International Journal of Future Generation Communication and Networking
Vol.6, No.6 (2013), pp.81-90
http://dx.doi.org/10.14257/ijfgcn.2013.6.6.09

Botnet Detection Based on Degree Distributions of Node Using Data Mining Scheme

Chunyong Yin[1, 2], Lei Yang[1] and Jin Wang[1]

[1]School of Computer & Software, Nanjing University of Information Science & Technology, Nanjing 210044, China
[2]Jiangsu Key Laboratory of Meteorological Observation and Information Processing, Nanjing University of Information Science & Technology, Nanjing 210044, China

Abstract

Botnet is most widespread and occurs commonly in today's cyber attacks and they become one of the most serious threats on the Internet. Most of the existing Botnet detection approaches concentrate only on particular Botnet command and control (C&C) protocols and structures, and can become ineffective as Botnets change their structure and C&C techniques. In this paper, we proposed a new general detection strategy. This proposed strategy was based on degree distributions of node and anomaly net flows, and combined data mining technology. In this scheme, we first constructed accurate traffic profile based on packet behavioral mode, and then introduced dialog flow to draw traffic profile of node. Finally we set up degree distributions of node and group and applied the degree distributions of node as input of data mining, which were then classified and distinguished to obtain reliable results with acceptable accuracy. The advantages of our proposed detection method is that there is no need for prior knowledge of Botnets such as Botnet signature and the accuracy of the experiment results is as much as 99%. The FP rate and the FN rate can be controlled within 3%, the best is almost 0.

***Keywords:** botnet, botnet detection, degree distribution, data mining*

1. Introduction

The growth of botnet attracts a lot of attention from the network security community and botnet becomes a dangerous threat to the Internet. To make distinction between Botnet and other kinds of malware, we have to comprehend the concept of the Botnet. For better understanding of Botnet, we focus on the three entities of botnet at first.

●Bot code: a piece of code to control the others' computers without the owner's permission or being aware of it.

●Bots: compromised hosts in which bot code are embedded, and then can be controlled remotely by attacker.

●Command and control server: a server used to communicate with the bots and command and control them to cooperate many malicious activities.

So the botnet is a collection of compromised hosts (bots) in which bot codes are unnoticed embedded, and then under control of the attacker. Bots are used to send spam emails, host phishing web sites, cooperate in distributed denial of service (DDoS) attacks, and other kinds of malicious activities. Attackers need a command and control (C&C) channel to command the bots and coordinate malicious activities. Most of botnets C&C channels are using IRC (Internet Relay Chat) protocol, in such a case, a IRC server becomes a command and control server.

Several studies have been conducted in order to detect and understand how botnets work. However, there are two essential techniques for Botnet detection: setting up honeynets and passive network traffic monitoring. Many papers discussed about using honeynets for Botnet detection. But we have to take into consideration that honeynets cannot detect bot infection most of the times and is just good for understanding Botnet characteristics. Botnet detection techniques based on passive traffic monitoring have been useful to identify the existence of botnets.There are mainly four techniques base on passive traffic monitoring: signature-based, anomaly-based, DNS-based and mining- based.

One effective technique for botnet detection is to identify botnet C&C traffic. However, botnet C&C traffic is difficult to detect. In fact, since botnets utilize normal protocols for C&C communication, the traffic is similar to normal traffic. Moreover, the C&C traffic is not high volume and does not cause high latency. Therefore, anomaly-based techniques are not useful to identify botnet C&C traffic. Several data mining techniques including machine learning, classification, and clustering can be used efficiently to detect botnet C&C traffic. Many papers also have been conducted to discuss how to use data mining to distinguish flows between the normal internet and botnet. Botminer [15] is the most recent approach which applies data mining techniques for botnet C&C traffic detection. Botminer is an improvement of Botsniffer. It clusters similar communication traffic and similar malicious traffic. Then, it performs cross cluster correlation to identify the hosts that share both similar communication patterns and similar malicious activity patterns. Botminer is an advanced botnet detection tool which is independent of botnet protocol and structure.

