PAYLOAD BASED INTERNET WORM DETECTION USING NEURAL NETWORK CLASSIFIER

A.Tharani MSc (CS)
M.Phil. Research Scholar Full Time

B.Leelavathi, MCA, MPhil.,
Assistant professor,
Dept. of information technology,
Sri Jayendra Saraswathy Maha Vidyalaya CAS

ABSTARCT

With the capability of infecting hundreds of thousands of hosts, worms represent a major threat to the Internet. The detection against Internet worms is largely an open problem. Internet worms pose a serious threat to computer security. Traditional approaches using signatures to detect worms pose little danger to the zero day attacks. The focus of this research is shifting from using signature patterns to identifying the malicious behavior displayed by the Internet worms. This paper presents a novel idea of extracting flow level features that can identify worms from clean programs using data mining technique such as neural network classifier. Our approach showed 97.90% detection rate on Internet worms whose data was not used in the model building process.

Keywords: Internet Worm Detection, Flow features, neural network, Novel detection.

1. INTRODUCTION

As computer and communication networks become prevalent, the Internet has been a battlefield for attackers and defenders. One of the most powerful weapons for attackers is the Internet worm [1]. Specifically, a worm attacks vulnerable computer systems and employs self-propagating methods to flood the Internet rapidly. As a result, worms, such as Code Red, Slammer, and Witty, have infected hundreds of thousands of hosts and become a significant threat to network security and management. Moreover, the attacking methods generated by worms' designers have become increasingly sophisticated, which poses considerable challenges to defenders.

The Internet is persistently threatened by many types of attacks such as viruses, and worms. A worm is a self-propagating program that infects other hosts based on a known vulnerability in network hosts. In contrast, a virus is a piece of code attached to another executable program, which requires human action to propagate. A major challenge in networking is how to detect new worms and viruses in the early stages of propagation in a computationally efficient manner [2]. During the past 20 years, thousands of different worms have been developed. Some of these worms have caused huge disruption to global networks. The most notable worms include Morris, Code Red and Code Red II, Nimda, Slapper, and Sapphire/Slammer worms, and recently, So-Big.F, MSBlast, and Mydoom. From the first worm that was released in 1988 (the Morris worm), the area of Internet worm detection has been a significant research problem. In order to understand the worm threat, it is necessary to understand the various types of worms, payloads, and attackers.

A network worm is defined as a process that can cause a (possibly evolved) copy of it to execute on a remote computational machine [3]. Worms normally self-propagate across networks by exploiting security or policy flaws in widely-used network services. Worms are different from Viruses in that Viruses piggy-back on files and therefore require user action to enable their propagation. Because of this, viruses propagate at a slower rate than worms. Worms on the other hand, spread extremely fast. During the Code Red I version 1 internet worm attack of the year 2001, over 359,000 computers were infected in under 14 hours [4]. During the more aggressive Slammer internet worm attack of the year 2003 more than 90% of 75,000 vulnerable hosts were infected in less than 10 minutes [5]. A properly constructed worm could infect vulnerable systems in the Internet

at an even greater speed [6]. Worms present a significant threat to the dependability of networking infrastructure. Defending against them in an automated fashion is a challenging task, and has sparked much interest in the research community.

The rest of the paper is organized as follows. Section 2 discusses computer worm behavior. Section 3 discusses various worm detection techniques, indicating the worm characteristics that they leverage for the detection and also points out their deficiencies. Finally Section 4 summarizes the gap that exists in the worm detection space.

2. RELATED WORKS

Internet worms infect the network through illegal traffic flow. Monitoring and detecting the malicious traffic behavior provides better and faster communication. Rather than payload Inspection, traffic flow monitoring detects the network traffic and exploits the internet worms illegal traffic. Various techniques proposed for Internet worm detection are listed below:

Chen et al. [7] proposed a novel approach, that it diminished the future internet epidemics using effective technique of Divide-conquer-scanning worm. This technique is faster and stealthier than the random-scanning worm. In this paper author also described two defense mechanism, they are infected host removal and active honey nets. Kong et al. [8] introduced a novel method for detecting the network based worm. It first generates the signatures automatically by Semantics Aware statistical algorithm. This is used to remove the non-critical bytes, which is combined with a hidden Markov model to automatically generate worm signatures.

