Hindawi
Mathematical Problems in Engineering
Volume 2017, Article ID 4934082, 9 pages
https://doi.org/10.1155/2017/4934082

Research Article

An Effective Conversation-Based Botnet Detection Method

Ruidong Chen,[1,2] Weina Niu,[1,2] Xiaosong Zhang,[1,2] Zhongliu Zhuo,[1,2] and Fengmao Lv[1]

[1] *School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan 611731, China*
[2] *Center for Cyber Security, University of Electronic Science and Technology of China, Chengdu, Sichuan 611731, China*

Correspondence should be addressed to Xiaosong Zhang; johnsonzxs@uestc.edu.cn

Received 25 January 2017; Accepted 12 March 2017; Published 9 April 2017

Academic Editor: Lixiang Li

A botnet is one of the most grievous threats to network security since it can evolve into many attacks, such as Denial-of-Service (DoS), spam, and phishing. However, current detection methods are inefficient to identify unknown botnet. The high-speed network environment makes botnet detection more difficult. To solve these problems, we improve the progress of packet processing technologies such as New Application Programming Interface (NAPI) and zero copy and propose an efficient quasi-real-time intrusion detection system. Our work detects botnet using supervised machine learning approach under the high-speed network environment. Our contributions are summarized as follows: (1) Build a detection framework using PF_RING for sniffing and processing network traces to extract flow features dynamically. (2) Use random forest model to extract promising conversation features. (3) Analyze the performance of different classification algorithms. The proposed method is demonstrated by well-known CTU13 dataset and nonmalicious applications. The experimental results show our conversation-based detection approach can identify botnet with higher accuracy and lower false positive rate than flow-based approach.

1. Introduction

Botnet [1] comprises many compromised hosts under the control of the botmaster remotely. Early botnets relied on Internet Relay Chat (IRC) [2] and Hypertext transfer protocol (HTTP) [3] to communicate. The problem of that is single-point invalid and easy to be detected and destroyed. Most botnets are decentralized and use P2P technology [4] to construct command and control (C&C) mechanism [5]. Noncentral node P2P botnet [6] is harder to detect than IRC and HTTP-based one. What is more, bots evolve into attacks which are difficult to track their position. Most current Denial-of-Service (DoS) and spam are caused by botnet [7]. Thus, the botnet is one of the greatest threats to network security.

In the past, researchers used signature-based [8] and anomaly-based intrusion detection systems (IDS) [9, 10] to detect botnet. However, the former has two shortcomings: one is that the original detection rules cannot effectively detect bot program that changes communication means; the other is that inaccurate signatures cause high false positive. Lack of scalability under huge network traffic is another problem.

Currently, the backbone network is based on 1 Gbps or 10 Gbps optical fibers, which renders massive traffic data in short time. Moreover, fast growing P2P applications pose significant strain to data storage. Therefore, identifying botnet traffic under high-speed network is a challenging issue [11]. In this paper, a detection platform with high detection accuracy and powerful traffic processing ability is proposed. It uses conversation-based network traffic analysis and supervised machine learning to identify malicious botnet traffic. The experimental results show that random forest algorithm [12] has higher detection accuracy and lower false positive rate. Moreover, we further explore the top five classifiers (RandomForest, REPTree, RandomTree, BayesNet, and DecisionTump [13]).

The contributions of the paper are threefold. First, a novel botnet detection system with low latency and high accuracy is introduced. Second, our detection method identifies botnet traffic using conversation-based traffic analysis and supervised machine learning. Our approach outperforms the accuracy based on flow since the false positive rate of botnet traffic decrease is 13.2 percent. In addition to the above two, we evaluate performances of the five well supervised machine learning algorithms (MLAs) [14]. The detection rate of the botnet is up to 93.6%, and the false alarm rate is about 0.3% by the random forest algorithm.

The remainder of this paper is organized as follows: Section 2 gives an overview of botnet detection related works; Section 3 shows the proposed detection method; Section 4 provides the preliminary experimental results; conclusion are summarized in Section 5.

2. Related Work

Botnet detection methods fall into two categories: host behavior-based detection [15] and network-based detection [16].

2.1. Host-Based Detection. Host-based detection is the earliest method. To determine whether a host is compromised, this method continuously monitors the change of process, files, network connections, and registries under a controlled environment [17, 18]. Host-based detection is useful in detecting known bots. However, it performs poorly, because it cannot detect new or variant bots. For example, host-based detection has a sense of inability to identify bots with new technologies like a rootkit, counter debug.

2.2. Network-Based Detection. Network-based detection [19–21] mainly identifies traffic in C&C control phrase of a botnet, because behavior features in this phrase are different from other phrases. Network-based detection mainly focuses on analyzing two kinds of network behaviors: the rate of failed connection and flow features. Most commonly used flow features include the number of uplink (downlink) data packets, the number of uplink (downlink) transmission bytes, the average variance-length of uplink (downlink) data packets, the maximum length of uplink (downlink) data packets, the average variance-length of uplink (downlink) data packets, the duration time of data flow (ms), the rate of the length of data packets in uplink and downlink, and the total length of loaded data packets in a flow. Nowadays, researchers introduce machine learning and neural network to network-based detection to identify unknown botnet traffic. Thus, this method is a hot research point in recognition and analysis of botnet traffic.

Network-based detection method has a high detection rate because it extracts common flow features independent of botnet category. However, in the high-speed and complex network, existing detection platforms based on flow features are ineffective due to high packet drop rate.

