Detecting Distributed Denial of Service Attacks Using Data Mining Techniques

Mouhammd Alkasassbeh
IT Department
Mutah University
Karak, Jordan

Ahmad B.A Hassanat
IT Department
Mutah University
Karak, Jordan

Ghazi Al-Naymat
King Hussein Faculty of Computing Sciences
Princess Sumaya University for technology
Amman, Jordan

Mohammad Almseidin
IT Department
Mutah University
Karak, Jordan

***Abstract*—Users and organizations find it continuously challenging to deal with distributed denial of service (DDoS) attacks. . The security engineer works to keep a service available at all times by dealing with intruder attacks. The intrusion-detection system (IDS) is one of the solutions to detecting and classifying any anomalous behavior. The IDS system should always be updated with the latest intruder attack deterrents to preserve the confidentiality, integrity and availability of the service. In this paper, a new dataset is collected because there were no common data sets that contain modern DDoS attacks in different network layers, such as (SIDDoS, HTTP Flood). This work incorporates three well-known classification techniques: Multilayer Perceptron (MLP), Naïve Bayes and Random Forest. The experimental results show that MLP achieved the highest accuracy rate (98.63%).**

Keywords—DDoS; IDS; MLP; Naïve Bayes; Random Forest

I. INTRODUCTION

Network security has become of utmost importance in all areas of business and industry, including bank transactions, Email, social media and university eServices, etc. Recently, web and network services have suffered from intruder attacks. Hackers are continually generating new types of Distributed Denial of Service (DDoS) which work on the application layer as well as the network layer. The vulnerabilities in the above mentioned areas allow hackers to deny access to web services and slow down access to network resources.

The IDS system is one of the most common solutions to dealing with the problem of DDoS and preserving the confidentiality, integrity and availability of web services and computer network resources. IDS system uses machine learning techniques to detect and classify types of DDoS in an intelligent way and will eliminate intrusion without referral to the System Security Officer (SSO), however achieving one hundred percent accuracy in detecting and classifying all new types of attacks is hard to achieve. Naïve

Many types of DDoS attacks are already known, such as a Smurf attack, which sends large numbers of Internet controlled message protocol packets to the intended victims. Another type of DDoS is R-U-Dead-Yet (RUDY), which simply consumes all available sessions of a web application which means sessions will never end. In other words, the web service will be unavailable for any new request from new users. One of the most up-to-date DDoS types is HTTP POST/GET, where attackers send a completely legitimate posted messages at a very slow rate, such as (1 byte/240 second), into a web server that is hosting a web application. This type of DDoS will have a harmful effect on a web service and cause it to slow down temporarily and interrupting the service. Another type of modern DDoS attack is an SQL Injection Dos (SIDDoS) where attackers insert a malicious SQL statement as a string that will pass to a website's database as an equation (e.g., through the input values in the website form), then illegally allowing access to the resources or to the stored data on servers.

Unfortunately, most of the common open access data sets have duplicated and redundant instances, which make the detection and classification of the DDoS unrealistic and ineffectual. There are also no avialable data sets (e.g. KDD 99) which include new DDoS types, such as HTTP flood and SIDDOS. In this research, we collected a completely new dataset in a controlled environment, which includes four harmful types of attack namely: UDP flood, Smurf, HTTP Flood and SIDDOS.

Machine learning is used to detect and classify network traffic based on some features (average packet size, inter arrival time, packet size, packet rate, bit rate, etc.) that are used to measure and determine whether the network traffic is normal or is a type of DDoS. DDoS attacks mostly have the same average packet size. The number of packets will increase in the attacked packet rather than the normal packet; also, the inter arrival time will be too small to allow attackers to consume resources rapidly. DDoS packets always have a high bit rate for network layer attack. Attackers focus on any attributes that help them to consume resources and make the service unavailable to end users.

In this work, we make the following contributions:

1) A new dataset has been collected including modern types of attack, which were not been used (to the best of our knowledge) in previous research. The dataset contains 27 *features and five classes. A network simulator (NS2) is used in this work, because NS2 can be used with high confidence due to its capablity of producing valid results that reflect a real environment.*

2) The collected data has been recorded for different types of attack that target the most critical network layers (Application and network). It should be known that this data has never been collected before.

3) Three machine learning algorithms are applied using the collected dataset to classify the DDoS types of attack.

The remainder of the paper is organized as follows: Section II presents the related work and provides a brief discussion of machine learning classifiers in the relevant area. In Section III, we present the architecture of intrusion detection system techniques. Section IV describes the possibility of an attack in the application layer and network layer, followed by details about the dataset collection in Section V. Section VI describes the IDS classifiers used in this paper. In section VII, we describe the evaluation metrics used to assess the performance of the classifiers used; the section also discusses the experiments and the achieved results. Finally, Section VIII conculdes the the paper and decribes the future work.

II. Related Work

Early investigation of IDS was carried out by Georgios Loukas and Glay [1], who began their work with the time-line significant DOS. In September 1996, a SYN Flood attack was discovered, a smurf attack began in January 1998 and a HTTP flood was the modern DOS that began in 2004. Specialists in the area suggest complete protection architecture through detection which can either be anomaly-based or signature based, or a hybrid of these two methods. Classifications such as neural networks, radial basis functions and genetic algorithms are increasingly used in DoS detection because of the automatic classification they can offer. The protection system either drops the attacking packets in a timely fashion or renders them inoperable. Traffic rate, SYN and URG flags, as well as some specific ranges of ports, are the most significant for the identification of a DoS attack, as some investigators have tested in their surveys.

Guiomar et al. [2] have implemented a Network Intrusion Detection System (NIDS) using OPNET simulation. This was based on misuse detection and network traffic was imported using an ACE module into OPNET. A NMAP port scanner was used to simulate a flood attack, and the proposed IDS was tested in a controlled environment; the result was satisfactory with around 93% accuracy rate.