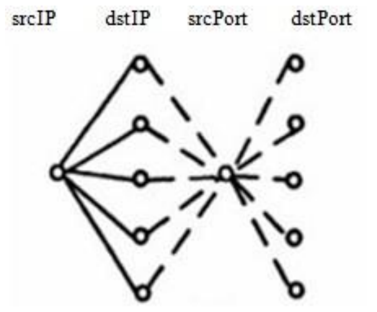
In our paper, we analysisd the anomaly of the botnet and the features of the botnet traffic, summarized the characteristics of the botnet traffic, then three classification schemes, namely J48, naive Bayes and LibSVM were used to build model to detect botnet node by using the characteristics of the botnet traffic.

2. Botnet Abnormal Behavior Analysis

A typical botnet can be created and maintained in five phases including: initial infection, secondary injection, connection, malicious command and control (attack), update and maintenance.

In the initial infection phase, the attacker scans a target subnet for known vulnerability, and infects victim machines through different exploitation methods. We can set up a model to represent botnet behavior of this phase as Figure 1 shows. The model tells that when there is a scanning happening in the network, a port of a node is trying to connect to a lot of ports of different nodes.

When the attacker or botmaster uses the worms to transmit bot program to expand the botnet, it is also will represent a similar one-to-many relationship, just as the follow Figure 2 depicts. Through the model, we can easily get that point that when worm breaks out, the compromised node is trying to infect more normal nodes, connecting to a same port of different normal nodes.



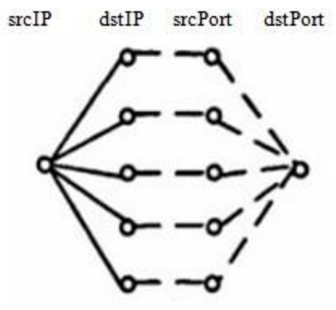


Figure 1. The Model of Scanning **Figure 2. The Model of Worm Transmitting**

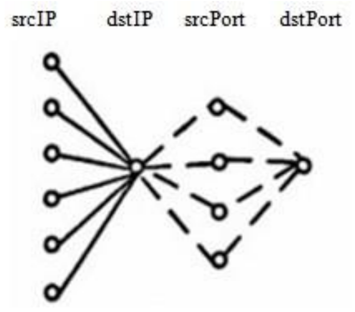


Figure 3. The Model of DDoS

In the initial infection phase, we can find the trail of botnet from the vulnerability scanning and worm transmitting. In connection phase, the bot program joins a command and control (C&C) channel, and connects the zombie to the command and control (C&C) server. Upon the connection of C&C channel, the zombie becomes a part of attacker's botnet army.This phase is most likely to expose the trail of botnet. Just as the previous DNS-based techniques mentioned, in order to access the C&C server bots perform DNS queries to locate the respective C&C server that is typically hosted by a DDNS provider. The domain list generated either by hard-coded or by algorithm is likely to be out of date, or the domain names does not exist at all. So there will be much more abnormal DNS traffic. For example, there may be many packets without responses. In P2P-based botnet, during the connection phase, the bots may access many ip addresses that does not exist, then it will also generate many packets without answers.

After connection phase, the actual botnet command and control activities will be started. The botmaster or attacker uses the C&C channel to disseminate commands to his bot army. Bot programs receive and execute commands sent by botmaster. The C&C channel enables the botmaster to remotely control the action of large number of bots to conduct various illicit activities. DDoS attacks and spam are the two mainly attack means used by botmaster using the botnet. The Figure 3 shows a DDoS attack model. The model shows that when DDoS attack happens, too many source ip addresses send a large volume of packets to only one destination ip address.