3. Our Detection Method

In this section, we describe the components of our proposed botnet traffic detection framework.

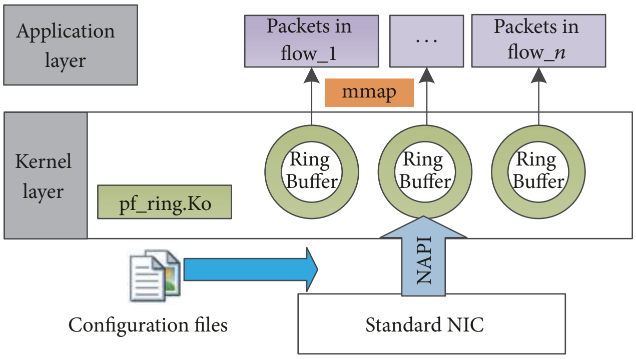


Figure 1: The packet process module architecture.

The framework consists in the following:

(1) Traffic process module for clustering captured packets into different flow buffer

(2) Flow-based feature extraction module for generating statistical characteristics of flow

(3) Conversation-based feature selection module for extracting promising conversation-based feature set

(4) Botnet detection module for identifying botnet traffic using machine learning algorithm.

Packet process module is used to extract the required fields out of the packets. After the extraction of the desired information from the packet process module, the flow-based feature extraction module is used for generating flow features. Based on the flow features, conversation-based feature selection module can obtain promising conversation feature set for the botnet detection module. Botnet traffic detection is accomplished using supervised classification algorithm [22].

3.1. Traffic Process Module. Libcap [23] is used for sniffing the packets from the network interface due to its simple operation and cross-platform. After the NIC captures the packets, Libcap copies packets from the driver to kernel-level using DMA in order to filter them. Then, Libcap copy filtered packets into application in user level for further analyzing, whereas multiple copies of Libcap impose more overhead and consume more time. The mechanics of Libcap makes packet loss and do not reduce the user session. PF_RING [24] is a new network socket that uses New Application Programming Interface (NAPI) and zero copy to capture packet data from a live network. Thus, PF_RING is used to capture traffic onto successive pcaps. The detailed packet capture process is shown in Figure 1.

First, the kernel layer of the packet process module reads the configuration file to set the parameter values, like packet length, ClusterId. ClusterId is the ID of Ring Buffer created by PF_RING. Parameter values are stored in the configuration file so that we can modify them at any time. Second, network devices are turned on and Ring Buffer is created using pfring_open (device, snaplen, flags) function of PF_RING, where device denotes the name of the network device, snaplen denotes the packet length, and flags denotes

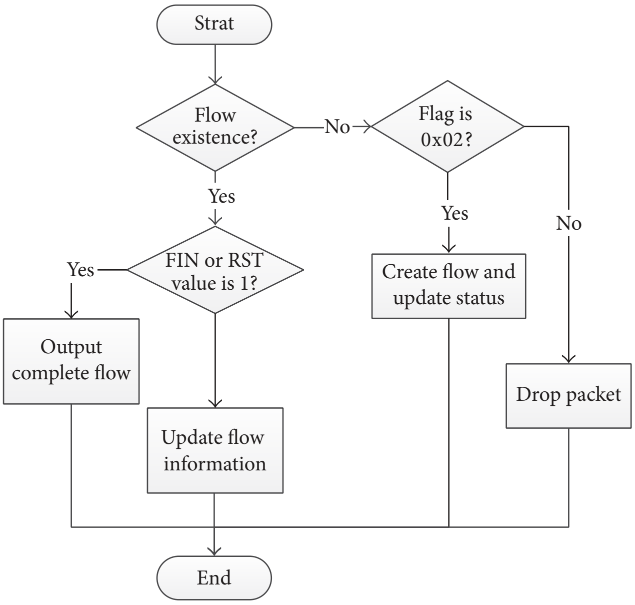


Figure 2: The packet process module architecture.

whether it is in mixed mode. Here, we set snaplen value as 60 because header fields of a packet are needed in this paper. Third, we save the header information, payload length, and arrival time of a packet in different flow buffer according to five tuples (SrcIp, DstIp, SrcPort, DstPort, Proto), which is used to mark a flow. That is, if two different packets have the same source/destination host/port and the same protocol, they belong to the same flow.

3.2. Flow-Based Feature Extraction Module. There are different flow reorganization methods for different transport layer protocols. Using TCP packets as an example, we use a three-way handshake to represent the start of a flow. When a packet whose FIN or RST value is 1 comes, the end of this flow is marked. The detailed TCP flow reorganization process is shown in Figure 2.

When a packet comes, we decide whether the flow this packet belongs to exists. If a packet whose flag value is 0x02, and the flow does not exist, we create a flow according to Ip, protocol, and port. When the flag of the packet takes other values, this packet needs to be dropped. An instance of a flow reorganization state machine can be in only one of the five states: handshake_1, handshake_2, handshake_3, data transmission, and end. If a packet whose flag value is 0x02, this process is in the status of handshake_1. Only when a packet whose flag value is 0x12 is coming, the flow reorganization will be in handshake_2 status. Then, the arrival of a packet whose flag value is 0x10 marks handshake_3 status. After the three-way handshake, data begins transmitting. In the procedure of flow reorganization, whenever there is a packet whose flag value is 0x02, it turns back to the handshake_2 status.

