HTTP Botnet Detection Using Adaptive Learning Rate Multilayer Feed-Forward Neural Network

G. Kirubavathi Venkatesh and R. Anitha Nadarajan

Department of Mathematics and Computer Applications
PSG College of Technology, Coimbatore, India
g.kiruba@gmail.com, anitha_nadarajan@mail.psgtech.ac.in

Abstract. Botnets have become a rampant platform for malicious attacks, which poses a significant threat to internet security. The recent botnets have begun using common protocols such as HTTP which makes it even harder to distinguish their communication patterns. Most of the HTTP bot communications are based on TCP connections. In this work some TCP related features have been identified for the detection of HTTP botnets. With these features a Multi-Layer Feed Forward Neural Network training model using Bold Driver Back-propagation learning algorithm is created. The algorithm has the advantage of dynamically changing the learning rate parameter during weight updation process. Using this approach, Spyeye and Zeus botnets are efficiently identified. A comparison of the actively trained neural network model with a C4.5 Decision Tree, Random Forest and Radial Basis Function indicated that the actively learned neural network model has better identification accuracy with less false positives.

Keywords: HTTP Botnet, Multilayer Feed-forward Neural Network, Bold Driver Back propagation algorithm.

1 Introduction

Botnets are organized networks of infected (Zombie) machines running bot codes, categorized by their use of a command and control (C&C) channel. Using the command and control of botnet, a botmaster can control a large group of compromised bots and then perform malicious attacks [1]. At early times, C&C communications were based on Internet Relay Chat (IRC) protocol. The attacker used to actively issue commands on the special channel of IRC server to all the bots. Recently, HTTP becomes a more popular communication protocol for bots. These web-based C&C bots try to blend into normal HTTP traffic, which makes them more difficult to be identified, since HTTP is a commonly used network communication protocol in many applications. The HTTP bots frequently request and download commands from web servers under the attacker's control. As a result, detecting bots with web-based controlling is more intricate than bots with IRC-based controlling. This paper identifies the anomaly in web flow behaviors of HTTP botnets, extracts some TCP related features and uses neural network with bold driver back propagation algorithm [2] to detect botnets. Neural networks have been successfully used in several applications like classification of internet users [3], intrusion detection at the host level [4-5] due to their advantages like parallel processing of information, capacity to recognize patterns of information in the presence of noise and handle non-linearity, capacity to classify information and quick adaptability of system dynamics. Also with the same features, training models like Decision tree, random forest and radial basis function have been created and tested for botnet detection. Extensive experimental results show that the neural network model is efficient and can detect HTTP bots with low false positive rate.

The rest of the paper is organized as follows: Section 2 presents the related work in botnet detection systems and Section 3 describes the proposed detection approach in detail. Section 4 gives the experimental results and Section 5 presents the performance evaluation of the proposed system. Finally, Section 6 concludes the paper.

2 Related Work

Botnets are now recognized as one of the most serious security threats. Detecting and tracking botnet is considered to be highly complicated and a daunting task and as a result it has become a major research topic in recent years. One of the earliest methods to detect botnet was based on the honeypots. Many researchers have used honeypots to detect botnets, but could not detect bot infection at all times. Freiling et al. [6], used honeypots to track botnets in the network and generated an early report for understanding the consequences of botnets. Goebel et al. [7] proposed a well-known signature based botnet detection technique called Rishi. Anomaly based detection approaches try to detect botnets based on a number of traffic anomalies such as high network latency, high volumes of traffic, traffic on unusual ports and unusual system behavior that could show existence of bots in the network. Binkley et al. [8] proposed an effective algorithm that combines TCP-based anomaly detection with IRC tokenization and IRC message statistics to create a system that can clearly detect client botnets. However, this approach could be easily defeated by simply using a trivial cipher to encode the IRC commands. Karasaridis et al. [9] presented an algorithm for detection and characterization of botnets using passive analysis based on flow data in transport layer and it could detect encrypted botnet communications also.

