A Clustering Approach for Detecting Auto-generated Botnet Domains

Yang Pu, Xiaojun Chen(✉), Yiguo Pu, and JinQiao Shi

Institute of Information Engineering, Chinese Academy of Sciences, Beijing, People's Republic of China
clarissayp@163.com, {chenxiaojun,puyiguo, shijinqiao}@iie.ac.cn

Abstract. Domain fluxing is a general method for botnet operators to control the victims and escape detection. Botnets based on domain fluxing, such as Conficker, Torpig, Kraken, generate a unique list of domain names based on a predefined domain generation algorithm (DGA). If the algorithm is known in advance, it is easy to identify and block botnet traffic. Unfortunately, exploiting details about the algorithm requires reverse-engineering technology and that is not always feasible.

In this paper, we propose a methodology to detect auto-generated domains by measuring the disparity between auto-generated domains and normal domains. The idea is based on the observation that the normal domain names differ from auto-generated domain names in readability, randomness etc., because botnet don't use well-formed words which is highly likely registered. Clustering algorithm is used to group auto-generated domains into several separated clusters and normal domains into other clusters. As shown in the validation and experiment phase, we prove this method can detect DGA domains with high performance.

Keywords: Clustering · Domain fluxing · Botnet domains · Network attack

1 Introduction

Domain fluxing is a useful and general method for botnet operators to control the victims and escape detection simultaneously. In domain fluxing process, bots periodically generate a large number of domain names using a domain generation algorithm (DGA) with a special seed. And then the bots query each of generated domains list until one of them is resolved to a command and control (C&C) server.

Domain fluxing botnets, such as Conficker, Torpig, Kraken, generate a unique list of domain names based on a predefined DGA. Different botnet based on different DGA generates domain names in a different way. For example, Conficker-A [1] bots use the current date and time at UTC (in seconds) as the seed, which is acquired by sending empty HTTP GET queries to a few legitimate sites such as baidu.com or google.com etc. In this way, all bots would generate exactly the same domain names at the same time every day. Torpig [2, 3] bots employ a special way where the seed for the random

W. Niu et al. (Eds.): ATIS 2015, CCIS 557, pp. 269–279, 2015.
DOI: 10.1007/978-3-662-48683-2_24

string generator comes from one of the most popular trending topics in Twitter. Kraken generates specific word which is similar to English language with properly matched vowels and consonants, and then combines each of them with a randomly chosen suffix, such as -able, -dom, -ment, -ship, or -ly.

Security vendors deal with this kind of attacks depending on domain names blacklist and reverse-engineering. They updated the domain names blacklist after running a process of domain discovery and then detected malicious DNS requests using technique similar to signature matching. But botnets employed DGA can dynamically produce a large number of random domain names and select a part of them to use for communication. Security vendors have to understanding the details of DGA relying on reverse-engineering technology and update the blacklist dynamically. This way is time-consuming and resource-intensive, and more is not available always.

So we want to know whether are there efficient method to auto detect the malicious domain names generated algorithmically by botnets?

In this papers, a clustering-based detection approach is proposed to decide if domain names are auto-generated domain names. This approach is based on the following two observations [4, 5]: The first one is that domain names generated by the same algorithm can be similar. Different DGAs use different algorithms and dictionaries to generate domain names, and the others generate domain names in a completely random way. The second one is that normal domain names can be expected to vastly differ from malicious domain names, because botnet don't use well-formed words which is highly likely registered, but non-malicious domain names composed with well-formed words usually because the web-owner wish the web domain is easy to remember.

We extract three kinds of features for the key clustering process. That include readability, entropy and structure features wherein readability describes the difference between the non-malicious domains and malicious domains in 2-gram frequency and one-gram frequency; entropy evaluate the randomness and structure features describe length and compositions of the domain name.