In another data mining approach, [9] used three different types of features and a variety of classifiers to detect malicious programs. Their primary dataset contained 3265 malicious and 1001 clean programs. They applied RIPPER (a rule based system) to the DLL dataset. Strings data was used to fit a Naive Bayes classifier while n-grams were used to train a Multi-Naive Bayes classifier with a voting strategy. No n-gram reduction algorithm was reported to be used. Instead data set partitioning was used and 6 Naive-Bayes classifiers were trained on each partition of the data. They used different features to built different classifiers that do not pose a fair comparison among the classifiers. Naive-Bayes using strings gave the best accuracy in their model.

Li et al. [10] Analyzed to find the vulnerable host. Here the author implemented the gradual hybrid anti-worm. This approach was combination of active and passive anti-worm. The work done by the active anti-worm was detecting the vulnerable host on the network and patches them up. Listening process was handled by passive anti-worm, that it attacks the worm from the host after patching it for the process. Wang et al. [11] proposed the approach to analyze the internet worm infection family tree and it is named as worm tree. Through mathematical analysis, captures the key characteristics of the internet worm detection and applying it for bot detection.

Yao et al. [12] Implemented an approach based on time delay to reduce the network worm and also decrease the economic loss rate. In this paper, one critical value is derived. If the time delay is greater than the critical value, then the worm will be eliminated from the network. Zaki et al. [13] Introduced WSRMAS; an anti-worm system. This approach effectively reduces the spreading of the infected worms in network routers and consists of a multi-agent system that can limit or even stop the worm spreading.

From the above related works, different methods have been proposed to detect the Internet worms infecting the network. From the observations, it is found that they detect through monitoring payload and traffic misbehaviors. Payload detection lacks detection of worms when they are encrypted. Monitoring traffic behavior detects only after their spread. To overcome the above limitations, the proposed approach detects the Internet worms by monitoring the traffic flow information.

3. DETECTION SYSTEM

The proposed approach finds out the malicious internet worm flow traffic payload based on the characteristics of network flow payload using neural network classification algorithm. To classify the Internet worms, TCP and UDP flows are examined, they are split into time windows and attributed vector is extracted. Based on the attribute vectors malicious and non-malicious traffic is detected and classified. Figure 1 below shows the complete process of detecting Internet worms through their traffic flows.

3.1 Flow Traffic

Network traffic refers to the amount of data moving across a network at a given point of time. Network data is mostly encapsulated in network packets, which provide the load in the network. Network traffic is the main component for network traffic measurement, network traffic control and simulation. The proper organization of network traffic helps in ensuring the quality of service in a given network. Network traffic is also known as data traffic.

Flows offer an aggregated view of network traffic, by reporting on the amount of packets and bytes exchanged over the network. Therefore, flows drastically reduce the amount of data to be analyzed. A flow is defined as a set of IP packets passing an observation point in the network during a certain time interval. All packets belonging to a particular flow have a set of common properties.

A flow can be defined using the following parameters (Source IP Address, Destination IP Address, Source Port, Destination Port, Protocol)

3.2 Feature Extraction

We use KDD CUP99 data set from MIT [14] and CAIDA dataset [15] for our experimental study, which is one of the world recognized off-line worm intrusion detection data set. This data set contains three sub data set, which are the whole data set, ten percent data set and the test set with correct labels named correct.gz. Specially, we sample 1% and 2% data set from the ten percent KDD CPU99 data set respectively in our experiments, which contains 49402 and 98804 samples in corresponding. There are 41 features in each sample as shown in table 1. The attack can mainly divided into the following four categories.

(1) DoS represents denial of service attack. The attackers make the memory of the computer too busy and cannot handle legitimate requests or refuse to legitimate user's access to the machine.

(2) U2R represents illegal access to the local super user002E the attackers access the root permissions using a loophole through a user without permissions or lower permissions, then login and make illegal operations using root.

(3) R2L represents remote user attack. The attackers remote login the computer, then use the account and password to access to the computer and make illegal operations.

(4) Probe represents the port scanning and vulnerability scanning. The attackers detect and search the computers and ports, and collect all kinds of information and system vulnerabilities, then use these information to attack the target.

Each sample of the data set is a vector, which is collected network connection data from the simulated invasions. The last feature of a sample is the label which denotes whether the sample is normal, and the other 41 feature are listed in the below table.

