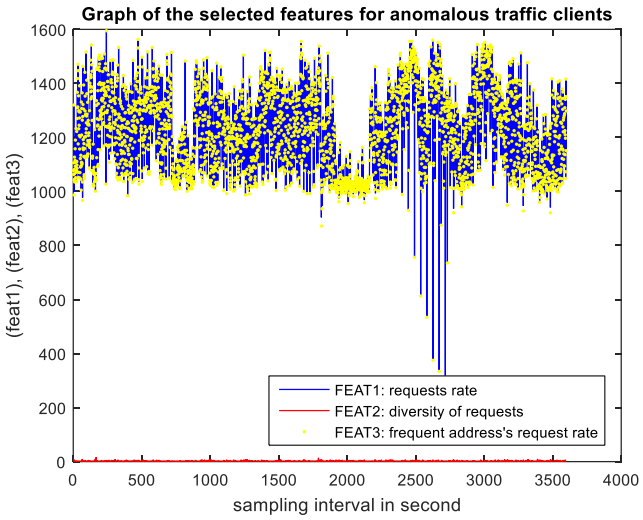

Figure 4.9. Selected Features for anomalous traffic generated by processing the 15th of June auditing activity from The UNB ISCX 2012 evaluation dataset records.

4.4.1.3. Binary2Common log format tool

A special tool is used to transform the binary access logs files to the common log format. This tool is provided alongside the dataset files and can be downloaded for

free. (40) Moreover, Cygwin package is used to compile and run this tool as it is written using C Programming Language. This tool is used for format conversion to reduce analysis time in addition to reduction of storage size required for the 1998 World Cup Web Site logs. Each instance represents a single HTTP request with a common structure. The format of a request in the binary log is shown in figure 4.10.

```
struct request
{
uint32_t timestamp;
uint32_t clientID;
uint32_t objectID;
uint32_t size;
uint8_t method;
uint8_t status;
uint8_t type;
uint8_t server;
};
```

Figure 4.10. The structure of 1998 World Cup access logs

This tool is provided to simplify working with the binary log files from the 1998 World Cup Web site. The source code for the tools along with a more complete description of how to use these tools is available in the tool archive. The following command converts the compressed binary log file named "`wc_day65_7.gz`" back to the Common Log Format and direct it to the output file named "`wc_day65_7.out`".

```
gzip -dc input/wc_day65_7.gz | bin/recreate state/object_mappings.sort > output/ wc_day65_7.out
```

Figure 4.11. Cygwin command to run 1998 World Cup Web access logs convertion tool.

Figure 4.12 is an example of a converted access log entry from binary format to common log format. This entry tells us that on June 10th, 1998, at one second past midnight, local time in France, the client 192.168.0.1 asked this server for the file /index.html. The server was informed that the client supported HTTP/1.0. The server successfully responded to this request (this is indicated by the status code of 200) and transferred 1,000 bytes of content data to the client.

192.168.0.1 - - [10/Jun/1998:00:00:01 +0200] "GET /index.html HTTP/1.0" 200 1000

Figure 4.12. An example for Common Log Format after conversion from the binary format.

In the next step, any appropriate analysis tool can be used to analyze CSV formatted data(in this work, Matlab is used). This appropriate analysis tool processes the access log format files converted in the previous step to calculate the time series data representing the normal profile. Figure 4.13 shows the graphs of the three selected features demonstrating the normal profile.