On the validation and experiment phase, three datasets are used that are Non-malicious dataset (non-m-ds), malicious dataset (m-ds) and online-traffic dataset (online-ds). Non-malicious data set is composed of normal domain names and collected from alexa [6]. Malicious data set is the auto-generated domain names collection, and we collected them from conficker (A, B, C) [7] and kraken [8] bonnets etc. Online-traffic dataset is the set of domains collected from real-time DNS traffic in our library network environment during one month. Non-malicious dataset and malicious dataset are used for validate the accuracy of the clustering approach. If the clusters after clustering process consists of elements from non-m-ds or consist of elements from m-ds mainly, that means most domain names with same labels (non-malicious or malicious) are grouped into the same cluster. Online-traffic dataset is used to test the clustering approach on real-time dns traffic. Actually, we can't know the labels of real-time domain names clusters. So two methods can be used to label the real-time domain names clusters. One is using Kullback–Leibler (K–L) divergence to judge every domain cluster whether malicious or non-malicious. K–L divergence computes the "distance" between real-time domain cluster and non-malicious dataset or malicious dataset, if the distance to non-malicious dataset is small, the real-time cluster is judged non-malicious and otherwise. Another method is checking every domain name in one real-time cluster manually. McAfee Site Advisor [9] and Web of Trust [10] can be used to query against domain reputation.

The rest of this paper is organized as follows. In Sect. 2, we compare our work against related literature. In Sect. 3, feature selection process is described which shows difference between malicious domain names and non-malicious ones and then introduce the cluster algorithms. Next, in Sect. 4, validation and experiment results applied to different data sets are demonstrated and discussed. Finally, in Sect. 5 we conclude.

2 Related Work

Detection method of domain-flux have been analyzed by Li and Chen [11]. They observed the differences between malicious domain name and non-malicious domain name. Our work build on this earlier work for detection algorithmically generated domain names used in domain-flux. Zhang et al. [12] specifically introduced domain generated algorithm and proved that domain names generated by same algorithm are similar in the measure of character features. In our work, we know that this similarity can be used to separate malicious domain and non-malicious ones. If we group the domain names which is similar to each other on the measure of character features, we have enough reason to believe that most domain names in this group probably have the same label, malicious or non-commercial.

Choosing features is very vital in our work. If the feature couldn't distinguish malicious domain names and legitimate ones, it won't have good result. Li [13] selected four characteristic of three types to detect Fast-Flux domain name, the proxy distribution, the structure characteristics of domain name and features of service quality. All the features in this paper, instead, are structure characteristics. Features from different type is hard to normalized and determine weight for small different in weight can draw big distinction in result.

Jiang et al. [14] introduce domain generated algorithm technique and domain fluxing. A large number of malicious software by means of the specific domain generation algorithm (DGA: Domain Generation Algorithms), generate a large number of domain names for improving their own controlling of organization, enhancing their ability to survive and prolonging the lifetime of the system. It says that different malicious software, since utilizing different algorithm, generated domain names will show different characteristics. In our work, we expect to find the different characteristics between malicious domain names and good ones instead of finding differences between different algorithm generated domain names.

3 Method of Detection

3.1 Framework

In this section, we present our detection method. First we select features which show differences between algorithmically generated domain names and legitimate ones. Then we use clustering algorithm to group domain names into clusters. Finally, we evaluate the group with KL-Divergence. The framework of this paper shows in Fig. 1.