DNS-based detection techniques are based on particular DNS information generated by a botnet. Dagon [10] proposed a mechanism to identify botnet C&C servers by detecting domain names with abnormally high or temporarily concentrated dynamic DNS query rates, but this method generated many false positives. Masud et al. [11] proposed robust and effective flow based botnet traffic detection by mining multiple log files. They introduced multiple log correlation for C&C traffic detection and categorized the entire flow to identify botnet.

Chia et al. [12] proposed a novel method to detect HTTP bots based on timeslot, mutual authentication and clustering analysis. This method provided an efficient and practical approach to identify web-bots hiding in HTTP protocol. Since most of the web-based bots communicate with TCP connection, Wang B et al. [13] extracted key statistical features such as request bytes, response bytes, number of efficient packets and classified them using X-means clustering algorithm and this method was computationally light with good detection accuracy. The limitation of this work is that the proposed method will fail if the bot master uses random delay technique to break the periodicity. Guofei Gu et.al. [14], proposed BotMiner which clustered similar communication traffic and similar malicious traffic, and performed cross cluster correlation to identify the hosts that share both similar communication patterns and similar malicious activity patterns. Nogueira et.al. [23] proposed a botnet detection approach based on the identification of traffic patterns and used an Artificial Neural Network to identify the licit and illicit patterns. In this paper, a multi-layer neural network is trained with some TCP connection based features for the detection of HTTP botnets. Extensive experimental results show that the proposed method is efficient and can detect HTTP bots with low false positive rate.

3 Proposed Botnet Detection System

The proposed botnet detection uses a Multilayer Feed-forward Neural Network with adaptive learning rate. Since HTTP botnets communicate with TCP connection, in this paper we have extracted TCP connection related features. To increase the dynamic range of learning rate coefficient, bold driver back-propagation algorithm is used. The block diagram of the proposed system is given in Fig. 1.

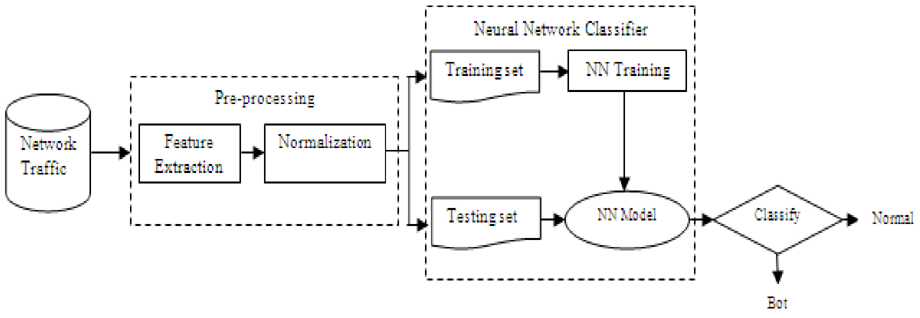


Fig. 1. Block diagram of the proposed system

3.1 Feature Extraction

Web botnets do not maintain a connection with C&C server, but they periodically download the instructions using web requests on a regular interval from the web server which is under the attacker control. Since, most of the communications between the web botnets is based on TCP connection, relative and direct features of TCP connections are extracted from the network traffic. Tu Xu et.al [25] have used some relative features like one-way connection density, ratio of TCP packets etc to efficiently identify DDoS attack. In this work, we use some of the relative features used by Tu Xu et.al for the detection of HTTP botnet. When the infected bot communicate with the web server which is under the control of the attacker, the number of half open connections will increase and hence one way connection ratio and Ratio of incoming and outgoing TCP packets are suitable features to reflect this scenario. In normal traffic, randomness will be there in the number of SYN flag, FIN flag and PSH flag send by a host, but when a bot communicates with a C&C server periodically there is a pattern and these counts will be high and almost a fixed one. Hence the SYN, FIN and PSH flag counts in the TCP packets send by the bot are considered as features.