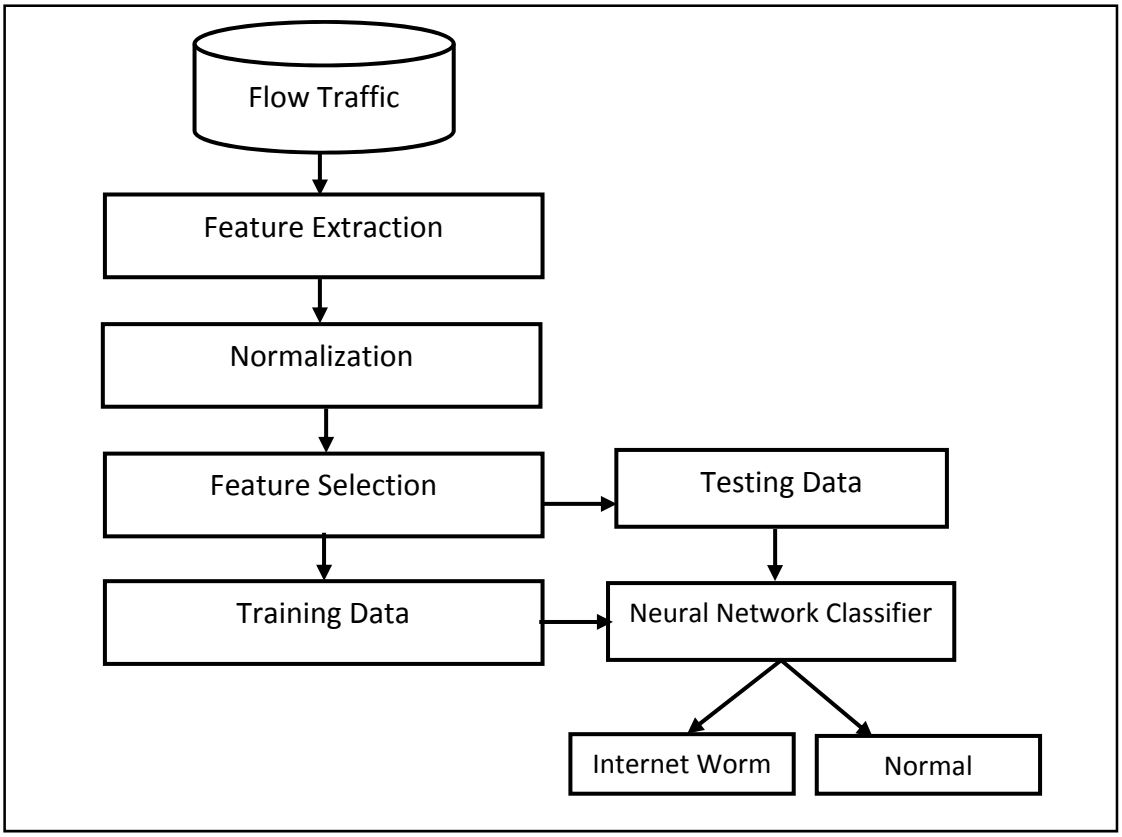

Figure 1. Proposed data mining framework for internet worm Detection

S.No	Features Name	S.No	Features Name
1	Duration	22	Is_guest_login
2	Protocol_type	23	Count
3	Service	24	Serror_rate
4	Src_bytes	25	Rerror_rate
5	Dst_bytes	26	Same_srv_rate
6	Flag	27	Diff_srv_rate
7	Land	28	Srv_count
8	Wrong_fragment	29	Srv_serror_rate
9	Urgent	30	Srv_rerror_rate
10	Hot	31	Srv_diff_host_rate
11	Num_failed_logins	32	Dst_host_count
12	Logged_in	33	Dst_host_srv_count
13	Num_compromised	34	Dst_host_same_srv_rate
14	Root_shell	35	Dst_host_diff_srv_rate
15	Su_attempted	36	Dst_host_same_src_port_rate
16	Num_root	37	Dst_host_diff_src_port_rate
17	Num_file_creations	38	Dst_host_serror_rate
18	Num_shells	39	Dst_host_rerror_rate
19	Num_access_files	40	Dst_host_rerrror_rate
20	Num_outboundcmds	41	Dst_host_srv_rerror_rate
21	Is_hot_login		