After analyzing the data characters of a botnet, we find that there is a flow similarity of the same botnet. Here, a conversation contains many flows with different source ports. That is, two flows having the same source/destination IP, destination port, and protocol can be classified as the same conversation. Promising conversation feature generating is based on the flow features. Thus, the flow-based feature module extracts statistical features including flow duration, the average interval of up (down) flow, the maximal/minimum/average length of up (down) flow, the number of valid up (down) packets in a flow, the number of transmission bytes of up (down) flows, and the number of small packets in a flow.

3.3. Conversation-Based Feature Selecting Module

3.3.1. Conversation Features

(1) The Duration Time of Flows in a Conversation. The communication between the botmaster and other bot hosts is done by bots. Thus, the duration time of botnet flow is usually fixed and short. However, the duration time of normal flow is determined by user behaviors. Here, we can extract the average duration time of flows in a conversation (avg_duration), the minimum and maximum duration time of flows in a conversation (min_duration, max_duration), the standard deviation of duration time of flows in a conversation (std_duration), and the average arriving intervals of up and down flows in a conversation (avg_finter, avg_binter). Assuming that there are n flows in a conversation, avg_finter = (avg_finter_f_1+$\cdots$+avg_finter_f_n)/n, where avg_finter_f_1 denotes the average arriving intervals of up packets in the first flow.

(2) The Distribution of Flows in Conversation. During the communication process among nodes in a botnet, the size and the number of transmitted data packets are small. And C&C communication flows produced from bot hosts in the same botnet have great similarity [25]. Thus, we extract the average length of up and down flows in a conversation (avg_fpkl, avg_bpkl), the minimum length of up and down flows in a conversation (min_fpkl, min_bpkl), the maximum length of up and down flows in a conversation (max_fpkl, max_bpkl), the standard variation of the length of up and down flows in a conversation (std_avg_fpkl, std_avg_bpkl), the average number of valid up and down flows in a conversation (avg_fpks, avg_bpks), the standard variation of the number of valid up and down flows in a conversation (std_avg_fpks, std_avg_bpks), the average number of transmission bytes of up and down flows in a conversation (avg_fpksl, avg_bpksl), and the standard variation of transmission bytes of up and down flows in a conversation (std_fpksl, std_bpksl).

(3) The Distribution of Small Packets in Conversation. There are many packets within the range of 40320 bytes [26] in the botnet traffic because bots need to constantly connect to new hosts. However, a packet size of the benign server traffic is

large. Thus, the distribution of small packet in a conversation is an interesting characteristic of botnet traffic, like the minimum of small packet in a conversation (min_spacket), the maximum of small packet in a conversation (max_spacket), the average of small packet in a conversation (avg_spacket), and the standard variance of small packet in a conversation (std_spacket). In conclusion, there are 26 features extracted from conversations, which are shown in Table 1.

3.3.2. Feature Selection. We use random forest algorithm [12] to select promising features. All the classification trees in random forest is binary tree. Construction of classification tree meets the principle of recursive splitting from top to bottom. For each classification binary tree, all the train set they used is sampled from the original dataset. In other words, several samples in the original train set may appear many times in the train set of one classification tree and may never appear in any classification tree samples. Algorithm 1 describes how to construct a random forest algorithm in detail.

In the procedure of random forest model establishment, Gini coefficient is used to select feature. Here are 2 classes; thus, the value of K is 2. We suppose that feature A_i ($i = 1, \ldots, 26$) splits dataset D into N parts: $D_1; \ldots; D_n$. On the condition of the feature A_i, Gini coefficient of dataset D is shown as follows:

$$\begin{gathered} \operatorname{Gini}(D, A\_i) = \sum_{n=1}^{N} \frac{|D_n|}{|D|} \operatorname{Gini}(D_n), \\ \operatorname{Gini}(D_n) = 1 - \sum_{k=1}^{K} \left(\frac{|C_k|}{|D_n|} \right)^2 . \end{gathered} \tag{1}$$

Then, we select promising features according to random forest model. The feature selection process is shown in Algorithm 2. In the algorithm, input data with 27 columns includes 26 conversation features and a class label.

In every iteration, we first rank features according to their importance and then delete the feature with minimum value until detection rate no longer changes. The formula for calculating an RF score of features is shown in (2). In the following equation, there is M decision tree with feature A_i. Gini(D, A_i) indicates Gini coefficient of dataset D using feature A_i in the current decision tree.

$$\text{RF\_score}(A\_i) = \sum_{m=1}^{M} K * \prod_{n=1}^{N_{-m}} \frac{|D_n|}{|D|} - \operatorname{Gini}(D, A\_i) . \tag{2}$$

Depending on the following random forest model, the detection rate is generated using testing data. In this work, we use the features including {std_bpksl, std_avg_fpkl, std_avg_fpks, std_f(b)pksl, std spacket}. Feature vectors constructed in this paper include {SrcIp, DstIp, DstPort, Pro, std_bpksl, std_avg_fpkl, std_avg_fpks, std_f(b)pksl, std_spacket}, and using {SrcIp, DstIp, DstPort, Proto} represents the conversion of visiting the same service.

Table 1: Conversation features.