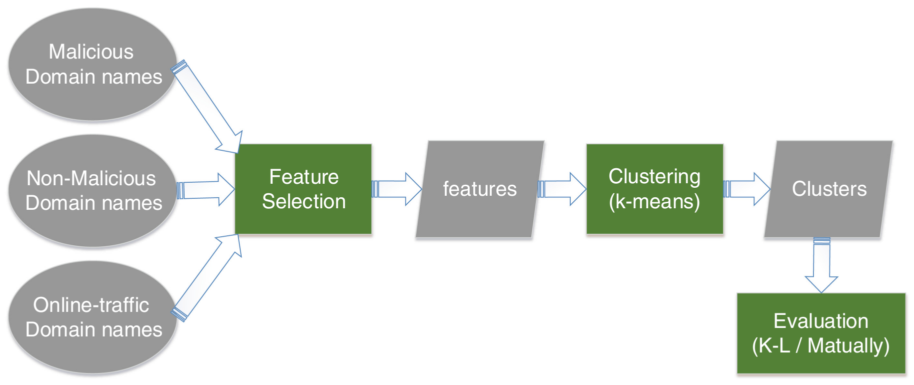

Malicious Domain names
Non-Malicious Domain names
Online-traffic Domain names
Feature Selection
features
Clustering (k-means)
Clusters
Evaluation (K-L / Matually)

Fig. 1. Framework of detection process

3.2 Features Extraction

In this paper, we proposed a botnet detection approach. The detection approach that is based on the observation about similarity of the generated domain in terms of alphanumeric characters.

We chose features based on three observations obtaining from knowledge above:

- Domain-fluxing domain names is a set which frequency of all characters is almost uniform distribution. This factor reflects the specialization of such domain names generated by the algorithm.
- The frequency of different character in normal domain has huge different between each other. But it has a similar statistics frequency of each character in the English text. This reflects that domain name of legitimate domain usually follows the principle of readability.
- In order to ensure the domain name has not been registered, the generated domain name required makes a difference to non-malicious domain name.

So present three kinds of features are picked to represent and describe the characteristic similarity between domain names: readability, entropy and structure features wherein readability describes the difference between the non-malicious domains and malicious domains in bigram frequency and one-gram frequency; entropy features evaluate the randomness and structure features describe other difference from non-malicious domains. To use these three kinds features to the actual Domain-flux botnet detection method, this section will discuss two questions: (a) can we make a distinction between malicious domains and legitimate ones through these three kinds features? And (b) how to get these features?

Readability. As the DGA domains are not generated by human and don't need to be easy remember, they are always clumsy to read with lower readability than legitimate domains. So readability is a good indicator to distinguish malicious domains and non-malicious ones. The lower the readability is, the greater the difference between unknown domain and domain from Alexa. Therefore, this unknown domain is most likely a malicious domain.

As the top domains in Alexa is popular domains where we can get the information about which kind of character combination can make high readable domains. We calculate every n-gram frequency then use this information to evaluate new domains. Firstly the domain is prefixed and suffixed with a '$', then the frequency and probability of every n-gram in the domain of top 1 million in Alexa are computed For a new domain, we get its readability by summing the probabilities of its n-grams as Eq. (1).

$$\text{Score} = \frac{\sum_{levels} \sum_{j} freq(gram_j)}{\sum_{levels} (len + n - 1)} \tag{1}$$

where $gram_j$ means the j-th gram and $freq(gram_j)$ representing frequency of $gram_j$ in top 10000000 domain in Alexa. levels means the level of domain. n is the number of character in the gram. In this paper, we use one gram and two gram to evaluate the readability.

Information Entropy. While readability describes the differences in frequency distribution of n-gram, entropy describes randomness in subdomain. When calculate entropy, the firs level and second-level domain is removed, which belong to the country domain suffix or the general domain suffix. For example, if a domain is google.com.cn, the subdomain here is Google.

Firstly, get the frequency of every character in the subdomain. The final aim is to measure the differences in randomness of characters between non-malicioudains and malicious domains. The higher the entropy is, the greater the difference between unknown domain and legitimate domain. Therefore, this unknown domain is most likely a malicious domain. The entropy is evaluated by Eqs. (2) and (3):

$$\text{E} = -\sum_{i=1,..,|z|} p_i log p_i \tag{2}$$

$$p_i = \frac{count_i}{length} \tag{3}$$

where p_i is the frequency of the i character in subdomain, length is the length of subdomain. $count_i$ is the number that i character shows in subdomain. $|Z|$ is the total number of character which can appear in domain names. In this paper, $|Z|$ is defined as 256.