Different techniques have been used for IDS. Chandrika Palagiri showed that a modelling network can achieve a realistic result to demonstrate a Neural Network, especially for an individual attack. They also used a support vector machine (SVMs) as a clustering approach with MLP which is a very useful technique to present a uniform or clustering group. Researchers often focus on a Neural Network that can make decisions quickly and for real-time detection [3].

Sujay Apale, Rupesh Kamblem and others [4] on the layer seven DDoS flooding attacks because such attacks are becoming more severe and are increasing in occurrence. Different machine learning techniques were used for the intrusion detection system. Some of these techniques achieved an acceptable detection rate, while others achieved only a poor detection rate. Moreover, the Naïve Bayes (NB) algorithm enables a faster learning/training speed than other machine learning algorithms, and it has a greater accuracy rate in classification and detection of a layer seven attack.

Kejie et al. [5] proposed a framework to detect DDoS attacks and identify attack packets efficiently. The purpose of the framework is to exploit spatial and temporal correlation of DDoS attack traffic. Such techniques can accurately detect DDoS attacks and identify attack packets without modifying existing IP forwarding mechanisms at the routers. This work achieved 97% for detection probability using the proposed framework.

Further, a hybrid Neural Network technique was used by Wei Pan and Weihua Li, who proposed a hybrid Neural Network consisting of a self-organizing map (SOM) and radial basis functions to detect and classify DDoS attacks. The proposed technique achieved a satisfactory accuracy rate result for detecting and classifying DDoS attacks [6].

Norouzian et al. [7] presented a most effective classification technique for detecting and classifying attacks into two groups normal or threat. They proposed a new approach to IDS based on a MultiLayer Perceptron Neural Network to detect and classify data into 6 groups. They implemented their MLP design with two hidden layers of neurons and achieved 90.78% accuracy rate.

A NIDS using a 2-layered, feed-forward neural network was proposed by Haddadi, F, Sara Khanchi and others [8]. The proposed system classified normal connections and attacks. Different types of attacks were determined, and they focused on using training function, data validation and a preprocess dataset that caused less memory usage, minimum resource consumption and faster training. After implementing the proposed system on a KDD cup 99 dataset, the result was very satisfactory, both on accuracy rate and performance.

Zheng et al. [9] implemented a Hierarchical Intrusion Detection (HIDE) system which can detect attacks using preprocessing statistical values and a Neural Network. They tested five classifiers: Perceptron, Backpropagation (BP), Perceptron-backpropagation-hybrid (PBH), Fuzzy ARTMAP, and Radial-based Function. They stated that BP and PBHachieved the highest accuracy rate for detecting and classifying network attacks.

Reyhaneh Karimazad and Ahmad Faraahi [10] proposed an anomaly-based DDoS detection approach using an analysis of network traffic. A radial-based function (RBF) Neural Network was used in this approach, and they tested their method on a UCLA dataset, achieving 96% accuracy rate for a DDoS attack.

Other work on preventing/ avoiding attacks by means of, for example, fuzzy clustering, genetic algorithm, and artificial neural network (ANN) has been conducted. Ms. Jawale and Prof. Bhusari presented research on ANN that achieved the highest accuracy rate. They proposed a system that uses multilayer perceptrons, back propagation and a support vector machine, consisting of multi modules such as packet collection and preprocessing data. This system achieved 90.78% detection rate [11].

An overview and broad classification of IDS was presented by Mohammed Alenezi and Martin J Reed. The difficulties and characteristics of DoS/DDoS attacks are discussed in this research. Three different classifications were chosen. They focus on general DoS and flooding attacks. The CUSUM approach has many advantages over statistical techniques, which this research effectively demonstrates [12].

Mouhammd Alkasassbeh and Mo Adda [13] showed in their work that mobile agent technology combined with statistical methods based on the Wiener filter can effectively provide the ability to detect network intrusion attempts. Wiener statistical filtering takes advantage of the correlation matrix between the input management information base (MIB) variables and cross correlation with the desired MIB variables to detect abnormal traffic. The MIB variables give a clear image when an abrupt change in network traffic occurs, for example, when a number of PCs start flooding the server with requests. This excessive number of ftp requests stresses the server, which leads to it crashing.

In [14], Dimitris Gorillas and Evangelos Dermatas presented and evaluated a Radial-basis-function (RFB) Neural Network for DDoS attacks dependent on statistical vectors through short-time window analysis. The proposed method was tested and evaluated in a controlled environment with an accuracy rate of 98% of DDoS detection.

III. INTRUSION DETETCTION SYSTEM (IDS)

An intrusion detection system is one of the solutions which can be used to prevent an intruder from launching a DDoS attack in a protected network. An effective IDS can detect a new DDoS in a short time without human intervention. An IDS system can be categorized into two types as follows.

- Host Intrusion Detection System (HIDS): this type of IDS can be implemented in network devices or workstations. HIDS techniques can be used to prevent a DDoS attack on a selected device, but it does not support monitor of a whole network.
- Network Intrusion Detection System (NIDS): this type of IDS can be implemented as a security strategy within a protected network, and can be used to detect and classify all network traffic from all devices. In this research, we will apply an NIDS security strategy in a network [12]. Figure 1 shows a NIDS and HIDS structure.