Feature value	Description of feature value
avg_duration	The average duration time of flows in a conversation
min_duration	The minimum duration time of flows in a conversation
max_duration	The maximum duration time of flows in a conversation
std_duration	The standard deviation of duration time of flows in a conversation
avg_f(b)inter	The average interval of up (down) flows in a conversation
avg_f(b)pkl	The average length of up and down flows in a conversation
min_f(b)pkl	The minimum length of up (down) flows in a conversation
max_f(b)pkl	The maximum length of up (down) flows in a conversation
std_avg_f(b)pkl	The standard variation of the length of up (down) flows in a conversation
avg_f(b)pks	The average number of up (down) valid flows in a conversation
std_avg_f(b)pks	The standard variation of the number of up (down) valid flows in a conversation
avg_f(b)pksl	The average of transmission bytes of up (down) flows in a conversation
std_f(b)pksl	The standard variation of transmission bytes of up (down) flows in a conversation
min_spacket	The minimum of small packet in a conversation
max_spacket	The maximum of small packet in a conversation
avg_spacket	The average of small packet in a conversation
std_spacket	The standard variance of small packet in a conversation

3.4. Botnet Detection Module. In order to achieve scalability in botnet detection module, we use API provided by Weka to implement machine learning algorithms [14]. The conversation feature need be saved in CSV format at the conversation-based feature selecting

```
Input: data a labeled dataset with $p$ features
Output: PF a random forest model
$n \leftarrow$ the number of decision trees, $m \leftarrow$ the number of selected features
initialization $m = M, n = N, i = 1$
while $i \leq N$ do
    draw a bootstrap sample $Z^{*}$ of size $S$ from data
    repeat
        select $M$ features at random from the $p$ features
        calculate the Gini coefficient of selected $M$ features
        select the feature with lower Gini coefficient among the $M$
        split the node into two daughter nodes
    until the minimum node size is reached
    construct decision tree $i$
    $i = i + 1$
end while
```

Algorithm 1: Random forest algorithm.

```
Input: train_data a labeled training set, test_data a labeled testing set
Output: PF a list of promising features
$\delta \leftarrow$ an error range, $BD\_l \leftarrow$ the botnet traffic detection rate, $BD\_c \leftarrow$
the current botnet traffic detection rate
initialization $i = 1, \delta = \Theta$
$BD\_l = BD\_c$ = Randonforest (all features)
while $|BD\_c - BD\_l| \leq \Theta$ do
    $BD\_l = BD\_c$
    calculate RF scores of importance
    rank the RF scores
    delete the feature with the smallest importance from train_data and test_data
$BD\_c$ = randomforest (remaining_features)
end while
```

Algorithm 2: Feature selection algorithm.

module. First, the module reads feature vectors using weka.core.converters.ConverterUtils.DataSource. However, some features like SrcIp, DstIp, DstPort, and Proto have no efficiency in identifying botnet traffic. Second, this module deletes them using weka.core.Instances. Third, we use random forest algorithm to train these data through weka.classifiers.trees.RandomForest. Fourth, this module uses the trained classifier to predict unlabeled data by calling classifyInstance(unlabeled.instance (i)) function. Here, "unlabeled" denotes testing data without a label.

4. Experimental Results and Performance Analysis

4.1. Experimental Setup. Famous public datasets used to detect botnet traffic include dataset disclosed from Information Security and Object Technology (ISOT) organization [25], Stratosphere [27], and the CTU University [28]. The dataset from Stratosphere contains many types of botnet behaviors traffic, such as the traffic of scanned port, traffic of C&C communication, and attack traffic. However, most botnet traffic of this dataset is IRC and HTTP botnet, and there is only one type of P2P botnet traffic. The dataset from ISOT only contains three types of botnet traffic, Waledac, Storm, and Zeus, and many background traffic. The dataset from the CTU University consists of thirteen scenarios of different botnet samples. Thus, in the experiment, we use dataset from CTU University. And the distributions of botnet types about training and test in our experiment are listed in Tables 2 and 3. For example, Rbot contains three types of botnets, namely, IRC, DDoS, and the US.

4.2. The Results and Analysis of Experiments. During the process of experiment, we assess our detection method by adopting the train set and test set from CTU13. The CUT13 dataset provides a better test environment for unknown botnet because this test set contains many types of botnet traffic which do not exist in the training set.

The effectiveness of the top five classifiers, namely, random forest, REPTree, randomTree, BayesNet, and DecisionTump [29], has been studied with the CTU botnet traffic and