Structure Features.

Length rate len_rate. As malicious domains are always longer than legitimate domains, we capture the difference by follow equation:

$$\text{LenRate} = \sum_{i=1,..,levels} \frac{len_i}{i} \tag{4}$$

where len_i is the length of the i-the level domain; levels is the number of level in the domain.

TLD rate TtlRate. We know .com is popular domain because it is always authoritative and difficult or expensive to apply. DGA domains, on the other hand, are inclined to choose domains that are low cost and easy to register, such as .info, .cc and so on. So we can statistics the probability of top level domain as one feature. The TLD probability is computed from the top domains in Alexa.

The number of different charactors NumDiffChar. DGA could choose its characters from a self-defined dictionary, which may be different from the dictionary that human use. This feature is calculated as following:

$$\text{NumDiffChar} = \frac{\sum_{i=1,..,levels} \text{NumDiffChar}(domain_i)}{levels} \tag{5}$$

where $NumDiffChar(domain_i)$ is the number of different characters in the i-th level domain.

The maximum length of continuous consonants, vowels and numbers: MaxConCon, MaxConVow, MaxConNum. As malicious domains do not care whether the domain name is easy to remember or not, they may have longer length of continuous consonants, vowels and numbers. We also choose these features to differentiate legitimate domains and malicious ones.

The frequency numbers and vowels: NumRate and VowRate. Legitimate domains always contain enough vowel characters and few number characters to make it simple and comply with English words. Malicious domains do not need obey these rules.

The level of domain Level. The domains generated by same DGA may have same level. We use these feature to cluster similar domains.

3.3 Clustering Algorithm

We use two different clustering algorithm, K-means and MBK-means. The biggest feature of K-means clustering is required to specify the value of k in advance (i.e., the number of clustering). K-means must determine the size of K, and K often cannot be determined by data set in advance.

K-means clustering approach comprises the following two steps:

Step 1: Given an original centroid set of m which is specified randomly
Step 2: Assign each object to the group that has the closest centroid, according to the distance between test point and centroid
Step 3: After all objects have been assigned, recalculate the positions of the *k* centroids. Stop until the result doesn't change after running step 2 and step 3 again

In order to compare the result after different clustering algorithms, we implement our data on MBK-means clustering too.

MBK-means clustering approach comprises the following two steps:

Step 1: it generates k new training sets $D_i(i = 1..k)$ from the given standard training set D

Step 2: Using D_i to run K-means. After obtaining every centroids in every training sets, we combine all centroids and update

3.4 Metric for Labeling Group of Domain Names

This Metric only used for labeling domain names extracted from DNS queries. In this paper we use KL-Divergence with bigrams distribution.

The K–L divergence metric is a non-symmetric measure of divergence between two probability distributions. There are two probability distribution here, P and Q. P represents the test distribution, and Q represents the base distribution from which the metric is computed. The following equation describes the divergence between two distributions P and Q, where i the number of possible values for a discrete random variable [14]

$$D_{kl}(P||Q) = \sum\nolimits_{i-1}^{n} P(i)log\frac{P(i)}{Q(i)} \tag{6}$$

For a given a test distribution q computed for the domain to be tested, non-malicious probability distribution g and malicious distribution q, respectively, we judge the distribution as malicious or not via the following formula:

$$\text{If} \quad D_{sym}(b||q) - D_{sym}(g||q) > 0 \tag{7}$$

Then, distribution q is a non-malicious distribution.

$$\text{If} \quad D_{sym}(b||q) - D_{sym}(g||q) < 0 \tag{8}$$

Then, distribution q is a malicious distribution.