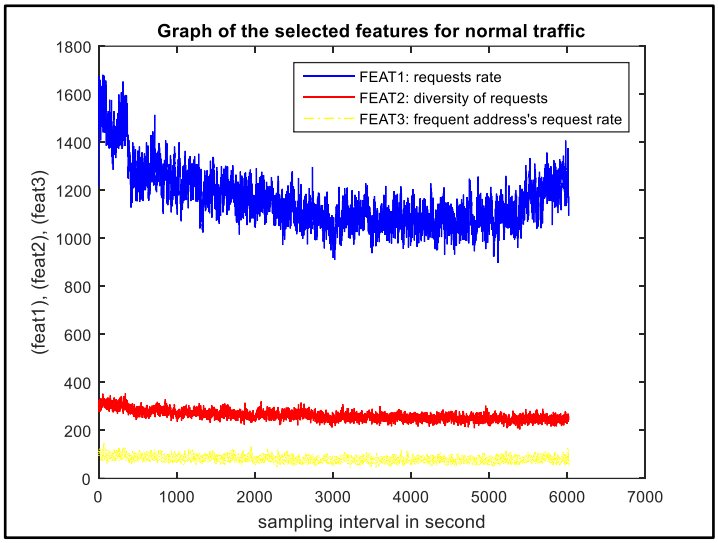

Figure 4.13. Selected Features for Legitimate traffic generated by processing 1998 World Cup Datasets records.

During previous step and while processing the UNB ISCX 2012 evaluation dataset, a different analysis tool, Wireshark, was used. Whereas in this step totally different tool is used to process the 1998 FIFA World Cup Dataset. This is due to the different formats of the two datasets representing the two profiles. The result of processing these datasets is CSV formatted files containing a time series data in an integer format representing the selected parameters (three parameters for each profile) as shown in figure 4.9 and figure 4.13. These data is fed to Matlab environment in the next stage to prepare the final model.

4.4.1.4. Matlab

The software environment used for running the experiment and evaluating and the proposed mechanism is Matlab R2015a software. Wavlet Toolbox Version 4.14.10 and Statistics and Machine Learning Toolbox Version 10 is used to train and build the proposed classifier. In the first step, the extracted time series data, in the previous step is processed with the windowing technique. Instead of processing each observation alone by the proposed system, a set or a window of observations are fed to that system. The window size is set to 60 sampling periods that equal one minute with a sliding of 30 sampling. This means that every 30 sampling periods (30 seconds) a new window (with 60 samples) is fed to the system as shown in figure 4.14 for the final decision.

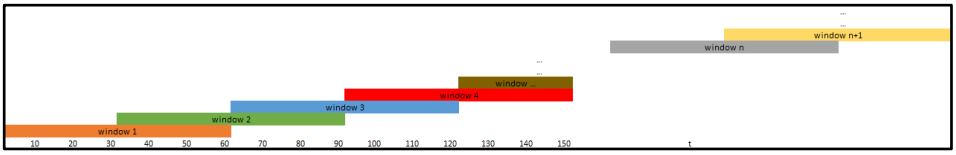

Figure 4.14. Sampling with window size equal 60 seconds and overlabing equal 30 seconds.

Then, a new technique called Discrete Wavlet Transform (DWT) is used to reduce the size of the collected window. This technique summarizes the 60 fed samples of the current window by extracting the potential energy using a symlets4 wavelet or sym4 at level 10. This means that instead of feeding the whole window, only the summarized observations are fed to the next step. This step supposed to increases the performance of the proposed system by reducing the processing time while making the final decision by the model. Figure 4.15 shows the graphs of the new three features after applying DWT technique over the time series data for the normal profile.

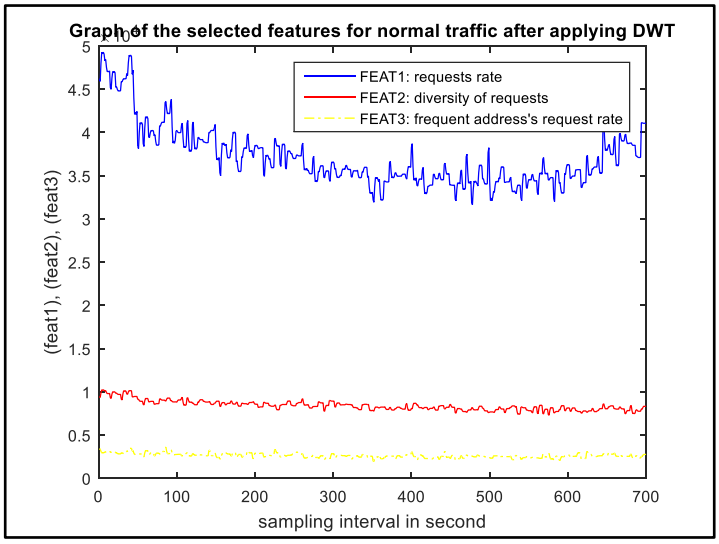

Figure 4.15. The DWT-based new generated features representing the normal traffic