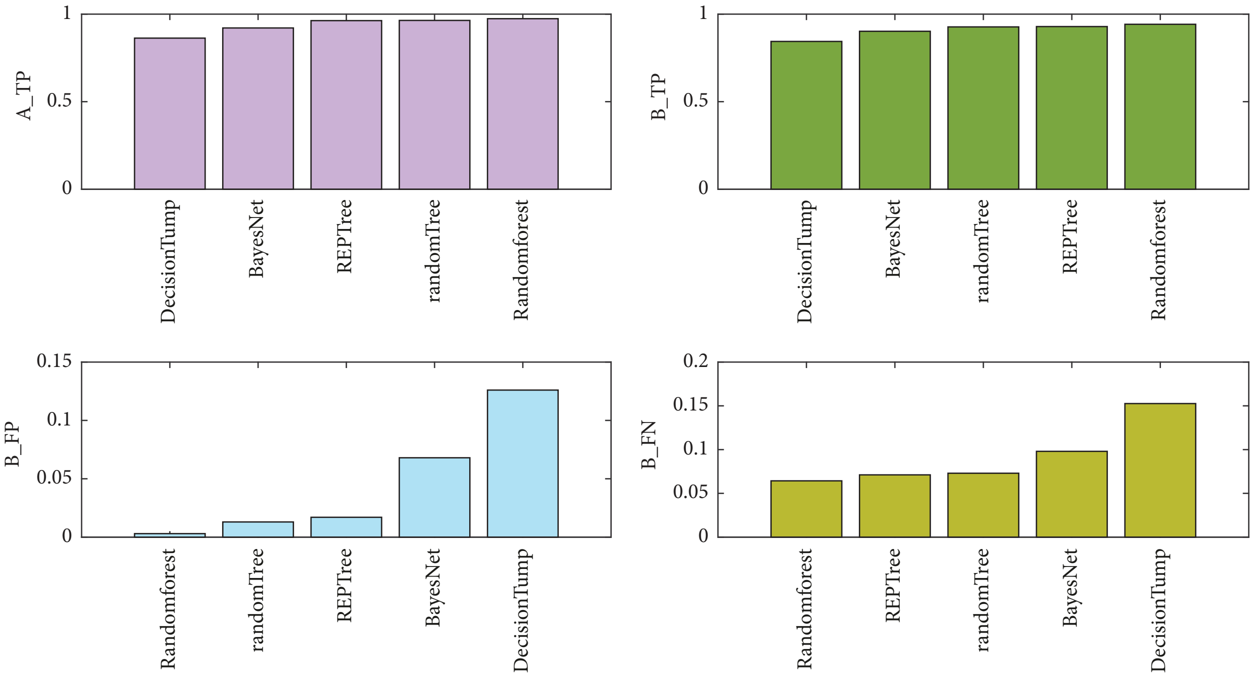


Figure 3: Detection rate of the top five classifiers.

Table 2: Distribution of botnet types in the training dataset.

Botnet name	Type	Portion of dataset
Rbot	IRC, DDoS, US	0.1%
Virut	SPAM, PS, HTTP	0.485%
Menti	PS	3.89%
Sogou	HTTP	0.035%
Murlo	PS	1.64%
Neris	IRC, SPAM, CF, PS	31.3%

Table 3: Distribution of botnet types in the testing dataset.

Botnet name	Type	Portion of dataset
Neris	IRC, SPAM, CF	3.21%
Rbot	IRC, PS, US	2.646%
Rbot	IRC, DDoS, US	0.088%
Virut	SPAM, PS, HTTP	0.4%
Menti	PS	3.33%
Sogou	HTTP	0.036%
Murlo	PS	1.4%
Neris	IRC, SPAM, CF, PS	28.9%
NSIS.ay	P2P	1.71%
Virut	SPAM, PS, HTTP	1.07%

normal traffic generated by benign programs. The detailed contrast tests are done in WEKA, in terms of A_TP, B_TP, B_FP, and B_TN, explained as follows: the ratio of benign traffic and botnet traffic recognized correctly, the ratio of botnet traffic detected as botnet conversation, the ratio of begin traffic classified as botnet traffic, and the ratio of botnet traffic identified as normal traffic. They are defined as follows:

$$\begin{aligned}\text{A\_TP} &= \frac{\text{TP} + \text{TN}}{\text{TM} + \text{TB}};\\ \text{B\_TP} &= \frac{\text{TP}}{\text{TM}};\\ \text{B\_FP} &= \frac{\text{TN}}{\text{TP}};\\ \text{B\_TN} &= \frac{\text{FN}}{\text{TM}},\end{aligned} \tag{3}$$

where true positive (TP) indicates that the number of botnet conversations is correctly classified; true negative (TN) indicates that the number of benign traffic conversations is correctly classified; false positive (FP) expresses that the number of benign traffics is detected as botnet traffic; false negative (FN) indicates that the number of botnet traffics is detected as benign traffic; TM indicates the total number of botnets, and TB expresses the total number of benign traffics.

The experiment result is shown in Figure 3.

The whole recognition rate of DecisionTump is the lowest because there is a one-level decision tree in the DecisionTump. Random forest algorithm selects variables automatically during the model formation and establishes the optimal discriminant model. Thus, the detection rate of random forest algorithm is the highest. Meanwhile, random forest has a lower false positive and false negative rates than the other four. Moreover, there is no obvious difference among the

Table 4: Experimental parameters settings.

	Internal time (s)			
Transmit speed (Gbps)	30		60	
	The number of flows	The number of conversations	The number of flows	The number of conversations
1	138825	39734	203741	61380
10	261630	92933	452930	158722

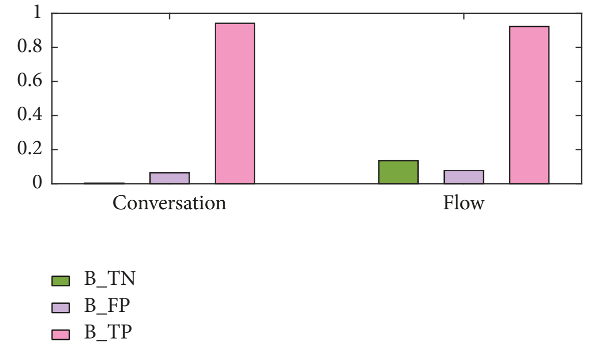


Figure 4: Detection effect of flow-based and conversation-based features.

detection effect of BayesNet, REPTree, and randomTree. The botnet traffic detection of Decision-Tump is 84.4%. However, the detection accuracy of the other four algorithms is more than 90%. The false positive rate and true negative rate of the top five algorithm are under 10% except for DecisionTump.

Kirubavathi and Anitha [20] proposed a botnet detection method via mining of traffic flow characteristics. In their work, they used features like small packets, packet ratio, initial packet length, and bot response packet to identify botnet traffic. Here, we compare the detection rates of flow feature and conversation feature. The result is shown in Figure 4.