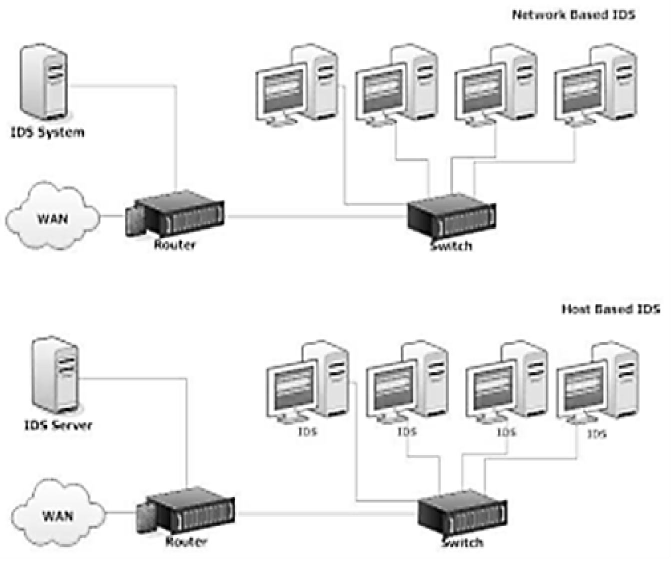

Fig. 1. NIDS and HIDS Structure

IDS can be applied, either anomaly-based or signature based, to detect and classify network traffic. The anomaly based is used to compare network traffic behavior with historical baseline data and so it requires training data to work logically. A signature-based focus is placed on each independent packet, and is then compared with the stored signature or known intruder attack. Detection time for the signature based is faster than the anomaly-based, as the anomaly-based requires training data, while the signature-based requires a stored signature. Figure 2 shows general IDS classification.

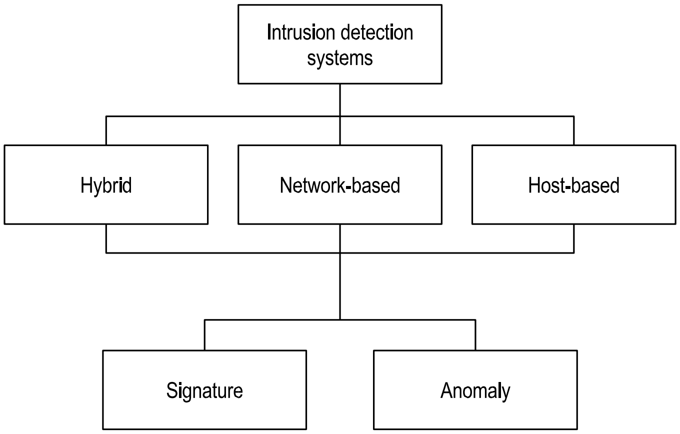

Fig. 2. IDS Classification

IV. Possibilities of Attacks

DDoS attacks; depending on the target protocol, different types of DDoS attack can be implemented on OSI layers; Figure 3 presents the seven layers of OSI. In this paper, we describe the testing of the most recent attacks on the application layer and the network layer.

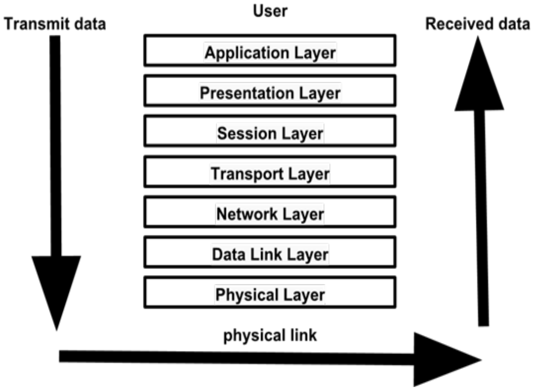

Fig. 3. OSI Seven Layers

A. Network Attack Layer

Attackers implement their DDoS on the victim servers by consuming the servers resources until the service becomes unavailable to all users. A Smurf attack and User Datagram Protocol (UDP) flood attack are part of the network layer attack where an attacker uses malicious (TCP / UDP) network traffic to deny access to the server service. Attackers implement a Smurf attack on malicious network traffic by spoofing IP addresses in a network for sending a large number of Internet Control Message Protocol (ICMP) floods to the victim machine. A large number of ICMPs causes denial of access to the server's services. Figure 4 shows how an attacker implements a Smurf attack on a network.

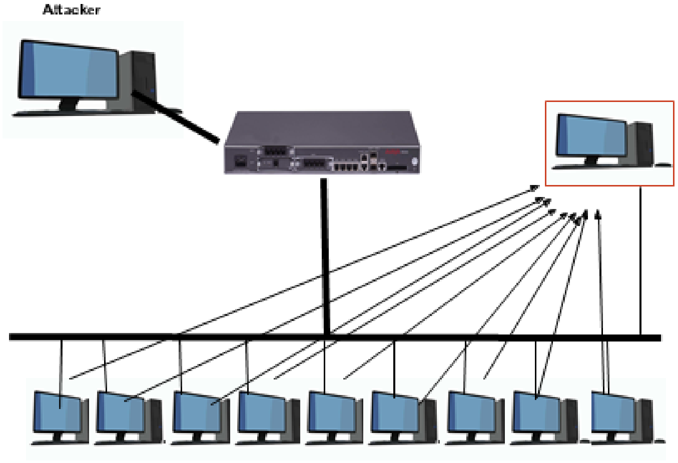

Fig. 4. Smurf Attack Structure

Another type of network layer attack is a UDP flood, where the UDP is a connectionless protocol. An attacker sends a command to the slave machine to use the network workstation as part of the attack. All the workstations send a great amount of UDP traffic to the victim server [15]. Figure 5 indicates how attackers implement a UDP flood on the network.