It is obvious from the graph 4.13 and the graph 4.7 that the 6019 x 3 time series matrix data is reduced to a lower dimension of 700 x 3 matrix data. The number of attributes (columns) in addition to the profile attribute are the same after applying the DWT technique. Only the number of observations is reduced by summarizing the energy inside the whole observations.

The calculated time series data or parameters alongside their profile are stored in an array. The profile attribute has only two values. The first value equal to one means a negative observation, whereas the other value equal to zero means a positive observation. The dataset is divided to two parts. Training part is used while building and training the model. The other part, testing, is used for testing and evaluation of the proposed model. The actual profiles vector found in the testing dataset is compared with the predicted profile generated by the proposed model. The correct rate is the percentage of the correctly classified observations in the testing dataset. The following Matlab code pieces are used to build and train the proposed intrusion detection model with the default configuration for DT, and SVM algorithms respectively.

```
model_DDoS_tree=fitctree(training(:,1:end-1),training(:,end));

model_DDoS_svm = fitcsvm(training(:,1:end-1),training(:,end));
```

Figure 4.16. Matlab code pieces used to train the proposed IDS's model

The set named “training” contains three vectors representing the selected parameters in addition to the profile vector. Then, the trained model generated in the previous step is tested with a different new set using only the first three vectors representing the selected parameters. The last vector represents the actual profile and it is important in the evaluation stage. The proposed model classifies the new dataset by assigning each observation to a relevant profile (e.g. 1 for negative profile and 0 for positive profile). Matlab commands shown in figure 4.17 is used to test the trained intrusion detection model in the previous step named “model_DDoS_svm” with the new dataset called “testing”. The output vector is named “IDX_DDoS_svm” which contain the predicted profiles assigned to the dataset observations by the model.

```
IDX_DDoS_svm = predict(model_DDoS_svm,testing(:,1:end-1));
```

Figure 4.17. Matlab code piece used for predection using the trained model

Finally, in the evaluation step, the predicted profile named “IDX_DDoS_svm” is compared with the actual profile using Matlab code shown in figure 4.18 to evaluate the proposed model.

```
CP_DDoS_svm = classperf(testing(:,end),IDX_DDoS_svm,'Positive',1,'Negative',0);
```

Figure 4.18. Matlab code piece for evaluating the trained model

4.4.2. Results

In this section, the results of the conducted experiment are presented as an evaluation to the proposed NIDS. As explained briefly in the previous section, the experiment is carried out in several steps using a set of suitable software tools. The performance of the proposed NIDS is measured during the final step of the experiment conducted using Matlab environment. Also, the effects of applying the DWT technique on that system is evaluated at the same phase. During the experiment, sampling period is set to one second which means the proposed features are calculated regularly each second

from the analyzed traffic. The window size is set to sixty samples meaning that a window represents the traffic within one minute.

4.4.2.1. Binary classification

Due to the vast differences found between the values of the selected features representing both traffic profiles, these suggested features together can be used as efficiently discriminating parameters to discriminate a real-world HTTP GET flood DDoS attack from legitimate clients. To show the effectiveness of the second and third parameters, during the first experiment, only the first parameter was used alone to build and train the detection engine. Table 4.2 shows the result for the classifier built using data extracted only from the first parameter representing the number of requests per sample interval.

Table 4.2. Experimental result for the proposed model based on the SVM algorithm and only feat1.