In this work, the following TCP features are used for efficient detection of HTTP botnet.

- $\text{One} - \text{way } \mathit{connection} \text{ ratio of TCP} = \frac{\text{Number of one-way connection TCP packets}}{\text{Number of TCP packets}} * 100$
- $\text{Ratio of Incoming Outgoing TCP packets} = \frac{\text{Number of Incoming TCP packets}}{\text{Number of Outgoing TCP packets}}$
- $\text{Ratio of TCP packets} = \frac{\text{Number of TCP packets}}{\text{Total } \mathit{number} \text{ of packets}}$
- SYN Flag Count – the number of TCP packets with SYN flag set
- FIN Flag Count – the number of TCP packets with FIN flag set
- PSH Flag Count – the number of TCP packets with PSH Flag set

The selected features of TCP connections give considerable information associated with the presence of web-based botnet.

3.2 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[15].

$$x' = \frac{x - \min_x}{\max_x - \min_x} \tag{1}$$

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

3.3 Classification Using Neural Network

Multilayer Feed-forward networks [16-17] 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 bold-driver back propagation learning algorithm, which is an appropriate learning algorithm for training multilayer feed- forward networks for vector classification. The input layer will have 6 neurons, corresponding to the dimensionality of the input vectors, and the output layer will have one neuron. 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 networks, is minimized. Each neuron i in the input layer has a signal x_i as network's input and each neuron j in the hidden layer receives the signal *In(j)* given by

$$In(j) = \theta_j + \sum_{i=1}^{n} x_i\, w_{ij} \tag{2}$$

where w_{ij} is the weight between the input layer and hidden layer and θ_j is the bias in the hidden layer. Then the signal *In(j)* is passed through sigmoid activation function to produce f(In(j)). The output of the output layer is calculated as

$$y_k = \theta_k + \sum_{j=1}^{m} w_{jk}\, f(In(j)) \tag{3}$$

The output value will be compared with the target and error will be estimated. The bold driver back propagation learning algorithm is used to speed up the convergence of network, in which, the weights are updated after the presentation of all input patterns and the learning rate η is varied with each weight update with the rule,

$$w(t+1) = w(t) - \eta(t)\left[\sum_{k} \frac{\partial E(t)}{\partial w(t)}\right] + \alpha \Delta w(t) \tag{4}$$

where η is the learning rate, α is the momentum term $(0 < \alpha < 1)$.

Also we have used C4.5 decision tree, Random Forest and Radial Basis Function classifiers for the detection of botnets with the same set of features and a comparison is made between their performances. C4.5 Decision tree is an entropy based approach in which the attribute with the highest information gain ratio is the one used to make the decision [18]. Random Forest (RF) [19] is a multi-class prediction algorithm that can be used to predict either a categorical or a continuous response. Radial Basis Function (RBF) neural network is another type of feed-forward neural network and it performs the classification by measuring distances between inputs and the centers of the RBF hidden neurons [20]. Jiang et al. [21] and Zhang et al. [22] compared the RBF and MLFF-BP networks for misuse and anomaly detection on the KDD99 dataset. Their experiments have shown that for misuse detection, BP has a slightly better performance than RBF in terms of detection rate and false positive rate.

4 Experimental Results and Analysis

Botnet setup is created in our SSE lab Network that simulates the behavior of the existing real time HTTP botnet. We have used seven systems (1-BotMaster, 5- Zombies and 1-Command & Control Server) for our experimental purpose (Fig. 2). The physical topology is star and the speed of the Ethernet cable is 100baseTX.