As it can be seen from Figure 4, the false positives rate of conversation-based detection and flow-based detection is 0.3% and 13.5%, respectively. Thus, the experimental results show that the false positives rate of our proposed method decreases more than ten times. Meanwhile, the botnet identification rate of our method does not reduce.

In theory, the higher the number of classification trees, the higher the classification accuracy rate. However, if the number and depth of classification tree are extremely high, they will reversely affect the classification speed of classifier. In order to determine the two parameter values of the number and depth of classification tree from random forest algorithm in this paper, we analyze the influence on the classification accuracy by adjusting parameters. In the experiment, the number of classification trees can be set as 10, 50, 100, and 200, and the depth of each classification tree can be set as 2, 4, 10, 20, and so forth. The experiment results of different classification tree size and different classification tree depth are shown in Figure 5.

When the number of the classification trees is 100 and the depth is 10, the detection rate of random forest algorithm reaches the maximum. Afterward, regardless of increasing the number or the depth of the classification trees, the detection rate does not increase anymore. Thus, when the number of the classification trees is set as 100, and the depth of classification tree is set as 10 in the experiment, the random forest works the best.

4.3. Online P2P Botnet Traffic Detection Platform. Our framework has been implemented in Python and utilizes Microsoft Network Monitor to capture packets from a network interface or a pcap file. Because the timeout value of TCP/UDP packets is 60 s, we set the time window as 60 s in this paper to extract conversation feature. While we experimented with different time window settings, the 60-second time window showed the best accuracy at considerably low computational complexity. In the high-speed network environment, we count the number of conversations and the data flows contained in the interval of 60 s and gather the 1 Gbps and 10 Gbps network in many times. The interval of the gathering is 60 s and 30 s, and then we compute the average value. The result is shown in Table 4. In Table 4, T stands for time, S stands for speed, and conversa stands for conversation.

According to Table 4, in the 10 Gbps network, and the interval of 60 s, the average number of passing flows is 452930 and the average number of conversations is only 158722. Thus, using the conversation features can greatly reduce the number of feature vectors. The reason of that is a conversation consists of any number of flows that have the same source/destination host/port and the same protocol. According to the foregoing experiments, we can see that the time of using random forest algorithm to detect 204711 feature vectors is 27.1 seconds. Thus, half real-time botnet detection platform based on random forest classifier and conversation features can identify botnet traffic under the high-speed network environment.

5. Conclusion and Future Work

In this paper, we propose an efficient botnet traffic detection system which can handle heavy network bandwidths. Our framework utilizes PF_RING to solve the high packet drop rate of Libcap. RF-RING has low latency and low overhead to extract required fields of traffic. Then, feature selection is conducted to reduce the dimensionality of data. Conversation features combine the advantages of the existing detection methods based on flow statistical behaviors and flow similarity. We select promising features using random forest algorithm in order to reduce the feature dimension. This

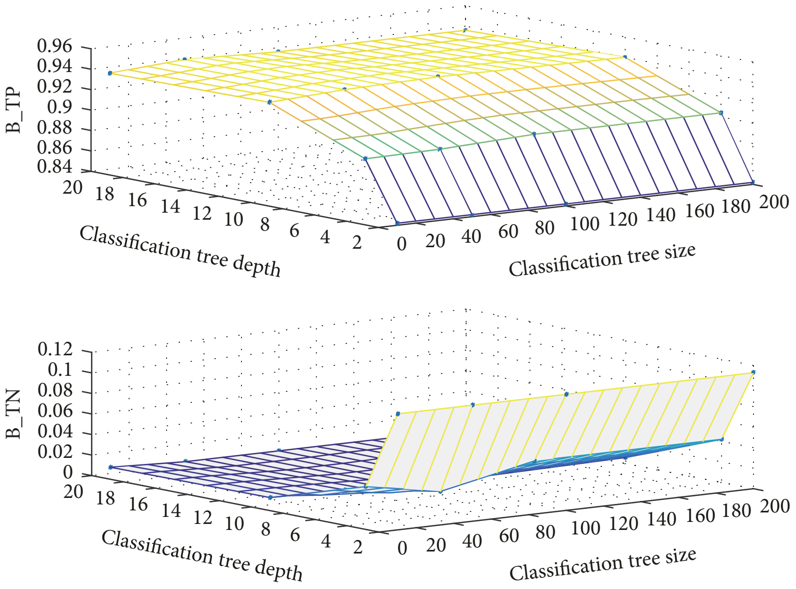


Figure 5: Detection rate for different number and different depth of classification trees.

framework selects the machine learning which obtained the best learning performance. The experiments are conducted on the offline public dataset and online real data. The experimental results show that conversation features used in this paper behave better than flow features in the CTU13 open source dataset. Among all the classification algorithms, the detection rate of random forest is the highest, which is up to 93.6%. And the false alarm rate is only 0.3%, which is ten times less than detection based on traffic flow characteristics.

The future work will focus on mining association rules according to our proposed conversation features. Moreover, we need to further identify specific botnet categories in order to design corresponding defense plans.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant nos. 61572115, 61502086, and 61402080) and the Key Basic Research of Sichuan Province (Grant no. 2016JY0007).

References

[1] Z. Zhu, G. Lu, Y. Chen, Z. J. Fu, P. Roberts, and K. Han, "Botnet research survey," in *Proceedings of the 32nd Annual IEEE International Computer Software and Applications Conference (COMPSAC '08)*, pp. 967–972, IEEE, August 2008.