Traffic type	operation	# samples	Required time (ms)	Accuracy
Both profiles	Features extraction (only feature-1)	9682	220.0 / sample	-
Both profiles	Training/testing (10-fold cross vaidation)	9682	002.0 / sample (predicting)	62.17%

The result show that the first feature alone is inefficient to train the detection engine. This means new discriminating features must be proposed for building and training the classifier model that can increase the ability of the detection engine to predict real anomalous traffic more efficiently.

In the next experiment, the other proposed featured, named URI diversity for the incoming requests, and the request rate for the most frequent requested URI address, are used with the first feature, named the number of requests, to increase the efficiency of the detection engine. The extracted time series data for the previous illustrated features is used together with the data from the first feature to build learning-based classifier using the training part of the generated data. Then, the built model is

evaluated using the test part, totally different part, of the generated data. The built model can discriminate the legitimate and the flood traffic perfectly. Table 4.3, 4.4 and 4.5 show the actual result based on feat1/feat2, feat1/feat3 and feat1/feat2/feat3 respectively using SVM learning algorithms.

Table 4.3. Experimental result for the proposed model based on the SVM algorithm and only feat1/feat2.

Traffic type	operation	# samples	Required time (ms)	Accuracy
Both profiles	Features extraction (feat1 & feat2)	9682	440.0 / sample	-
Both profiles	Training/testing (10-fold cross vaidation)	9682	002.0 / sample (predicting)	100.00%

Table 4.4. Experimental result for the proposed model based on the SVM algorithm and only feat1/feat3.

Traffic type	operation	# samples	Required time (ms)	Accuracy
Both profiles	Features extraction (feat1 & feat3)	9682	440.0 / sample	-
Both profiles	Training/testing (10-fold cross vaidation)	9682	002.0 / sample (predicting)	100.00%

Table 4.5. Experimental result for the proposed model based on the SVM algorithm and feat1/feat2/feat3.

Traffic type	operation	# samples	Required time (ms)	Accuracy
Both profiles	Features extraction (feat1 & feat2 &bfeat3)	9682	660.0 / sample	-
Both profiles	Training/testing (10-fold cross vaidation)	9682	002.0 / sample (predicting)	100.00%

The training process occurs only once during the training phase of the proposed system. In other word, training will not take place inline. Therefore, the training phase is not considered while evaluating the proposed system performance. The main two factors for the evaluation are model accuracy and the required time to process and assign each sample inside the current captured window of the network traffic. The proposed model takes about 662.0 milliseconds to represent and assign each sample

inside the current window to a relevant profile. As illustrated in tables 4.2 – 4.5 significant part of the processing time is consumed while extracting the features representing each sample. This is due to the fact that the proposed system read the captured data offline from access log files from the hard disc. In real time capturing, pipelining or buffering could be used for inter-process communication where the capturing process can send data to it and analyzing process can read it directly from the system memory which could reduce the processing time significantly. like concurrent session inspection, this issue is as an advanced and it is out of the scope of this work and it may be a proposed during my future works. However, the proposed model manages to assign each traffic sample to the correct profile. In other words, it detects any anomalous traffic perfectly within two milliseconds. The performance evaluation is carried using the Matlab built-in tool called “tic toc”. It is used to measure the elapsed time for any script or program. For example, figure 4.19 shows how “tic toc” build-in tool can be used to measure the performance for a SVM-based classifier under Matlab environment.

```
73
74      % tic
75      % IDX_svm = predict(svm_model,test(1:1,1:end-1));
76      % toc
77
```

Figure 4.19. Built in tic toc command for elapsed time measurment under Matlab environment

4.4.2.2. DWT analysis

The same experiment, conducted with the time series data in the previous step, is conducting with the data generated from the DWT technique. There for the time required for predicting or clarifying the whole window of 60 samples, 180 milliseconds. The aim of this step is to evaluate the effects of applying the DWT technique on the performance of the proposed system. Table 4.6 show the actual result and the elapsed time to build and test the proposed model after applying the DWT technique. The results are based on SVM algorithm.