The above is the research on the common attacks of the botnet. The models used for analysis represent the relationship between the points and edges. So we introduce the knowledge of graph theory to describe the models with that relationship. According to graph theory, in an undirected graph, the number of edges connecting to a point is the degree of that point. In a directed graph, the number of edges out of a point is the out-degree of that point; the number of edges enter into a point the in-degree of that point. We use the concept of degree to describe the abnormal behavior of a node. In the models of Figure 1, Figure 2, Figure 3, there all is a one-to-many or many-to-one relationship.

The node with source ip is the source node; the node with destination ip the destination node. In the model of Figure 2, a source node sends volume of packets to many destination nodes. In order to represent the out-degree and in-degree of network node, we come up with the concept of the dialog flow and redefine the in-degree and out-degree of the node.

Dialog flow: the packets with the same source ip, the same destination ip, the same source port as well as the same destination port belong to a same dialog flow.

The out-degree of the node: the number of dialog flow coming out of the node is the out-degree of that node.

The in-degree of the node: the number of dialog flows enter into a node is the in-degree of that node.

The out-degree of the source port: the number of the destination ports to which the source node sends dialog flows is the out-degree of the source port.

The in-degree of the destination port: the number of the source ports used by source node sends dialog flows to destination port is the in-degree of the destination port.

We can use the redefinition of degree of the node to express the model of abnormal behavior. In Figure 1, the dialog flows transmit from one source node to many destination nodes, from one source port to many destination ports. The out-degree of the source node and source port is relatively big, the in-degree of the destination node and destination port is small, almost just 1. In Figure 2, for a certain period of time, the out-degree of source code is big, the in-degree zero; the in-degree of destination node is 1 and the out-degree is 0; the out-degree of source port is 1 and the in-degree of destination port is big. In Figure 3, the in-degree of destination node and destination port is very big, the out-degree of them is 0; the in-degree of the destination node and destination port is very big.

By means of the out-degree and in-degree of node, out-degree of source port as well as in-degree of destination port, the common attack models of botnet can be exactly expressed.

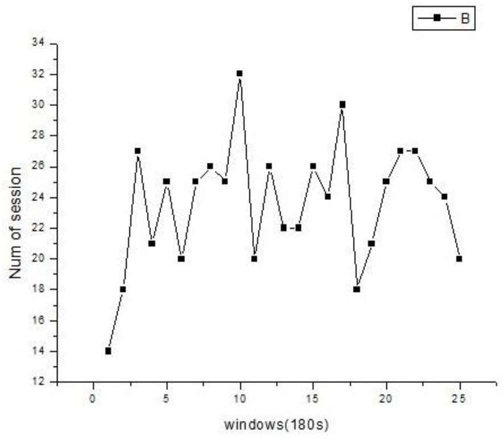
Last phase is to maintain bots lively and update. In this phase, bots are commanded to download an update binary. Bot controllers may need to update their botnets for several reasons. For instance, they may need to update the bot binary to evade detection techniques, or they may intend to add new functionality to their bot army. Moreover, sometimes the updated binary move the bots to a different C&C server. This process is called server migration and it is very useful for botmasters to keep their botnet alive. The characteristics of botnet traffic in this phase are that transmitting packets is few and the network traffic is little. For example, in IRC-based botnets, in order to make sure every bot of the botnet is active, botmaster will send a "PING" command to every bot, the active bot will respond a "PONG" command to botmaster to tell the botmaster it is active. Here we introduce the degree of the dialog flow to represent the volume of traffic.

The degree of dialog flow: The number of the packets involved in a dialog is called the degree of a dialog. If a dialog contains too many packets, we called the degree of the dialog is very deep, Otherwise we say the degree of the dialog is very shallow.

In the maintain phase, if the botmaster send command to every bot to ensure every bot is active, there will be too many dialog flows generated in this phase, and every dialog flow contain a few packets. That is to say the degree of every dialog is very low. This is very different from the normal network traffic.

The Figure 4 describes the characteristic of dialog flows in the normal network traffic. We count dialog flows from the network traffic in every 180 seconds, the vertical axis represents the number of dialog flows, and the horizontal axis shows time windows in 180 seconds.