TCP connections features are extracted from the network traffic by sampling at an interval of 5 seconds. The bot traces have been collected for 2 hours a day for five days in a week. In a similar manner, normal web traffic has been collected from the National Knowledge Network with the bandwidth of 100Mbps/s. The working mechanism is mentioned as below:

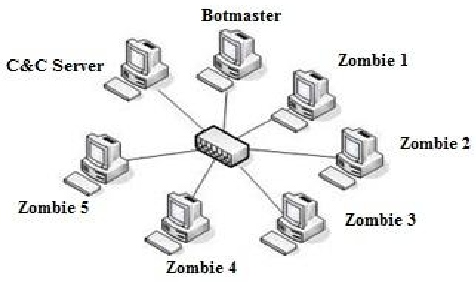


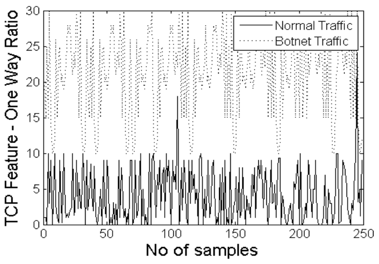
Fig. 2. Experimental Setup

Botmaster sends the Trojan or backdoor to these zombies using email spamming technique. Using the backdoor, botmaster installs the HTTP bot binaries in these machines. Now the victim machine will periodically communicate with the Command & Control server and follows the instructions from the botmaster without the knowledge of the user. In this work, Spyeye [23] and Zeus [24] bots are used and table 1 shows the information of the bot traces collected. The traces are analyzed for the selected features with normal web traffic and botnet flow. The neural network is trained with 10250 normal samples, 8500 Zeus bot samples and 8250 Spyeye bot samples. The learning rate of the neural network is varied from 0.4 to 0.7. Fig.3 (a) shows the one way connection ratio of TCP Connection feature for Normal and Spyeye flow and Fig. 3(b) shows that of normal and Zeus. The graph clearly depicts that for normal flow the value of one way ratio is small, while for botnet flow it is comparatively large.

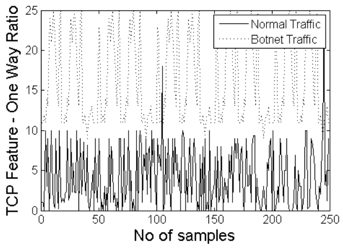
Table 1. Traces of different Web-based Botnets

Bot Fam-	Trace Size	Packets number
Zeus-1	5.85 MB	53,220
Zeus-2	4.13 MB	37,252
Spyeye-1	25.17 MB	1,75,870
Spyeye-2	3.90 MB	35,180

In case of Ratio of Incoming Outgoing TCP packets, from Fig. 4(a) we can see that for normal flow, there is an irregular deviation while in case of Spyeye it maintains similarity in the connections. Fig. 4(b) shows the feature for normal and Zeus flow.