As shown in table 4.6, applying the DWT technique on the proposed IDS can increase the performance of that system. The time required to generate one a dwt-based sample representing each window of the traffic alongside the time required to predict that generated dwt-based sample is less than the time required to predict each sample within each traffic window. Inequation languages, it is possible to say that the total of 4.5 milliseconds and 3 milliseconds is more less than the result of multiplying 3 milliseconds by the window size (i.e. 7.5 <<<< 180).

Table 4.6. Experimental result for the proposed model based after applying the DWT

Traffic type	Operation	# samples	Required time (s)	Accuracy
Both profiles	Features extraction (feat1 & feat2 &bfeat3)	60 (60 sample per window)	(39,600) 0.660 * 60	-
Both profiles	Decomposition / Reconstruction	60 (60 sample per window)	0.0045	-
Both profiles	Training/testing (10-fold cross vaidation)	1664	0.025	100.00%

By increasing the window size this technique can increase the performance of the proposed system, however it may increase the vulnerability to attacks. The same as in the previous experiments, the training time is outside the performance evaluation since it is carried only as the very first step for building and training the model. To classify each sample within the network traffic, only the predictor is required to assign samples to the right profile (i.e. normal or anomalous). Therefore, training time is neglected. The features extraction step is a common prerequisite for both techniques which incur the same time overhead per sample.

CHAPTER 5. CONCLUSION

Application-layer DDoS attacks detection is a hot and difficult research topic in the field of intrusion detection. Based on the behavior of DDoS attack, this work proposes a novel approach to detect DDoS attacks. This work provides two contributions: DDoS attack detection based on behavioural parameters and representation of the extracted features to concentrate the energy inside the time series data of these parameters using DWT. The detection scheme against DDoS attacks is proposed, and it can achieve high detection efficiency and flexibility as the expierments show. In our future work, more advanced application-layer DDoS attacks will be discussed. The main limmitation is the labelled and the evaluation dataset. This limitation is a big obstacle for developing more advance anomaly-based detection mechanism that can encompase all advances and sever types of DdoS attack. Moreover, part of the available evaluation dataset related to this attack are restrictted to a didicated geographical regions (e.g. CAIDA dataset). This indicate the urgent need for creating publically available dataset for most recent security breach. Such dataset gives the researchers the ability to develop and evalute more acurate IDS.

Moreover, more advanced techniques and tricks can be implemented for increasing the robustness of the IDS. Such techniques should be subtle to discriminate DdoS attack from the legitimate trafic. For instance, rather than inspecting all the traffic, each connection by itself can be inspected and try to evaluate if a connection is a legitimate connection or an attacker bot based on parameters like the diversity of the requested objects per session that can be monitored. Then, the legitimate user behavior can be compared with bot behavior for intrusion detection. If a connection is suspected to be a bot connection, that connection can be prohibited from keeping its connection alive and forced to initiate a new connection.

REFERENCES

[1] Anomaly Detection: A Survey. VARUN CHANDOLA, ARINDAM BANERJEE ve VIPIN KUMAR, ACM, 2009, ACM Comput. Surv., Cilt 41, s. 58.

[2] Anomalous Payload-Based Network Intrusion Detection. Wang, K. ve Stolfo, S.J. . Berlin, Heidelberg, Springer, 2004. Recent Advances in Intrusion Detection. RAID 2004. s. 203-222. 978-3-540-30143-1.

[3] An Evaluation of Anomaly-Based Intrusion Detection Engines for Mobile Ad Hoc Networks. Panos, C., Xenakis, C. ve Stavrakakis, I. Berlin, Heidelberg : Springer, 2011. Privacy and Security in Digital Business. Cilt 6863, s. 150-160. 978-3-642-22890-2.

[4] Network Anomaly Detection:Methods, Systems and Tools. Monowar, Bhuyan H., Bhattacharyya, D. K. ve Kalita, J. K., IEEE , 2014. IEEE COMMUNICATIONS SURVEYS & TUTORIALS. Cilt 16.