Seeing from the Figure 4, we can get that the maximum of the dialog flows is below 34. Next we do analysis on the characteristics of the average degree of dialog flow of normal network, the result is shown in Figure 5. The average degree of normal dialog flow is relatively deep. The deepest reaches up to 500 (packets), and the shallowest is almost 100 (packets).So we can get the conclusion, in the normal network traffic, the dialog flows in a time interval, for example 30 seconds or 180 seconds, is very few, but the packets in every dialog is enormous, that is to say, the degree of the dialog flows is very deep.



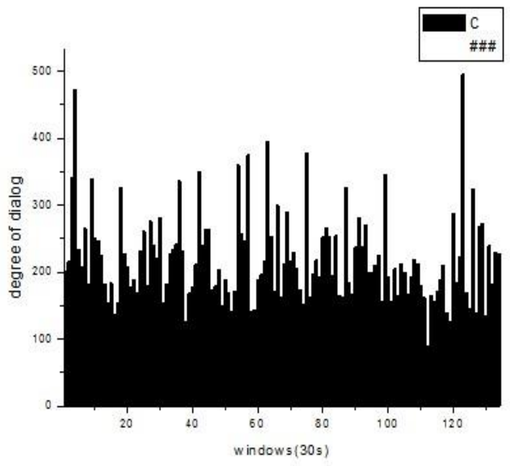
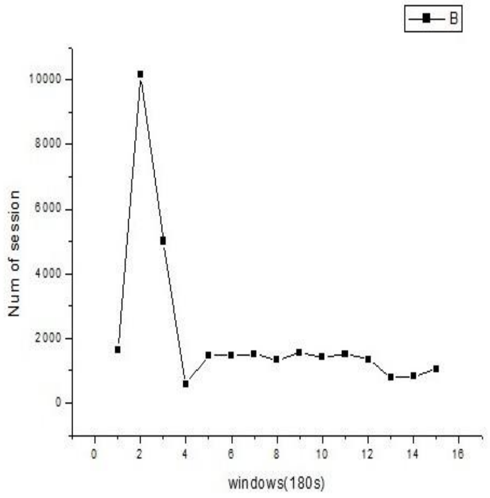


Figure 4. The Dialog Flows of Normal Traffic

Figure 5. The Average Degree of Normal Dialog Flows

But in the botnet, the situation is very different. Figure 6 shows the characteristics of the dialog flows in the botnet traffic. As can be seen from the Figure 6, it is obviously that the dialogs are very many, mostly are above 1000. Just as we can see before, the botmaster want to communication or keep in touch with the bots, he was bound to set up large amount connections to lots of different hosts. Therefore, it generated a mass of dialogs.



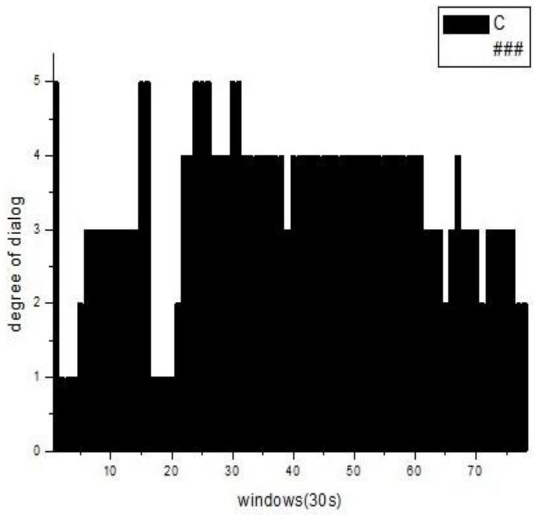


Figure 6. The Dialog Flows of Botnet Traffic

Figure 7. The Average Degree of Botnet Dialog Flows

Figure 7 depicts the average degree of botnet dialog flows. Also as we expect before, in order to keep the privacy of the botnet, botnet communication used data used data at minimum level as much as possible. The deepest average degree of the botnet dialog flows is just 5 (packets).So we can also draw a conclusion for the botnet dialog flows: the number of dialog flows in every time interval is very big, but the average degree of ever dialog flow is very shallow.