Table 1. Features of Internet worm

3.3 Normalization

During training of the neural network, higher valued input variables may tend to suppress the influence of smaller ones. Also, if the raw data is directly applied to the network, there is a risk of the simulated neurons reaching the saturated conditions. If the neurons get saturated, then the changes in the input value will produce a very small change or no change in the output value. This affects the network training to a great extent. To minimize the effects of magnitudes among inputs as well as to prevent saturation of the neuron activation function, the input data is normalized before being presented to the neural network. One way to normalize a feature *x* is using min-max normalization. Min-max normalization performs a linear transformation on the original data values and it preserves the relationship among the original data values. So normalization is done by using the following formula:

$$x' = \frac{x - \min_x}{\max_x - \min_x}$$

Where x' is the normalized value, $\min_x$, and $\max_x$ are the minimum and maximum values of the features respectively.

3.4 Feature Selection

Feature selection and ranking are very crucial for worm detection. Feature selection is the process of obtaining the score for each potential feature and then obtaining the excellent 'k' features. Scoring is done by counting the frequency of a feature in training positive and negative class samples separately and then obtaining a function of both. There are many features that have to be monitored for worm detection out of which certain features will be useful and others may be useless. The removal of useless features enhances the accuracy and decreases the computation time thereby achieving higher performance. The chi-square feature selection metric is used in our research.

A Chi-Square [16] approach is a simple and general algorithm that can automatically select a proper χ 2 value, statistic to discretize numeric features repeatedly until some inconsistencies are found in the data, and achieves feature selection via discretization. The χ^2 method evaluates features individually by measuring their chi-squared statistic with respect to the classes. For a numeric attribute, the method requires its range to be discretized into several intervals. The χ^2 value of an attribute is defined as:

$$\chi^2 = \Sigma_{i=1}^{m} \Sigma_{j=1}^{n} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$

Where, m is the number of intervals, n is the number of classes,

O_{ij} is the number of samples in the i^{th} intervals of j^{th} class,

$E_{ij} = \frac{R_i * C_j}{N}$, is the expected frequency of O_{ij}

R_i the number of samples in the i^{th} interval.

Cj is the number of samples in the jth class, N is the total number of samples.

Degrees of freedom of Chi square Test is (m – 1) (n – 1).

Finally we have selected the top 15 features which are listed in Table 2 along with the features description. When the number of features increases, the classifiers need to process additional data which leads to higher processing power consumption as well as longer time.

S.No	Feature Name	Description
1	source bytes	number of data bytes from source to destination
2	Num-root	Number of 'root' accesses
3	Wrong-fragment	number of 'wrong' fragments
4	Srv-rerror-rate	% of connections that have 'REJ' errors
5	Urgent	number of urgent packets
6	Rerror-rate	% of connections that have ``REJ'' errors
7	Root-shell	1 if root shell is obtained; 0 otherwise
8	Logged-in	1 if successfully logged in ; 0 otherwise
9	Protocol-type	type of the protocol, e.g. tcp, udp, etc.
10	Su-attempted	1 if 'su root' command attempted; 0 otherwise
11	Num-file-creations	Number of file creation operations
12	Dst-host-serror-rate	% of connections that have 'SYN' errors
13	Dst-host-srv-count	No. of connections to the same service as the current connection in the past two seconds
14	Dst-host-count	No. of connections to same host as the current connection in the past two seconds
15	Service	network service on the destination, e.g., http, telnet, etc.

Table 2. Selected Significant features

3.5 Neural Network Classifier

A neural network consists of units (neurons), arranged in layers, which convert an input vector into some output. Each unit takes an input, applies a (often nonlinear) function to it and then passes the output on to the next layer. Generally the networks are defined to be feed-forward: a unit feeds its output to all the units on the next layer, but there is no feedback to the previous layer. Weightings are applied to the signals passing from one unit to another, and it is these weightings which are tuned in the training phase to adapt a neural network to the particular problem at hand. This is the learning phase.