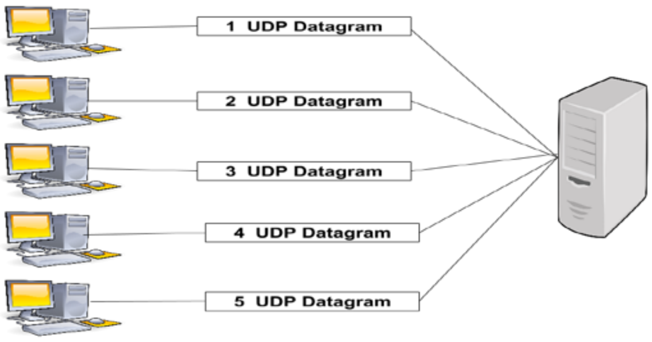

Fig. 5. UDP Flood Structure

B. Application Attack Layer

One of the most common protocols supported by the application layer is HTTP protocol. Any web applications implemented over HTTP are accessible from anywhere, so this makes them difficult to detect and classifying application layer attacks is difficult to prevent. In most situations, intruders send the completely legitimate request for service, and the attackers act as a normal user request service from the server. Figure 6 shows web applications on the World Wide Web.

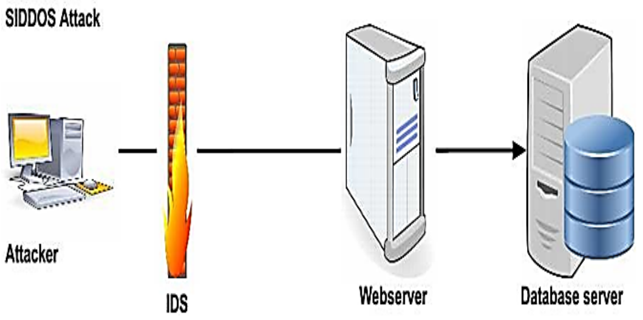

Fig. 6. Web Application Structure

Moreover, most modern DDoS application layer attacks are SQL Injection Distributed Denial of Service (SIDDOS), where Attackers start from the client side, for example the browsers, by inserting a malicious code, and forwarding it to the server side (e.g., input inbox values in a web form) [16]. A SIDDOS attack consumes the servers resources if the malicious code is then forwarded to the servers execution indefinitely. On the other hand, a SIDDOS attack can also be used to steal personal information and make the service unavailable for clients by changing their personal information.

V. Dataset Collection

A new dataset was collected in this work because there is no existing data sets that contain a modern DDoS attack such as (SIDDOS, HTTP Flood), and furthermore, other available data sets may include a great deal of duplicate and redundant records, and that may result in an ultimate unrealistic outcome. Our collected dataset contains four types of DDoS attack as follows: (HTTP Flood, SIDDOS, UDP Flood, and Smurf) without redundant and duplicate records. Table 1 lists number of records of these types of attack. Table 2 shows the dataset features we dealt with.

TABLE I. DISTRIBUTION OF ATTACKS

Attack Name	*No. of Records*
Smurf	12590
UDP Flood	201344
SIDDOS	6665
HTTP Flood	4110

TABLE II. EXTRACTED DATA SET FEATURES

Variable No	*Description*	*Type*
1	SRC ADD	continuous
2	DES ADD	Continuous
3	PKT ID	Continuous
4	FROM NODE	Continuous
5	TO NODE	Continuous
6	PKT TYPE	Continuous
7	PKT SIZE	Continuous
8	FLAGS	Symbolic
9	FID	continuous
10	SEQ NUMBER	continuous
11	NUMBER OF PKT	continuous
12	NUMBER OF BYTE	continuous
13	NODE NAME FROM	Symbolic
14	NODE NAME TO	Symbolic
15	PKT IN	continuous
16	PKTOUT	continuous
17	PKTR	continuous
18	PKT DELAY NODE	continuous
19	PKTRATE	continuous
20	BYTE RATE	continuous
21	PKT AVG SIZE	continuous
22	UTILIZATION	continuous
23	PKT DELAY	continuous
24	PKT SEND TIME	continuous
25	PKT RESEVED TIME	continuous
26	FIRST PKT SENT	continuous
27	LAST PKT RESEVED	continuous

The proposed system that is used to collect a new dataset is organized as illustrated in Figure 7. The selected classifiers were tested in 734,627 records which they were fully randomized to obtain realistic results.

- Collection and auditing: in this step, all network traffic is collected and audited from NIDS.
- Preprocessing file format: redundant and duplicate records are removed.
- Feature extraction: extracts features parameters from the collected network traffic and assigns each feature to the first column; these will be used as a vector in the new dataset.
- Statistical measurements: in this step additional features are calculated using statistical equations.

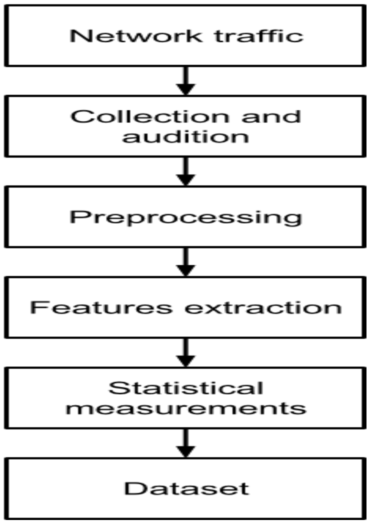

Fig. 7. Dataset Collection Steps

VI. IDS CLASSIFIERS

In this work, we investigated and tested three different classifiers based on the dataset collected and described in the previous section. The models are the MLP, Random Forest and Naïve Bayes classifiers. The models are described as follows.

A. MLP -ANN

The most common and well-known Feedforward Neural Network (FFNN) model is called MLP. MLP has been successfully applied in a number of applications, including regression, classification, or time series prediction problems using simple auto-regressive models [17] [18] [19]. The structure of a simple MLP network is clarified in Figure 8.