In fact, from all the analysis, we can conclude that not only in the maintain phase, but also in the initial infection phase, the connection phase as well as the maintenance phase, the dialog flows of botnet traffic is always much more bigger than the dialog flows of the normal network traffic. So the feature of dialog flow is an important characteristic in detecting botnet or even in detecting abnormal traffic.

3. Attribute Analysis

This module is implemented for finding out abnormal behaviours and traffic from botnets and discovering abnormal flow between botmaster and bots. From the point of degree in graph theory, combining the analysis of the preceding sections, we can represent a node's behavioural traits by a four-tuple, as shown below

F=(out_degree,in_degree,out_sport,out_dport)

●out_degree: the out-degree of a node

●in_degree: the in-degree of a node

●out_sport: the in-degree of destination port

●out_dport: the out-degree of source port

Through the four-tuple, we tell whether there is a scanner or not, whether there is abnormal traffic. But we can't distinguish the botnet traffic from the normal traffic just by the four-tuple. Based on the previous analysis, we introduced an important feature used to detection botnet: the characteristics of dialog flow. But as for a single node, this information is not useful at all. Botnet is a network, all the bots are a group. Bots belonging to the same botnet receive the same commands that causes them having similar netflow characteristics and performing same attacks. There is coordination and correlation among the bots. The node should be put into a group with which it has something to do. Then we can analyze the features of the group, for example, the dialog flows of the group. And here, we introduce a new concept-communications group. Communication group: communications group is a set of nodes that communicate with each other. Based on the analysis above, if there are too many nodes whose out-degree or in-degree is zero in a communications group, the communications group is likely to be a group belonging to a botnet. And second, if there are a large amount of dialog flows and the degree of the dialog flow is very shallow, then the group also may be a botnet group. So we use a four-tuple to represent the communications group traits, too.

G=(group_out0,group_in0,group_dia,avg_dia)

●group_out0: the number of the nodes whose out-degree is zero

●group_in0: the number of the nodes whose in-degree is zero

●group_dia: the number of dialog flows that the communication group contains.

●avg_dia: the average degree of the dialog flows

$$avg\_dia = total\_packets / total\_dia\log s \quad (1)$$

●total_packets: total packets a communication group has

●total_dialogs: total dialog flows a communication group has

We combine the node futures with the communications group futures to get the node personal and group traits. Then we get the ultimate node feature vector, it is a eight-tuple.

B=(F,G)=(out_degree,in_degree,out_sport,out_dport,group_out0,group_in0,group_dia,avg_dia)

4. The Architecture of Our Detection Framework

Figure 8 shows the architecture of our proposed botnet detection system, which consist of 4 main components: Capturing network traffic, Generating dialog flows, Generating feature vectors, classifier. Capturing network traffic is realized by the wireshark [12]. Generating dialog flow module is responsible to convert the raw data captured by wireshark to dialog flows. Generating feature vectors module is responsible to extract the feature vectors from the dialog flows. The whole process goes like this: First, the raw packet data captured by wireshark was passed to dialog flow generator for calculation and statistics and then output the result into a TXT text file. Then, passed the TXT file to features vectors generator to process and output the result into another TXT text file. Second, we need transform the second TXT text file into a CSV text file. Then we passed the CSV file to WEKA for classifying.