[5] Nguyen, Thuy T.T. ve Armitage, Grenville. A survey of techniques for internet traffic classification using machine learning. Communications Surveys & Tutorials . 2008, Cilt 10, 4, s. 56 - 76.

[6] Intrusion Detection and Correlation: Challenges and Solutions. Christopher, Kruegel, Fredrik, Valeur ve Giovanni, Vigna. Springer, 2005, Advances in Information Security, Cilt 14. eBook ISBN:0-387-23399-7.

[7] Towards a taxonomy of intrusion-detection systems. Debar, Herve´ , Dacier, Marc ve Wespi, Andreas., Elsevier Science, Computer Networks, 1999.

[8] Lazarevic, A., Kumar, V. ve Srivastava, J. Intrusion Detection: A Survey. MANAGING CYBER THREATS Issues, Approaches, and Challenges. Boston, MA : Springer, Boston, MA, 2005, Cilt 5, 2, s. 20-77.

[9] George, Farah. nformation Systems Security Architecture: A Novel Approach to Layered Protection. A Case Study. SANS Institute Reading Room. [Çevrimiçi] 09 Sep. 2004. [Alıntı Tarihi: 15 Nov. 2017.] https://www.sans.org/reading-room/whitepapers/auditing/information-systems-security-architecture-approach-layered-protection-1532.

[10] SANS. Intrusion Detection Systems: Definition, Need and Challenges. SANS Institute InfoSec Reading Room. [Çevrimiçi] 2001. [Alıntı Tarihi: 15 Nov. 2017.] https://www.sans.org/reading-room/whitepapers/detection/intrusion-detection-systems-definition-challenges-343.

[11] Vulnerability. National Institute of Standards and Technology. [Çevrimiçi] NIST. [Alıntı Tarihi: 17 Nov. 2017.] https://nvd.nist.gov/vuln.

[12] CVSS Severity Distribution Over Time. National Institute of Standards and Technology. [Çevrimiçi] NIST. [Alıntı Tarihi: 17 Nov. 2017.] https://nvd.nist.gov/vuln-metrics/visualizations/cvss-severity-distribution-over-time.

[13] Evaluating Intrusion Detection Systems: The 1998 DARPA Offline Intrusion Detection Evaluation. Lippmann, R. P., ve diğerleri., DARPA Information Survivability Conference and Exposition, 2000. s. 12-26.

[14] Ghorbani, A. A., Lu, W. ve Tavallaee, M. . Chapter 7: Evaluation Criteria. Network Intrusion Detection and Prevention : Concepts and Techniques., Springer, Advances in Information Security, 2009.

[15] Microsoft®Computer Dictionary, fifth edition ed., ch. A,. [safari IT book online], Microsoft Press, 2012.

[16] Bandwidth, Packets Per Second, and Other Network Performance Metrics. Cisco Security Research & Operations. [Çevrimiçi] Cisco. [Alıntı Tarihi: 19 Nov. 2017.] https://www.cisco.com/c/en/us/about/security-center/network-performance-metrics.html#3.

[17] Towards Survivable Intrusion Detection System. Dong, Yu ve Frincke, Deborah, Proceedings of the 37th Hawaii International Conference on System Sciences, IEEE, 2004. DOI: 10.1109/HICSS.2004.1265702.

[18] Weiss, S. M. ve Zhang, T. The handbook of data mining, chapter 7: Performance Alanysis and Evaluation, page 430, Lawrence Erlbaum Assoc Inc, 2003.

[19] Data Mining for Network Intrusion Detection. Dokas, P. , ve diğerleri, NSF Workshop on Next Generation Data Mining, 2002. DAAD19-01-2-0014.

[20] Ghorbani, Ali, Lu, Wei ve Tavallaee, Mahbod . Chapter 2: Detection Approaches. Network Intrusion Detection, Springer, 2009.