[2] C. Mazzariello, "IRC traffic analysis for botnet detection," in *Proceedings of the 4th International Conference on Information Assurance and Security (IAS '08)*, pp. 318–323, IEEE, September 2008.

[3] J.-S. Lee, H. C. Jeong, J.-H. Park, M. Kim, and B.-N. Noh, "The activity analysis of malicious http-based botnets using degree of periodic repeatability," in *Proceedings of the International Conference on Security Technology (SECTECH '08)*, pp. 83–86, IEEE, December 2008.

[4] W. Zhou and X. Wu, "Survey of p2p technologies," *Computer Engineering and Design*, vol. 27, no. 1, pp. 76–79, 2006.

[5] H. R. Zeidanloo and A. A. Manaf, "Botnet command and control mechanisms," in *Proceedings of the International Conference on Computer and Electrical Engineering (ICCEE '09)*, pp. 564–568, IEEE, December 2009.

[6] D. Dittrich and S. Dietrich, "P2P as botnet command and control: a deeper insight," in *Proceedings of the 3rd International Conference on Malicious and Unwanted Software (MALWARE '08)*, pp. 41–48, IEEE, October 2008.

[7] M. Feily, A. Shahrestani, and S. Ramadass, "A survey of botnet and botnet detection," in *Proceedings of the 3rd International Conference on Emerging Security Information, Systems and Technologies (SECURWARE '09)*, pp. 268–273, IEEE, June 2009.

[8] R. Villamarín-Salomón and J. C. Brustoloni, "Bayesian bot detection based on DNS traffic similarity," in *Proceedings of the 24th Annual ACM Symposium on Applied Computing (SAC '09)*, pp. 2035–2041, ACM, March 2009.

[9] S. Arshad, M. Abbaspour, M. Kharrazi, and H. Sanatkar, "An anomaly-based botnet detection approach for identifying stealthy botnets," in *Proceedings of the IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE '11)*, pp. 564–569, IEEE, December 2011.

[10] M. N. Sakib and C.-T. Huang, "Using anomaly detection based techniques to detect HTTP-based botnet C&C traffic," in *Proceedings of the IEEE International Conference on Communications (ICC '16)*, pp. 1–6, IEEE, Kuala Lumpur, Malaysia, May 2016.

[11] P. V. Amoli and T. Hämäläinen, "A real time unsupervised NIDS for detecting unknown and encrypted network attacks in high

speed network," in *Proceedings of the 2nd IEEE International Workshop on Measurements and Networking (M & N '13)*, pp. 149–154, IEEE, October 2013.

[12] K. Singh, S. C. Guntuku, A. Thakur, and C. Hota, "Big data analytics framework for peer-to-peer botnet detection using random forests," *Information Sciences*, vol. 278, pp. 488–497, 2014.

[13] S. Kalmegh, "Analysis of WEKA data mining algorithm REPTree, simple CART and RandomTree for classification of Indian news," *International Journal of Innovative Science, Engineering, and Technology*, vol. 2, no. 2, pp. 438–446, 2015.

[14] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software," *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[15] S. Saad, I. Traore, A. Ghorbani et al., "Detecting P2P botnets through network behavior analysis and machine learning," in *Proceedings of the 9th Annual International Conference on Privacy, Security and Trust (PST '11)*, pp. 174–180, IEEE, Montreal, Canada, July 2011.

[16] M. R. Rostami, B. Shanmugam, and N. B. Idris, "Analysis and detection of P2P botnet connections based on node behaviour," in *Proceedings of the World Congress on Information and Communication Technologies (WICT '11)*, pp. 928–933, IEEE, December 2011.

[17] H. Zhang, M. Gharaibeh, S. Thanasoulas, and C. Papadopoulos, "Botdigger: detecting DGA bots in a single network," in *Proceedings of the IEEE International Workshop on Traffic Monitoring and Analaysis*, Louvain La Neuve, Belgium, April 2016.

[18] W. Wang, B.-X. Fang, and X. Cui, "Botnet detecting method based on group-signature filter," *Journal on Communications*, vol. 31, no. 2, pp. 29–35, 2010.

[19] K. Shanthi and D. Seenivasan, "Detection of botnet by analyzing network traffic flow characteristics using open source tools," in *Proceedings of the 9th IEEE International Conference on Intelligent Systems and Control (ISCO '15)*, pp. 1–5, IEEE, January 2015.

[20] G. Kirubavathi and R. Anitha, "Botnet detection via mining of traffic flow characteristics," *Computers and Electrical Engineering*, vol. 50, pp. 91–101, 2016.

[21] J. Zhang, R. Perdisci, W. Lee, X. Luo, and U. Sarfraz, "Building a scalable system for stealthy P2P-botnet detection," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 1, pp. 27–38, 2014.

[22] M. Stevanovic and J. M. Pedersen, "An efficient flow-based botnet detection using supervised machine learning," in *Proceedings of the International Conference on Computing, Networking and Communications (ICNC '14)*, pp. 797–801, IEEE, February 2014.

[23] L. M. Garcia, "Programming with libpcap—sniffing the network from our own application," *Hakin9-Computer Security Magazine*, p. 2-2008, 2008.

[24] M. M. Rathore, A. Ahmad, and A. Paul, "Real time intrusion detection system for ultra-high-speed big data environments," *The Journal of Supercomputing*, vol. 72, no. 9, pp. 3489–3510, 2016.