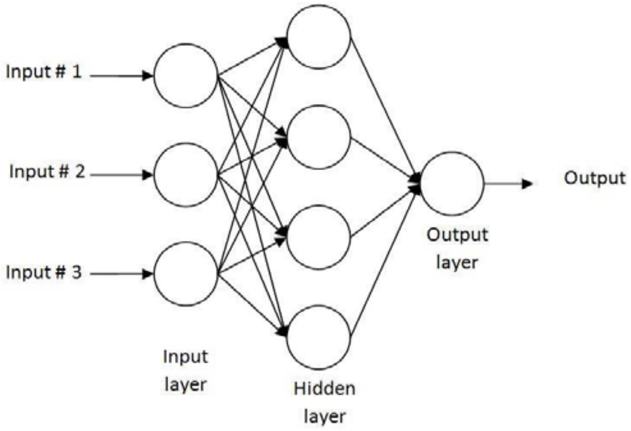

Fig. 8. MLP Structure

MLP permits the data flow to travel one way, from input to output. There is no feedback; it tends to be straight-forward networks that companion inputs with outputs. According to [20] [21], any MLP network can be distinguished by a number of performance characteristics, which can be summarized in three points:

- Neural Network Architecture: Overall, MLP architecture can be clarified as the pattern of connections between the neurons in different layers.

The architecture consists of three layers: input layer, hidden layers, and output layer. Two nodes of each end-to end layer are connected. Furthermore, MLP is always fully connected. Each link has a weight which is limited based on the training algorithm. More complex architectures have more layers.

- Training Algorithm: The method of selecting one model from a set of models, which determines the weights of the connections.
- Transfer Function: Transfer Function is applied by each neuron to its net input to determine its output signal. This function is usually non-linear.

Sigmoid Function is one of the most commonly used transfer functions. The use of the sigmoid function has an advantage in neural networks trained by a backpropagation learning algorithm. The sigmoid function and other common transfer functions are listed in Table 3.

TABLE III. EXAMPLES OF SOME COMMON TRANSFER FUNCTIONS

Transfer Function	***Detention***
Linear	$f(x) = x$
Segmoid	$f(x) = \frac{1}{1 + e^{-x}}$
Hyperbolic	$f(x) = \frac{(e^{x}) - (e^{-x})}{1 + e^{-x}}$
Hard Limit	$f(x) = \begin{pmatrix} 0, & x < 0 \\ 1, & x \geq 0 \end{pmatrix}$
Symmetric Hard Limit	$f(x) = \begin{pmatrix} -1, & x < 0 \\ 1, & x \geq 0 \end{pmatrix}$

In order to understand the algorithm of the learning process on MLP, suppose that a given MLP has N neurons in the input layer and M neurons in the hidden layers, and one output neuron. The learning process can be divided into a number of stages and is described as follows:

- Hidden layer stage: Given a number of inputs ψi and a set of corresponding weights between the input and hidden neurons wij, the outputs of all neurons in the hidden layer are calculated by (1) and (2):

$$O_i = \sum_{i=0}^{n} w_{ij}\, \varphi_i \tag{1}$$

$$y_i = z(O_j) \tag{2}$$

Where i = 1, 2..., N and j = 1, 2, M. z and yj are the activation function and output of the jth node in the hidden layer, respectively. z is usually a sigmoid function given in 3.

$$z(x) = \frac{1}{1+e^{-x}} \tag{3}$$

- Output stage: The outputs of all neurons in the output layer are given in 4. l is defined as the number of neurons in the output layer. For simplicity, l is one.

$$\mathrm{Y}\hat{}\ = \mathrm{f}\left(\mathrm{X} \sum_{j=0}^{m} w_{jl}\, y_j^{H} \right) \tag{4}$$

f0 is the activation function of the output layer which is usually a linear function. Yˆ is the neural network output from the single neuron in the output layer. The MLP network is attempting to minimize the Error via the classical Back-propagation (BP) Training algorithm. BP learning starts with all weights initialized randomly, then weights are modified as the algorithm progresses until steady state values are reached.

- Error validation stage: ANN continues the learning process until the error minimization criteria are reached. Assuming that the desired output is Y and the produced ANN output is Yˆ, the learning process should stop when the error difference given in Equation 5 is minimal. T is the total number of observations used to build the ANN model during training. Another set of data must be used to validate the developed ANN model performance.

$$Error = \frac{1}{T} \sum_{i=1}^{T} (Y_i - {Y^a}_i)^2 \tag{5}$$

In MLP, all the network weights and bias values are assigned with random values initially, and the goal of the training is to find the set of network weights that cause the output of the network to match the real values as closely as possible. However, we cannot forget that the MLP always takes the longest time for training [8].

B. *Random Forest*

Random forest classifiers were developed by LEO Breiman and Adele Cutler. They combine tree classifiers to predict new unlabeled data, the predictor depends on the number of as that are represented by the number of trees in the forest, the attributes are selected randomly, each number of trees represents a single forest and each forest represents a predation class for new unlabeled data [4]. In this algorithm, random features selection will be selected for each individual tree. A random forest classifier ensemble learning algorithm is used for classification and prediction of the outputs based on an individual number of trees [22]. Using random forest classifiers, many classification trees will be generated, and each individual tree is constructed by a different part of the general dataset. After each tree is classified in an unlabeled class, a new object will be implemented under each tree vote for decision. The forest chosen as the winner is based on the highest number of votes recorded. Figure 9 shows decision forest architecture and how the number of votes is calculated. Random forest algorithms:

- If you have a dataset you need to split n samples from the whole dataset, so that now we have (n samples= number of trees).
- Each dataset sample needs to be regressed or classified for each record randomly split among all predictor classes to reach an approximately optimal split. bagging can be learned as a special scenario when m(tries) = P (number of predictors).
- Predict unlabeled class based on a reassembled number of aggregation predictions re number of trees.