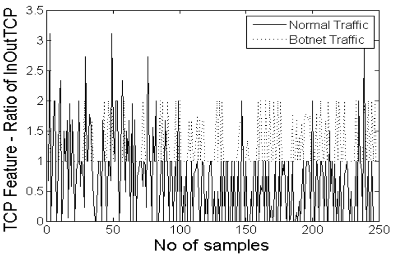


(a) For Normal and Spyeye flow

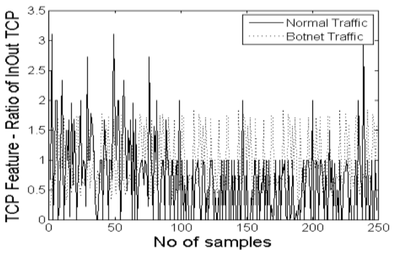


(b) For Normal and Zeus flow

Fig. 3. One-way connection ratio of TCP packets



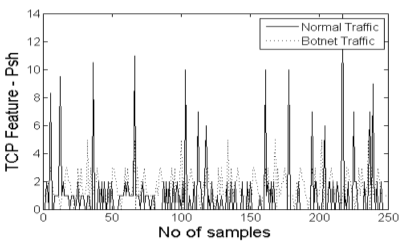
a) For Normal and Spyeye flow



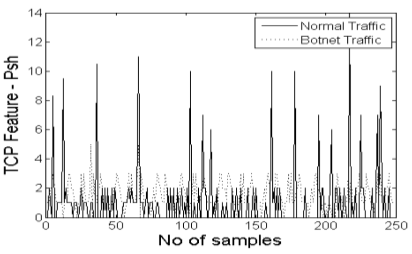
(b) For Normal and Zeus flow

Fig. 4. Ratio of Incoming Outgoing TCP packets

Fig. 5(a) shows PSH feature for Normal and Spyeye flow. In case of PSH, normal traffic and botnet traffics are mixed. Normal traffic flows have randomness, while web botnet maintains some similarity in connections. Fig. 5(b) displays PSH feature for Normal and Zeus flow. Testing is done with 5200 normal samples, 3100 zeus bot samples and 3200 sypeye bot samples. For testing, both trained and un-trained data have been used. For the best network performance, an optimal number of hidden neurons must be properly determined using the trial and error procedure. We experimented with 4 neurons to 18 neurons in the hidden layer. With 18 neurons in the hidden layer; the correct identification accuracy is increased within the minimum number of epochs. Table 2 gives the correct identification percentage of different botnets.



(a) For Normal and Spyeye flow



(b) For Normal and Zeus flow

Fig. 5. TCP PUSH Flag

Table 2. Identification accuracy of web botnet traffic profiles

Traffic traces	No. of Neurons in input layer	No. of Neurons in the hidden layer	Correct Identification
Spyeye-1	6	18	99.03%
Spyeye-2	6	18	99.02%
Zeus-1	6	18	99.01%
Zeus-2	6	18	99.04%

5 Performance Evaluation

The proposed method has been evaluated with different classifiers. The performance of the neural network model is compared with that of C4.5 decision tree, Random Forest and Radial Basis Function algorithm.

To evaluate the performance of neural network recognition system, measures like accuracy, precision, recall and F-measure are calculated as bellow:

$$Accuracy(A) = \frac{Number\ of\ correctly\ classified\ \ patterns}{Total\ number\ of\ patterns}$$

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

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

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

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

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

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

Table 3 gives the accuracy of the botnet detection system using various classifiers with the same set of TCP features and the performance measures of the proposed detection system for Spyeye Botnet and Zeus botnet. It shows Multilayer Feed Forward Neural Network with Bold driver back propagation learning algorithm gives the better accuracy compared with other methods.

The proposed model's performance is compared with some of the existing botnet detection techniques and table 4 shows the comparison results. Another performance indicator is the ROC curve. The ROC curve shows the tradeoff between the detection rate and false-positive rate under various detection algorithms. Our detection methodology achieves a detection rate of 99% with less than 1% of false positive. This result proves the effectiveness of the proposed detection mechanism.

Table 3. Performance Measures of Spyeye and Zeus Botnet

Method	Botnet	Precision	Recall	F-Measure	Accuracy
Decision Tree	Spyeye	0.968	0.931	0.949	96.5333
Random Forest		0.968	0.934	0.950	96.667
RBF		0.976	0.927	0.950	96.5333
Proposed Model		0.964	0.983	0.973	99.03
Decision Tree	Zeus	0.956	0.930	0.941	96.1333
Random Forest		0.952	0.930	0.940	96.00
RBF		0.959	0.922	0.940	95.8667
Proposed Model		0.948	0.992	0.969	99.04

Table 4. Comparison of Performance

Method	Average Detection Accuracy
Gu et al (2008) BotMiner [14]	96.825
Anotnio, Nogueira et al (2010) [23]	94.9175
Proposed NN Model	99.025

Fig. 6 shows the ROC curve for Spyeye botnet whereas Fig. 7 shows that of Zeus botnet. From the figures we can conclude that the proposed method outperforms other classifiers.