[25] D. Zhao, I. Traore, B. Sayed et al., "Botnet detection based on traffic behavior analysis and flow intervals," *Computers and Security*, vol. 39, pp. 2–16, 2013.

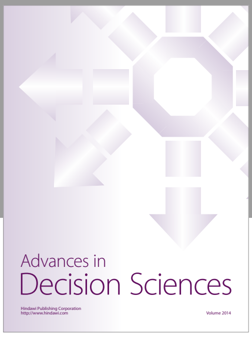
[26] H. Choi, H. Lee, and H. Kim, "BotGAD: detecting botnets by capturing group activities in network traffic," in *Proceedings of the 4th International ICST Conference on Communication System Software and Middleware*, p. 2, ACM, June 2009.

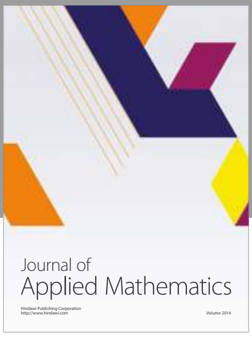
[27] P. Judge, D. Alperovitch, and W. Yang, "Understanding and reversing the profit model of spam (position paper)," in *Proceedings of the 4th Workshop on the Economics of Information Security*, June 2005.

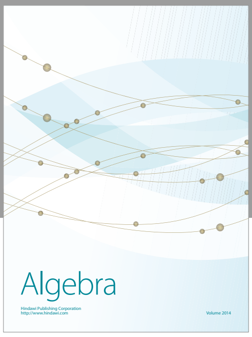
[28] F. Haddadi, D.-T. Phan, and A. N. Zincir-Heywood, "How to choose from different botnet detection systems?" in *Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS '16)*, pp. 1079–1084, IEEE, April 2016.

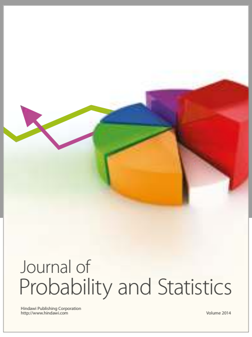
[29] A. Sharma and S. K. Sahay, "An effective approach for classification of advanced malware with high accuracy," *International Journal of Security and Its Applications*, vol. 10, no. 4, pp. 249–266, 2016.

Advances in
Operations Research

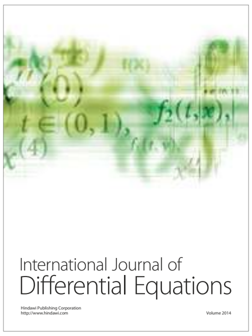

Advances in
Decision Sciences

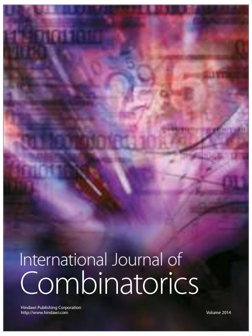

Journal of
Applied Mathematics

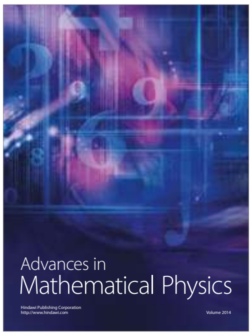

Algebra

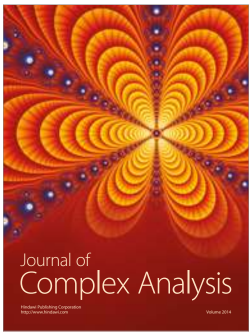

Journal of
Probability and Statistics

The Scientific
World Journal

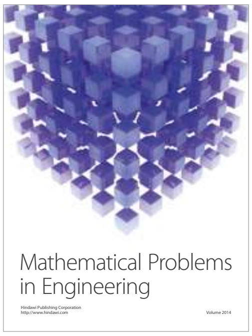

International Journal of
Differential Equations

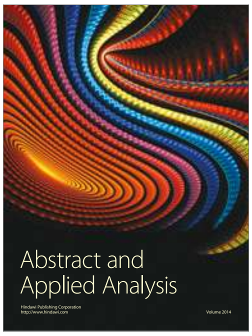

International Journal of
Combinatorics

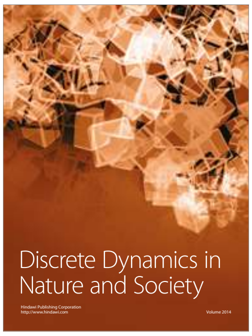

Advances in
Mathematical Physics

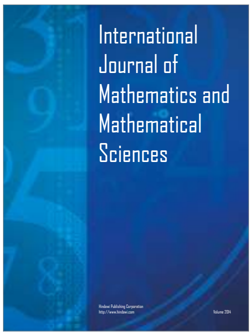

Journal of
Complex Analysis

Journal of
Mathematics


Mathematical Problems
in Engineering

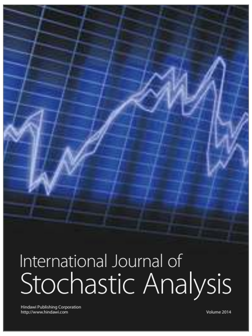

Abstract and
Applied Analysis


Discrete Dynamics in
Nature and Society


International
Journal of
Mathematics and
Mathematical
Sciences

Journal of
Discrete Mathematics

Journal of
Function Spaces


International Journal of
Stochastic Analysis

Journal of
Optimization