The Multilayer feed forward neural networks [17, 18, & 19] are appropriate for solving problems where all the information can be presented to the neural network at once. In the training phase, a training set is presented as input to the neural network which iteratively adjusts network weights and biases in order to produce an output that matches, within a certain degree of accuracy, a previously known result. In the testing phase, a new input is presented to the network and a result is obtained based on the network parameters that were calculated during the training phase. In this work, the network is trained with back propagation learning algorithm, which is an appropriate learning algorithm for training multilayer feed forward networks for vector classification. The input layer has 6 neurons corresponding to the dimensionality of the input vectors, and the output layer has two neurons. The number of neurons in the hidden layer is empirically selected such that the performance function, which is the mean square error for feed forward neural network is minimized

4. EXPERIMENTAL RESULTS AND DISCUSSIONS

All the experiments are conducted using NSL-KDD dataset and CAIDA dataset that has 60438 training instances, 22544 instances for testing with select most prominent 15 attributes and random forest classification to build an efficient internet worm detection system. We have evaluated our classifier with various evaluation measures, such as accuracy, F-measure and false positive rate.

Accuracy is percentage of correctly identified Internet worms.

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN}$$

True Positive (TP) =Number of samples correctly predicted as worm.
False Positive (FP) = Number of samples incorrectly predicted as worm.
True Negative (TN) = Number of samples correctly predicted as normal.
False Negative (FN) = Number of samples incorrectly predicted as normal.

Precision is a measure of what fraction of test data is detected as worm are actually from the worm classes.

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

Recall measures the fraction of worm class that was correctly detected.

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

F-Measure is a measure of test's accuracy, which measures the balance between precision and recall.

F-Measure = $\frac{2 * Precision * Recall}{Precision + Recall}$

False Positive Rate (FPR) is percentage of wrongly identified normal classes.

Positive Rate (FPR) = $\frac{FP}{FP+TN}$

The experimental results are given in table 3.

Measures	**Values**
Precision	0.954
Recall	0.980
F-Measure	0.979
Accuracy	97.90
False Positive Rate	0.057

Table 3. Experimental results of neural network Classifier

From the table, one can observe that the random forest classifier achieves highest accuracy of 97.90% and false positive rate of 0.057 when detecting the internet worms.

For the better visualization the tabulated values of precision, recall and f-measure are represented in graphical format in figure 2.

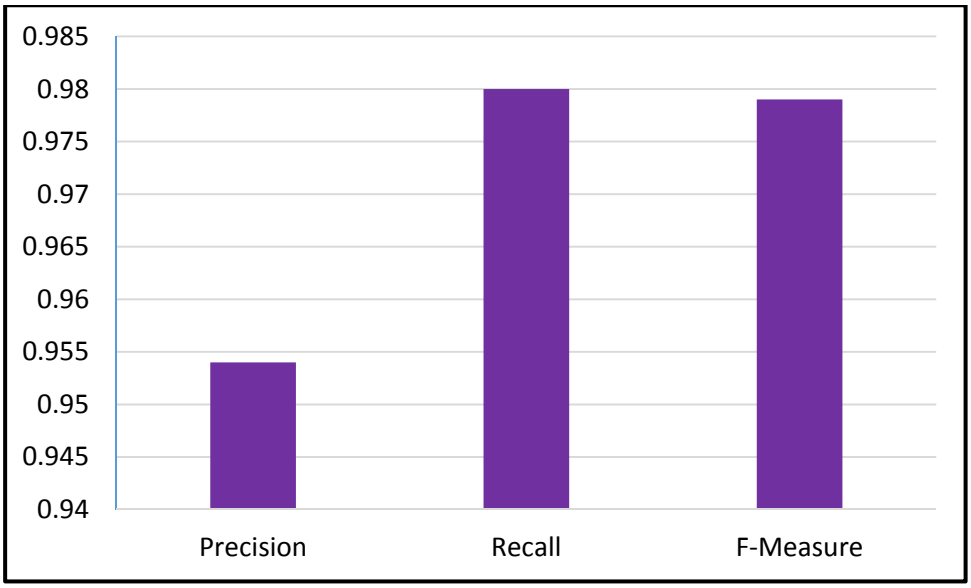

Figure 2. Graphical representation of precision, recall and F-measure

5. CONCLUSION

In this paper we presented a data mining framework to detect Internet worms. The primary feature used for the process was the flow level payload features from network flow traffic has been used in the classifier. We used the flow features common to both worms and clean programs to remove any biases caused by the features that have all their occurrences in one class only. We showed 97.90% detection rate with a 0.057% false positive rate.