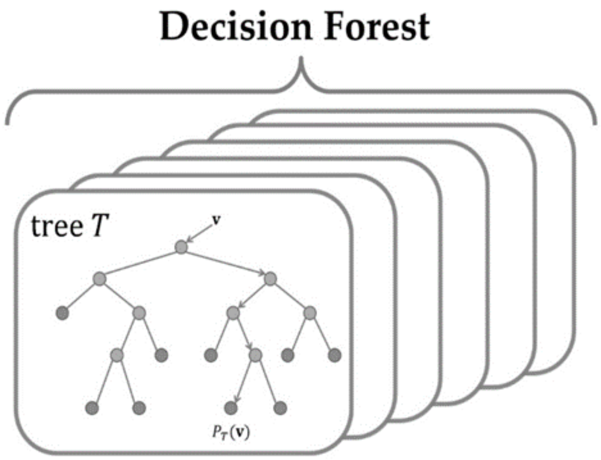

Fig. 9. Decision Forest Architecture

Random Forest Tuning parameters The accuracy rate and error rate for Random Forest (RF) classifiers can be measured by splitting a whole dataset for testing, e.g. (40%) and for training, e.g. (60%). After the random forest, a model test (40%) can be used to calculate the error rate, and the accuracy rate can be measured based on comparing correctly classified instances with incorrectly classified instances. Out of bag (OOB) is another way of calculating the error rate. In this technique, there is no need to split the dataset because calculation occurs in the training phase. The following parameters need to be adjusted correctly in order to achieve the highest accuracy rate with minimum error rate:

- Number of trees.
- Number of descriptors that occurs randomly for present candidates m(tries).

After analysis and studying many cases, 500 trees are needed within the descriptor. Even if there is a great number of trees that will not achieve the highest accuracy rate and will only waste training time and resources [23], so that random forest tuning parameters are a vital research area that needs to be fine-tuned.

C. *Naïve Bayes*

Naïve Bayes [24] is a simple probabilistic classifier that returns p(y|x), Naïve Bayes calculates probabilities for each class in a dataset and determines discriminative learning to predict values of the new class. The main formulation for Naïve Bayes may be found in [9].A Naïve classifier link the dataset attributes x ∈ X that are used as inputs to the class labels Z ∈ {1,2,,,C}, where X is attributes space and Z is a class space. Let X = IRD where D is a real number. Naïve classifier may be used with discrete and continuous attributes. This model is called a multi-label problem. The learning function that directly computes the class is called the discriminates model, the main purpose of which is to learn the conditional class that is used for nonlinear and multi-label problems. We will use the 6,7,8,9,10:

$$p(y|x) = \frac{p(x,y)}{p(x)} = \frac{p\left(\frac{x}{y}\right)p(y)}{\sum_{y\prime=1}^{c} \frac{p(x/y\prime)}{p(y')}} \qquad (6)$$

Naïve classifier achieve outputs based on argument max function as follows:

$$f(x) = y'(x) = arg\{\ (max)\left(p\left(\frac{y}{x}\right)\right)\} \qquad (7)$$

Probabilistic classifiers have the following advantages:

- Option to reject: this is used when we are uncertain of the prediction result and so the prediction result can be ignored since human effort is present.
- Allow learn function to be changed: combinations of probability function can be used to achieve the best performance of the main issues if we use the direct learning function $p(\frac{y}{x})$ and the probability function is changed, so there is no need to re calculate p(xy).
- Balanced classes : some parts of the collected dataset have unbalanced classes which means that if we have one million records of normal network traffic where there is only 1 abnormal for 1000 records, that means we can directly train an unbalanced training dataset and easily achieve 99.9% accuracy rate by just using class always = normal. To deal with this problem, balanced classes are used.

$$pbal((y|x)\alpha P(x,y)Pbal(y) \qquad (8)$$

$$\mathrm{P\ true(yx)} \propto \mathrm{P\ true(x,y)P\ true(y)} \propto \frac{\mathrm{p}\left(\frac{\mathrm{y}}{\mathrm{x}}\right)}{\mathrm{P\ bal(y)}}\ \mathrm{P\ true(y)} \qquad (9)$$

- Models combinations it is very useful when the collected dataset contains a mix of feature types, such as if there is a collected dataset and each feature vector represents distinguished data types (text, images, numbers, etc.) Two or more kinds of attributes using model combinations means that we can build two or more classifiers, such as and so on [25]. To combine two different information sources the following formula is used:

$$P(x1\ , x2\,\middle|\, xy) = P(x1/y)\ \ P(x2/y) \qquad (10)$$

VII. Experiments and Results

In this work, the experiments were performed on Ubuntu 13.10 platform, Intel, Xeon (R) CPU E5-2680 @ 2.70 GHZ x 4, 12.0 GB RAM. A machine learning tool called WEKA version 3.7.12 was used for the application of the classification techniques.. To measure the efficiency of the algorithms, each algorithm was trained on our dataset using 66% of the collected data and the 34% were used as a test data. In fact, we tryied the ten-fold validation but our previous partitioning worked better.

A. *Evaluation Metrics*

To evaluate the performance of the classifiers applied in this work, we use primary performance indicators based on confusion matrix shown in Table 4. This matrix contains information about real and predicted classifications carried out by the classification models. Performance metrics of such systems are commonly evaluated using the information in the matrix [26]. We use MLP, Random Forest and Naïve Bayes classifiers based on the dataset collected in this study. MLP parameters are tuned as listed in Table 5. While for Random forest the ideal number of trees is empirically set to 50.