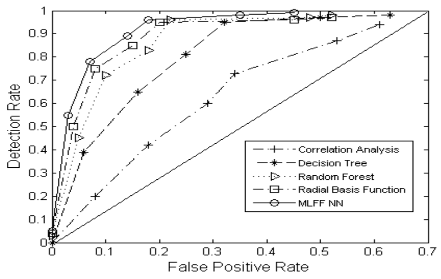


Fig. 6. ROC curve for Spyeye Botnet

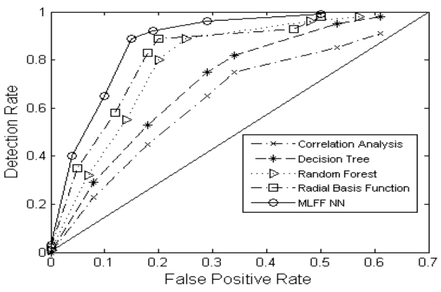


Fig.7. ROC curve for Zeus Botnet

6 Conclusion

In this work, we propose a new method to identify HTTP-based botnet by using the network behavior of botnet. On observation of activities of web-based botnet, we notice that most of the communications of web-based botnets are based on TCP connections. The TCP connection behavior shared by web-based botnets have been extracted, used as features and a neural network model is created and it is able to detect the HTTP botnet traffic. The proposed detection method's performance is compared with that of some more classifiers like Decision Tree C4.5, Random forest, Radial Basis function network. The results obtained showed that the proposed method using neural network is able to achieve good identification results with less false positive. Another advantage of the proposed model is that it can detect HTTP botnets even if the communications are encrypted.

Acknowledgement. This work is a part of the Collaborative Directed Basic Research on Smart and Secure Environment project, funded by NTRO, Govt. of India.

References

1. Lai, G.H., Chen, C.M., Tzeng, R.Y., Laih, C.S., Faloutsos, C.: Botnet Detection by Abnormal IRC Traffic Analysis. In: Proceedings of the Fourth Joint Workshop on Information Security, JWIS (2009)
2. Sarkar, D.: Methods to speed up error back-propagation learning algorithm. ACM Computing Surveys 27(4), 519–542 (1995)
3. Nogueira, A., de Oliveira, M.R., Salvador, P., Valadas, R., Pacheco, A.: Classification of internet users using discriminant analysis and neural networks. In: First Conference on Traffic Engineering for the Next Generation Internet, pp. 341–348 (April 2005)
4. Debar, H., Becker, M., Siboni, D.: A neural network component for an intrusion detection system. In: Proceedings of the ACM/IEEE Symposium on Research in Security and Privacy, Los Almitos, CA, May 4-6, pp. 240–250 (1992)
5. Salvador, P., Nogueira, A., Franca, U., Valadas, R.: Framework for Zombie Detection Using Neural Networks. In: Proceedings of the Fourth International IEEE Conference on Internet Monitoring and Protection ICIMP 2009, pp. 14–16 (2009)
6. Freiling, F.C., Holz, T., Wicherski, G.: Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks. In: di Vimercati, S.d.C., Syverson, P.F., Gollmann, D. (eds.) ESORICS 2005. LNCS, vol. 3679, pp. 319–335. Springer, Heidelberg (2005)
7. Goebel, J., Holz, T.: Rishi: Identify bot contaminated hosts by irc nickname evaluation. In: Proceedings of USENIX HotBots 2007 (2007)
8. Binkley, J.R., Singh, S.: An algorithm for anomaly based botnet detection. In: Proceedings of the 2nd Conference on Steps to Reducing Unwanted Traffic on the Internet (SRUTI 2006), San Jose, CA (July 2006)
9. Karasaridis, A., Rexroad, B., Hoeflin, D.: Wide-scale botnet detection and characterization. In: First Workshop on Hot Topics in Understanding Botnets (HotBots 2007), Cambridge, MA (April 2007)
10. Dagon, D.: Botnet Detection and Response. In: Operations, Analysis and Research Center Workshop (July 2005)
11. Masud, M.M., Al-khateeb, T., Khan, L., Thuraisingham, B., Hamlen, K.W.: Flow- based identification of Botnet traffic by mining multiple log files. In: Proceedings of the International Conference on Distributed Framework & Application, Penang, Malaysia (2008)
12. Chen, C.-M., Ou, Y.-H., Tsai, Y.-C.: Web Botnet Detection based on Flow Information. In: International Computer Symposium 2010, pp. 381–384. IEEE (2010)
13. Wang, B., Li, Z., Li, D., Liu, F., Chen, H.: Modeling Connections Behavior for WebBased Bots Detection. In: IEEE International Conference on e-Business and Information System Security, EBISS 2010, Wuhan, pp. 1–4 (2010)