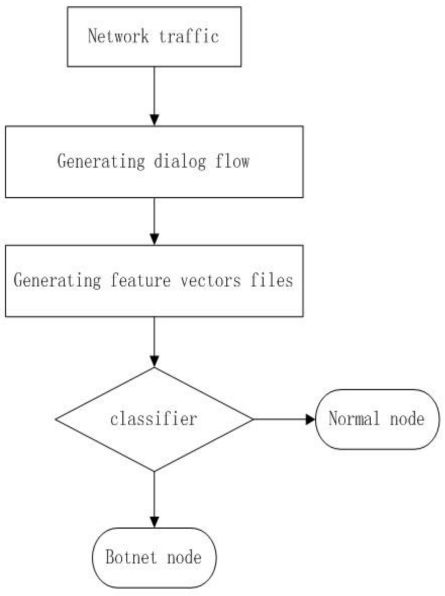


Figure 8. Architecture Overview of our Proposed Detection Framework

5. Result and Discussion

The data was obtained from university of Victoria. We use the false negative counts (FN counts) and false positive counts (FP counts) to evaluate the performance of the classifiers considered. False negative means that botnet nodes were classified into the normal group, and false positive is that botnet nodes were not be identified. So a low FN counts guarantees that only a small fraction of the botnet nodes will be discarded during our botnet identification process. A low FP counts guarantees that the set of nodes identified as botnet node will not be infested by normal nodes.

Table I. Comparison of Three Algorithms

algorithm	accuracy	Accurate counts	FP counts	FN counts	build model time
J48	99.80%	29018	7	49	0.39s
NaiveBa-yes	99.48%	28924	11	139	0.35s
LibSVM	99.46%	28919	100	55	30.6s

Conclusion in Table I were drew after WEKA analyzing experiment data and cross-validation folds of preset test options were selected as 10. Accuracy of three algorithms were respectively 99.8%, 99.48% and 99.46%, among which J48 was proved more appropriate because of higher distinguishing ratio, least FN counts and least FP counts. As NaiveBayes, its FP counts are less than LibSVM's, but its FN counts are more than LibSVM's. The main drawback of LibSVM is that it take too more time to build model than J48 and NaiveBayes. In conclusion, J48 was more appropriate among the three algorithms.

In order to prove that the features and algorithm we selected have certain adaptive and universality, three experiments were done on three different data using J48(TABLE II). Accuracy of the three experiments were respectively 99.8%, 100% and 99.8%. In the first experiment, the false negative rate is 0.173%, and the false positive rate is 0.8%. In the second experiment, the false negative and the false positive is both 0. The false negative of the third is 0.189% and the false positive is 0.2%. The results of three experiments are perfect just as the Table II shows.

Table II. Result of Experiments Using J48

File name	Accuracy	Accurate counts	FP counts	FN counts
1.pcap	99.8%	29018	7	49
2.pcap	100%	30314	0	0
3.pcap	99.8%	30002	3	54

6. Conclusions and Future Work

This paper proposed a botnet detection method relying on degree distribution of node and using data mining technology to analyze network behavior. The method is also not confined to a specific botnet C&C communication protocols and structures, and regardless of the packets that were encrypted. The method is simple and useful. The attributes selected to represent node behavior do a fine job of describing the characteristics of botnet traffic. However, there are still some flaws. Because it takes much time to generate the dialog flow, the method is not performing up to on-line detection. We can only analyze the traffic off-line. We could add some modules into the detection framework.

Acknowledgements

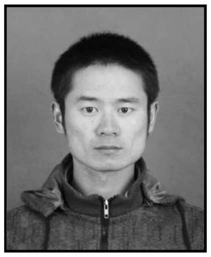
This work was funded by Jiangsu Key Laboratory of Meteorological Observation and Information Processing (No.KDX1105) to Dr. Chunyong Yin and by the National Natural Science Foundation of China (No.61373134). It was also supported by the Industrial Strategic Technology Development Program (10041740) funded by the MOTIE Korea, and by the Natural Science Foundation of Jiangsu Province (No. BK2012461).