[21] Machine Learning Approach for IP-Flow Record Anomaly Detection. Cynthia Wagner, ve diğerleri. Valencia, Spain : Springer, 10th IFIP Networking Conference (NETWORKING), 2011. s. 28-39.

[22] Ertoz, L., ve diğerleri. ch. 2 MINDS - Minnesota Intrusion Detection System. Data Mining - Next Generation Challenges and Future Directions. MIT Press, 2004.

[23] Learning Nonstationary Models of Normal Network Traffic for Detecting Novel Attacks. Chan, K. ve Matthew, V. Edmonton, Alberta, Canada : Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, 2002. s. 376-385. ISBN:1-58113-567-X .

[24] Network Traffic Anomaly Detection Based on Packet Bytes. Matthew V. Mahoney. Melbourne, Florida : Proceedings of the 2003 ACM symposium on Applied computing, 2003. s. 346-350. ISBN:1-58113-624-2.

[25] ADAM: a testbed for exploring the use of data mining in intrusion detection. Daniel Barbará, ve diğerleri. New York, NY, USA : ACM SIGMOD Record, 2001. doi>10.1145/604264.604268.

[26] Vacca, John. Computer and Information Scurity Handbook. pg. 494-495. Morgan Kauffman, 2013.

[27] Ossipov, Andrew , Santos, Omar ve Frahim, Jazib . Cisco ASA: All-in-one Next-Generation Firewall, IPS, and VPN Services. CISCO, 2014. s. XXXIV.

[28] Web Application Firewall. OWASP Projects. [Çevrimiçi] 18 09 2016. [Alıntı Tarihi: 2 12 2017.] https://www.owasp.org/index.php/Web_Application_Firewall.

[29] OWASP ModSecurity Core Rule Set (CRS). OWASP Project . [Çevrimiçi] 19 02 2017. [AlıntıTarihi:12, 2017]https://www.owasp.org/index.php/Category:OWASP_ModSecurity_Core_Rule_Set_Project.

[30] ModSecurity Core Rule Set (CRS) . [Çevrimiçi] OWASP Project, 12 05 2017. [Alıntı Tarihi: 03 12 2017.] https://coreruleset.org/.

[31] Cyber Security Challenges: Designing effcient Intrusion Detection Systems and Antivirus Tools. Mukkamala, Srinivas , Sung, Andrew ve Abraham, Ajith . CRC Press, 2005. s. 125{161.

[32] Holmes, David . The DDoS Threat Spectrum. F5 Networks, Inc., 2012.

[33] Denial of Service Attacks: A Comprehensive Guide to Trends, Techniques, and Technologies. ADC Monthly Web Attacks Analysis. [Çevrimiçi] September 2012. [Alıntı Tarihi: 4 12 2017.] https://www.imperva.com/docs/HII_Denial_of_Service_Attacks-Trends_Techniques_and_Technologies.pdf.

[34] Layer Seven DDoS Attacks. http://resources.infosecinstitute.com. [Çevrimiçi] 24 09 2013. [Alıntı Tarihi: 4 12 2017.] http://resources.infosecinstitute.com/layer-seven-ddos-attacks/#gref.

[35] Network Infrastructure Security Report VI,Q4 2016. Arbor Networks , 2016.

[36] Vrizlynn L. L. Thing, Morris Sloman, and Naranker Dulay. A Survey of Bots Used for Distributed Denial of Service Attacks. International Federation for Information Processing, Volume 232, 2007.

[37] Holmes, David. The DDoS Threat Spectrum. F5 Networks, Inc., 2013.

[38] Chee, W.O. ve Brennan, T. Layer 7 DDoS. OWASP Project, 2010.

[39] Toward developing a systematic approach to generate benchmark datasets for intrusion detection. Ali Shiravi, Hadi Shiravi, Mahbod Tavallaee, Ali A. Ghorbani. 3, ELSEVIER, 2012, Cilt Volume 31. ISSN 0167-4048.