TABLE IV. CONFUSION MATRIX

		Predicted	
		Positive	***Negative***
True	***Positive***	TP	FN
	Negative	FP	TN

TABLE V. MLP SETTINGS

Parameter	***value***
Number of neurons in hidden layer	16
Learning rate	0.3
Momentum	0.2
Maximum number of epochs	500

- Accuracy - measures the rate of the correctly classified attack instances of both classes.

$$Accuracy = \frac{TP+TN}{TP+FN+FP+TN} \qquad (11)$$

- Precision: It is the ratio of the number of relevant attacks retrieved to the total number of irrelevant and relevant attacks retrieved. It is also called positive predictive, which can be calculated by the following equation.

$$Precision = \frac{TP}{TP+FP} \qquad (12)$$

- Recall: It is the ratio of the number of relevant attacks retrieved to the total number of relevant attacks. It is also called positive sensitivity value, which can be calculated by the following equation.

$$Recall = \frac{TP}{TP+FN} \qquad (13)$$

B. *Result Discussion*

The classifiers were evaluated and assessed using the confusion matrix based on the evaluation metrics listed in section VII (A). The resultant confusion matrices for MLP, Random Forest and Naïve Bayes are shown in Tables 6, 7 and 8, respectively. From these confusion matrices we calculated the accuracy, precision and recall of the models. The overall accuracy was 98.63%, 98.02% and 96.91% for MLP, Random Forest and Naïve Bayes, correspondingly.

However, taking into consideration only the accuracy rate is not sufficient, especially when the data are imbalanced as in our case, where the number of instances in the normal class was much higher than the other classes. Therefore, the precision and recall were calculated for each class: Normal, UDP-Flood, Smurf, SIDDOS and the HTTP-FLOOD.

Figures 10 and 11 depict a comparison between the obtained precision and recall values of the developed models for each class. According to the Figures, it can be noted that all classifiers achieved high precision and recall rates for the normal class. However, their performance varies regarding the other four classes of attacks.

MLP achieved high precision and recall results for the minority classes, while Naïve Bayes was the worst. It can be noted also that the Smurf class was the most challenging for all classifiers. This is due to the nature of Smurf type of attack, which aims to send large number of ICMP echo packet requests that are hard to be classified as normal or abnormal traffic. It should be noted that ICMP complements the missing flow control and traffic management of IP4. As a result, Random Forest and Naïve Bayes failed to show good rates for the Smurf class, while, on the other hand, MLP achieved a high precision rate. In general, it can be concluded that MLP is the best classifier for detecting DDoS with promising performance results.

TABLE VI. CONFUSION MATRIX FOR MLP

	Normal	***UDP-Flood***	***Smurf***	***SIDDOS***	***HTTP-FLOOD***
Normal	657961	0	0	70	20
UDP-Flood	6767	61765	0	0	0
Smurf	2817	0	1396	132	10
SIDDOS	115	0	0	2136	0
HTTP-FLOOD	0	0	0	86	1352

TABLE VII. CONFUSION MATRIX FOR RANDOM FOREST

	Normal	***UDP-Flood***	***Smurf***	***SIDDOS***	***HTTP-FLOOD***
Normal	653545	3117	1258	112	19
UDP-Flood	6728	61794	10	0	0
Smurf	2798	10	1414	121	12
SIDDOS	160	0	75	1972	44
HTTP-FLOOD	8	0	11	77	1342

TABLE VIII. CONFUSION MATRIX OF NAÏVE BAYES

	Normal	***UDP-Flood***	***Smurf***	***SIDDOS***	***HTTP-FLOOD***
Normal	646612	0	10375	981	83
UDP-Flood	6705	61765	59	3	0
Smurf	2717	0	92	140	1406
SIDDOS	115	0	16	2120	0
HTTP-FLOOD	0	0	0	86	1352

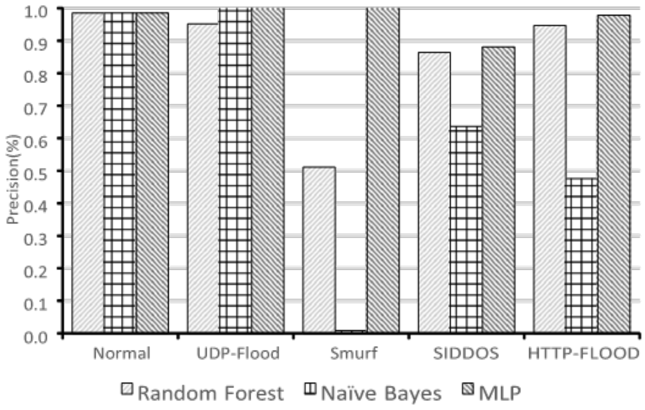

Fig. 10. Precision Results

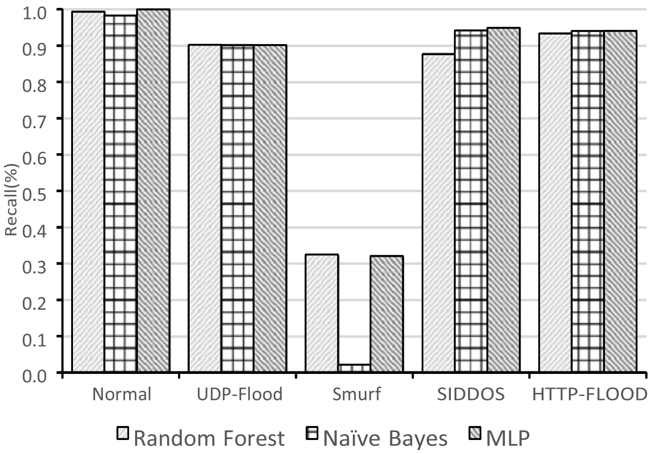

Fig. 11. Recall Results

VIII. Conclusions

In this paper, we collected a new dataset that includes modern types of attack, which were not been used in previous research. The dataset contains 27 features and five classes. A network simulator (NS2) was used in this work, because NS2 can be used with high confidence due to its capablity of producing valid results that reflect a real environment. The collected data has been recorded for different types of attack that target the Application and network layers. Three machine learning algorithms (MLP, Random Forest, and Naïve Bayes) were applied on the collected dataset to classify the DDoS types of attack namely: Smurf, UDP-Flood, HTTP-Flood and SIDDOS. The MLP classifier achieved the highest accuracy rate.