References

[1] P. Cooreia, E. Rocha and A. Nogueira, “Statistical Characterization of the Botnets C&C Traffic”, Procedia Technology, vol. 1, **(2012)**.
[2] L. Weimin, M. Chen and L. Fang, “Analysis on the time-domain characteristics of botnets control traffic”, The Journal of China University of Posts and Telecommunications, vol. 18, no. 2, **(2011)**.
[3] W. Jinsong, L. Fan and Z. Jian, “Botnet detecting method based on group-signature filter”, Journal on Communications, vol. 31, no. 2, **(2010)**.
[4] W. WuZuo, “Research on Network Traffic Anomaly Detection Method Using Degree Distributions”, Wuhan: Wuhan University of Sicence and Technology, **(2011)**.
[5] Z. Wei, Y. Chunyang and L. Jianyi, “Botnet domain name detection features analysis base on DNS traffic”, http://www.paper.edu.cn, **(2010)**.
[6] H. R. Zeidanloo, A. Bt Manaf and P. Vahdani, “Botnet detection based on traffic monitoring”, International Conference on Networking and information Technology (ICNIT), IEEE, **(2010)**.
[7] L. Wenhwa and C. Chiaching, “Peer to Peer Botnet Detection Using Data Mining Scheme”, International Conference on Internet Technology and Applications, IEEE, **(2010)**.
[8] C. Chiamei, O. Yahui and T. Yuchou, “Web Botnet Detection Based on Flow Information”, International Computer Symposium (ICS), IEEE, **(2010)**.
[9] S. Arshad, M. Abbaspour and M. Kharrazi, “An anomaly-based botnet detection approach for identify stealthy botnets”, IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE), IEEE, **(2011)**.
[10] M. Feily, A. Shahrestani and S. Ramadass, “A survey of botnet and botnet detection”, Third International Conference on Emerging Security Information, Systems and Technologies. SECURWARE’09, **(2009)**.
[11] C. Livadas, R. Walsh and D. Lapsley, “Using machine learning techniques to identify botnet traffic”, Proceedings 2006 31st IEEE Conference on Local Computer Networks, IEEE, **(2006)**.
[12] J. Goebel and T. Holz, “Rishi: Identify bot contaminated hosts by irc nickname evaluation”, Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets, **(2007)**.
[13] G. Gu, R. Perdisci and J. Zhang, “BotMiner: Clustering Ananlysis of Network Traffic for Protocol- and Structure-Independent Botnet Detection”, USENIX Security Symposium, **(2008)**.
[14] F. Tegeler, X. Fu and G. Vigna, “BotFinder: finding bots in network traffic without deep packet inspection”, Proceedings of the 8th International Conference on Emerging Networking experiments and technologies, ACM, **(2012)**.
[15] G. Gu, P. Porras and V. Yegneswaran, “Bothunter: Detecting malware infection through ids-driven dialog correlation”, Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium, USENIX Association, **(2007)**.

Authors

Chunyong Yin Dr. Chunyong Yin is currently an associate Professor and Dean with the Nanjing University of Information Science & Technology, China. He received his Bachelor (SDUT, China, 1998), Master (GZU, China, 2005), PhD (GZU, 2008) and was Post-doctoral associate (University of New Brunswick, 2010).He has authored or coauthored more than twenty journal and conference papers. His current research interests include privacy preserving and network security.



Lei Yang He received bachelor degree in computer science from the Nanjing institute of technology in 2011 and now studying for his Master Degree in computer science and application in Nanjing University of Information Science & Technology. His research interesting includes computer networks, network security, and data encryption.

Jin Wang Dr. Jin Wang received the B.S. and M.S. degree from Nanjing University of Posts and Telecommunications, China in 2002 and 2005, respectively. He received Ph.D. degree from Kyung Hee University Korea in 2010. Now, he is a professor in the Computer and Software Institute, Nanjing University of Information Science and Technology. He has published more than 120 journal and conference papers. His research interests mainly include routing protocol and algorithm design, performance evaluation and optimization for wireless ad hoc and sensor networks. He is a member of the IEEE and ACM.