14. Gu, G., et al.: BotMiner: Clustering Analysis of Network traffic for protocol and structure independent botnet detection. In: Proceedings of 17th Conference on Security Symposium, pp. 139–154. ACM Digital Library (2008)
15. Shalabi, A.L., Shaaban, Z.: Normalization as a preprocessing engine for data mining and the approach of preference matrix. In: Proceedings of the International IEEE Conference on Dependability of Computer Systems, 2006, pp. 207–214 (2006)
16. Moradi, M., Zulkernine, M.: A neural network based system for intrusion detection and classification of attacks. In: Proceedings of the 2004 IEEE International Conference on Advances in Intelligent Systems - Theory and Applications, Luxembourg-Kirchberg, Canada, November 15-18 (2004)
17. Kukiełka, P., Kotulski, Z.: Analysis of Different Architectures of Neural Networks for Application in Intrusion Detection Systems. In: Proceedings of the IEEE International Multi Conference on Computer Science and Information Technology, pp. 807–811 (2008)
18. Abbes, T., Bouhoula, A., Rusinowitch, M.: Protocol Analysis in Intrusion Detection Using Decision Tree. In: Proceedings of the IEEE International Conference on Information Technology: Coding and Computing (ITCC 2004), pp. 404–408 (April 2004)
19. Zhang, J., Zulkernine, M., Haque, A.: Random-Forests-Based Network Intrusion Detection System. IEEE Transactions on Systems, Man, and Cybernetics 38(5), 649–659 (2008)
20. Rapaka, A., Novokhodko, A., Wunsch, D.: Intrusion detection using radial basis function network on sequence of system calls. In: Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN 2003), Portland, OR, USA, July 20-24, vol. 3, pp. 1820–1825 (2003)
21. Jiang, J., Zhang, C., Kamel, M.: RBF-based real-time hierarchical intrusion detection systems. In: Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN 2003), Portland, OR, USA, July 20-24, vol. 2, pp. 1512–1516. IEEE Press (2003)
22. Zhang, C., Jiang, J., Kamel, M.: Comparison of BPL and RBF Network in Intrusion Detection System. In: Wang, G., Liu, Q., Yao, Y., Skowron, A. (eds.) RSFDGrC 2003. LNCS (LNAI), vol. 2639, p. 466–470. Springer, Heidelberg (2003)
23. Anotnio, N., Salvador, P., Blessa, F.: A Botnet Detection System Based on Neural Networks. In: Proceedings of Fifth International Conference on Digital Telecommunications, pp. 57–62 (2010)
24. Binsalleeh, H., Ormerod, T., Bouhtouta, A., Sinha, P., Youssef, A., Debbabi, M., Wang, L.: On the Analysis of the Zeus Botnet Crimeware Toolkit. In: Proceedings of the IEEE Eighth Annual Conference on Privacy, Security and Trust, PST, Ottawa, Canada, August 17–19 (2010)
25. Xu, T., He, D., Luo, Y.: DDoS attack detection based on RLT features. In: Proceedings of International Conference on Computational Intelligence and Security, pp. 697–700 (2007)