[40] Arlitt, M. and T. Jin. 1998 World Cup Web Site Access Logs. The Internet Traffic Archive. [Çevrimiçi] August 1998. [Alıntı Tarihi: 05 12 2017.] http://www.acm.org/sigcomm/ITA.

[41] Unsupervised Learning. Learn from Stanford Online. [Çevrimiçi] [Alıntı Tarihi: 01, 2018]https://lagunita.stanford.edu/c4x/HumanitiesScience/StatLearning/asset/unsupervised.pdf.

[42] Poole, David , Mackworth, Alan ve Goebel, Randy . Computational Intelligence and Knowledge. [kitap yaz.] David Poole, Alan Mackworth ve Randy Goebel. Computational Intelligence A Logical Approach. New York : Oxford University Press, 1998.

[43] Zamani, Mahdi ve Movahedi, Mahnush . Machine Learning Techniques for Intrusion Detection. [PDF Document], Cornell University Library, 2015.

[44] The use of computational intelligence in intrusion detection systems: A review. Xiaonan Wu , Shelly ve Banzhaf, Wolfgang . Elsevier, Applied Soft Computing 10 (2010), 2010, s. 1–35. https://doi.org/10.1016/j.asoc.2009.06.019.

[45] Beghdad, Rachid. Critical study of neural networks in detecting intrusions. Computers & Security. 2005, Cilt 27, 5-6, s. 168-175.

[46] Tajbakhsh, Arman, Rahmati, Mohammad ve Mirzaei, Abdolreza. Intrusion detection using fuzzy association rules. Applied Soft Computing. 2009, Cilt 9, 2, s. 462-469.

[47] Gomez, J. ve Dasgupta, D. Complete expression trees for evolving fuzzy classifier systems with genetic algorithms and application to network intrusion detection. the 2002 IEEE Workshop on Information Assurance,. 202, Cilt 14, s. pp. 93-110.

[48] Sommer, Robin ve Paxson, Vern . Outside the Closed World: On Using Machine Learning for Network Intrusion Detection. Security and Privacy (SP), 2010 IEEE Symposium on. 2010.

[49] Zisserman, Andrew. Lecture 2: The SVM classifier. Information Engineering In The Department of Engineering Science. [Çevrimiçi] 2015. [Alıntı Tarihi: 29 12 2017.] http://www.robots.ox.ac.uk/~az/lectures/ml/lect2.pdf.

[50] Mitchell, Tom. Decision Tree Learning. Machine Learning. McGraw Hill, 1997, 3.

[51] Wang, Taehyung. Decision Tree Algorithm Decision Tree Algorithm. Data Mining: lecture slides. [Çevrimiçi] 18 02 2011. [Alıntı Tarihi: 03 01 2018.] http://www.csun.edu/~twang/595DM/Slides/Week4.pdf.

[52] MathWorks. Multisignal 1-D Wavelet Analysis. Wavelet Toolbox. [Çevrimiçi] MathWorks.

[53] Lazarevic, A., Kumar, V. ve Srivastava, I. Intrusion Detection: A Survey. MANAGING CYBER THREATS Issues, Approaches, and Challenges. basım yeri bilinmiyor : Springer, 2005.

RESUME

Mohammed Salim was born in Gaza, He finished the primary school at UNRWA schools in Deir Elbalah city in 1999. He continued the high school in the same city at Almanfaluti high school. He graduated from the science-branch in 2002 where he was ranked among the top ten in his class. In 2002 he joined the Department of Computer Engineering. He was interested in classes related to building computer systems including logic design, advanced programming languages and artificial intelligence. He was graduated with distinction in 2007. In 2008 he started working in the Ministry of Finance as a network specialist. In 2010 he moved to work as a professional trainer in many computer fields at different vocational training centers in the Ministry of Labor. In 2014 he was selected by the committee of the Turkish scholarship to continue his MSc studies at Sakarya University within the Turkish scholarship program. Mr. Salim is looking forward to pursuing the Ph.D. in the same discipline in data analysis and object detection using machine learning techniques in addition to his interest in software development.