The future work is to include more types of modern attacks in different OSI layers, such as the transport layer. In addition, we will examin different features selection techniques to choose the appoppriate features.

References

[1] G. Loukas and G. Oke,¨ "Protection against denial of service attacks: a survey," The Computer Journal, p. bxp078, 2009.

[2] G. Corral, A. Zaballos, J. Abella, and C. Morales, "Building an ids using opnet," OPNETWORK2005, 2005.

[3] A. Bivens, C. Palagiri, R. Smith, B. Szymanski, M. Embrechts, et al., "Network-based intrusion detection using neural networks," Intelligent Engineering Systems through Artificial Neural Networks, vol. 12, no. 1 , pp. 579–584, 2002.

[4] S. Apale, R. Kamble, M. Ghodekar, H. Nemade, and R. Waghmode, "Defense mechanism for ddos attack through machine learning,"

[5] K. Lu, D. Wu, J. Fan, S. Todorovic, and A. Nucci, "Robust and efficient detection of ddos attacks for large-scale internet," Computer Networks, vol. 51, no. 18, pp. 5036–5056, 2007.

[6] W. Pan and W. Li, "A hybrid neural network approach to the classification of novel attacks for intrusion detection," in Parallel and Distributed Processing and Applications, pp. 564–575, Springer, 2005.

[7] M. R. Norouzian and S. Merati, "Classifying attacks in a network intrusion detection system based on artificial neural networks," in Advanced Communication Technology (ICACT), 2011 13th International Conference on, pp. 868–873, IEEE, 2011.

[8] F. Haddadi, S. Khanchi, M. Shetabi, and V. Derhami, "Intrusion detection and attack classification using feed-forward neural network," in Computer and Network Technology (ICCNT), 2010 Second International Conference on, pp. 262–266, IEEE, 2010.

[9] J. Jorgenson, C. Manikopoulos, J. Li, and Z. Zhang, "A hierarchical anomaly network intrusion detection system using neural network classification," in Proceedings of the 2001 IEEE Workshop on Information Assurance and Security, 2001.

[10] R. Karimazad and A. Faraahi, "An anomaly-based method for ddos attacks detection using rbf neural networks," in 2011 International Conference on Network and Electronics Engineering, IPCSIT, vol. 11 , 2011.

[11] M. D. R. Jawale and V. Bhusari, "Technique to detect and classify attacks in nids using ann," 2014.

[12] M. Alenezi and M. J. Reed, “Methodologies for detecting dos/ddos attacks against network servers,” in The Seventh International Conference on Systems and Networks Communications, ICSNC Semi-Markov models, 2012.

[13] M. Al-Kasassbeh and M. Adda, “Network fault detection with wiener filter-based agent,” Journal of Network and Computer Applications, vol. 32, no. 4, pp. 824–833, 2009.

[14] D. Gavrilis and E. Dermatas, “Real-time detection of distributed denialof-service attacks using rbf networks and statistical features,” Computer Networks, vol. 48, no. 2, pp. 235–245, 2005.

[15] S. Chithra and E. G. D. P. Raj, “Overview of ddos algorithms: A survey,” 2013.

[16] H. Shahriar, S. North, and W. Chen, “Early detection of sql injection attacks,” International Journal of Network Security & Its Applications (IJNSA), vol. 5, no. 4, pp. 53–65, 2013.

[17] M. Alkasassbeh, A. F. Sheta, H. Faris, and H. Turabieh, “Prediction of pm10 and tsp air pollution parameters using artificial neural network autoregressive, external input models: A case study in salt, jordan,” Middle-East Journal of Scientific Research, vol. 14, no. 7, pp. 999 – 1009, 2013.

[18] H. Faris, M. Alkasasbeh, and A. Rodan, “Artificial neural networks for surface ozone prediction: models and analysis,” Polish Journal of Environmental Studies, vol. 23, no. 2, 2014.

[19] O. Adwan, H. Faris, K. Jaradat, O. Harfoushi, and N. Ghatasheh, “Predicting customer churn in telecom industry using multilayer preceptron neural networks: Modeling and analysis,” Life Science Journal, vol. 11 , no. 3, pp. 75–81, 2014.

[20] B. Krse and P. van der Smagt, An Introduction to Neural Networks. CRC Press, The University of Amsterdam.

[21] L. V. Fausett, Fundamentals of Neural Networks: Architectures, Algorithms, and Applications. Prentice Hall, 1994.

[22] A. ARAAR and R. BOUSLAMA, “A comparative study of classification models for detection in ip networks intrusions.” Journal of Theoretical & Applied Information Technology, vol. 64, no. 1, 2014.

[23] M. A. M. Hasan, M. Nasser, B. Pal, and S. Ahmad, “Support vector machine and random forest modeling for intrusion detection system (ids),” Journal of Intelligent Learning Systems and Applications, vol. 2014 , 2014.

[24] K.-S. Fu, Pattern Recognition and Machine Learning: Proceedings of the JapanUS Seminar on the Learning Process in Control Systems, held in Nagoya, Japan August 18–20, 1970. Springer Science & Business Media, 2012.

[25] K. P. Murphy, “Naive bayes classifiers,” University of British Columbia, 2006.

[26] V. M. Patro and M. R. Patra, “Augmenting weighted average with confusion matrix to enhance classification accuracy,” Transactions on Machine Learning and Artificial Intelligence, vol. 2, no. 4, pp. 77 –91, 2014.