REFERENCES:

[1] Kwon, S. K., Choi, Y. H., & Baek, H. (2016). Internet Worm Propagation Model Using Centrality Theory. Kyungpook Mathematical Journal, 56(4).

[2] Liang, H., Li, M., & Chai, J. (2014). Internet Worm Detection and Classification Based on Support Vector Machine. In Computer Engineering and Networking (pp. 643-651). Springer International Publishing.

[3] Patel, K. N. (2014, January). Internet Worm Detection Using Distributed Blackhole Networks. In International Journal of Engineering Research and Technology (Vol. 3, No. 1 (January-2014)). ESRSA Publications.

[4] Wattanapongsakorn, N., Wonghirunsombat, E., Assawaniwed, T., Hanchana, V., Srakaew, S., & Charnsripinyo, C. (2013, December). A network-based internet worm intrusion detection and prevention system. In IT Convergence and Security (ICITCS), 2013 International Conference on (pp. 1-4). IEEE.

[5] Rasheed, M. M. (2015). Behavioural Detection for Internet Scanning Worm Attack. Computer Science & Telecommunications, 46(2).

[6] Khule, M., Singh, M., & Kulhare, D. (2014). Netflow Traffic Analyzer For worm detection–A Survey. International Journal Of Engineering And Computer Science, Jgg, 3, 5441-5446.

[7] Chen, C., Chen, Z., & Li, Y. (2010). Characterizing and defending against divide-conquer-scanning worms. Computer Networks, 54(18), 3210-3222.

[8] Kong, D., Jhi, Y. C., Gong, T., Zhu, S., Liu, P., & Xi, H. (2011). SAS: semantics aware signature generation for polymorphic worm detection. International Journal of Information Security, 10(5), 269-283.

[9] Kolter, J. Z., & Maloof, M. A. (2004, August). Learning to detect malicious executables in the wild. In Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 470-478). ACM.

[10] Li, J. Q., Qin, Z., Ou, L., Salman, O., Liu, A. X., & Yang, J. M. (2011). Modeling and analysis of gradual hybrid anti-worm. Journal of Central South University of Technology, 18(6), 2050-2055.

[11] Wang, Q., Chen, Z., & Chen, C. (2012). On the characteristics of the worm infection family tree. IEEE Transactions on Information Forensics and Security, 7(5), 1614-1627.

[12] Yao, Y., Xie, X. W., Guo, H., Yu, G., Gao, F. X., & Tong, X. J. (2013). Hopf bifurcation in an Internet worm propagation model with time delay in quarantine. Mathematical and Computer Modelling, 57(11), 2635-2646.

[13] Zaki, M., & Hamouda, A. A. (2010). Design of a multi_agent system for worm spreading_reduction. Journal of Intelligent Information Systems, 35(1), 123-155.

[14] Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009, July). A detailed analysis of the KDD CUP 99 data set. In Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009. IEEE Symposium on (pp. 1-6). IEEE.

[15] Shannon, C., & Moore, D. (2004). The caida dataset on the witty worm. Support for the Witty Worm Dataset and the UCSD Network Telescope are provided by Cisco Systems, Limelight Networks, the US Department of Homeland Security, the National Science Foundation.

[16] Thaseen, I. S., & Kumar, C. A. (2016). Intrusion Detection Model Using Chi Square Feature Selection and Modified Naïve Bayes Classifier. In Proceedings of the 3rd International Symposium on Big Data and Cloud Computing Challenges (ISBCC–16') (pp. 81-91). Springer International Publishing.

[17] Kukielka, P., & Kotulski, Z. (2008, October). Analysis of different architectures of neural networks for application in intrusion detection systems. In Computer Science and Information Technology, 2008. IMCSIT 2008. International Multiconference on (pp. 807-811). IEEE.

[18] Ray, L. L., & Felch, H. (2014). Improving Performance and Convergence Rates in Multi-Layer Feed Forward Neural Network Intrusion Detection Systems: A Review of the Literature. International Journal of Strategic Information Technology and Applications (IJSITA), 5(3), 24-36.

[19] Brown, J., Anwar, M., & Dozier, G. (2016, August). An Evolutionary General Regression Neural Network Classifier for Intrusion Detection. In Computer Communication and Networks (ICCCN), 2016 25th International Conference on (pp. 1-5). IEEE.