4 Validation and Experiments

4.1 Data Set

In this part, we describe the datasets and the way we obtain them. Three datasets are used on the validation and experiment phases. First is non-malicious data set. This set is composed of the most popular domain names collected from alexa [6]. Second is malicious data set which collected them from Conficker (A, B, C) and kraken bonnets etc. The last one is Online-traffic data set is the set of domains collected from real-time DNS traffic in our library network environment during one month. If we know the domain is malicious or non-malicious before detection, then we can regard this kind of domain names as domain name with labels

This set of domain names with labels obtained from good domain names from Alexa and malicious domain names generated from domain generated algorithms, while the set of domain names without labels of domain is Online-traffic data set.

4.2 Features Evaluation

To test the efficiency of the four feature above, we selected 50 domain names both in Alexa and malicious domain names set. Then we extract features. Choosing three characteristics every time, we painted three-dimensional map. Black points represent a domain from the collection of malicious domains, red from Alexa. In Fig. 2, each axis represent one feature. We can conclude from the following four figure in Fig. 2 draw by Origin 8.0 that feature selection is successful.

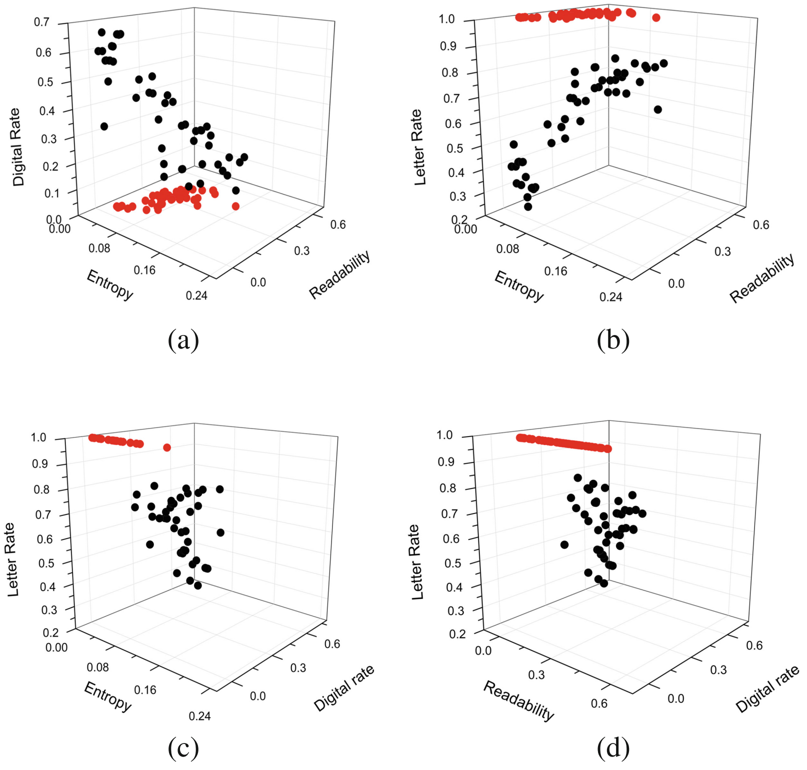


Fig. 2. Features

4.3 Clustering

After preparing the data of features, we implement K-means and MBK-means by means of Python.

Domain Names with Labels. The dataset is 10,000 domain names from Alexa, 15,000 from conficker and 1,000 from kraken. For K-means, MBK-means need to define group number k first, we define k as 2, 5, 10, 20, 40, 50. We expect that the consistent rate goes up as the group number increase. The result prove our prediction Table 1

Table 1. Result of clustering

K	Groups	Total	Alexa	Con-ficker	Kra-ken	Consistance Rate
2	1	123974	97767	26178	29	78.86%
	2	77033	2233	73822	978	97.10%
5	1	42010	332	40902	776	99.21%
	2	47316	39564	7751	1	83.62%
	3	50212	3635	46350	227	92.76%
	4	57518	52518	4997	3	91.31%
	5	3951	3951	0	0	100.00%

The table show part result of k-means clustering. K means the total group number.

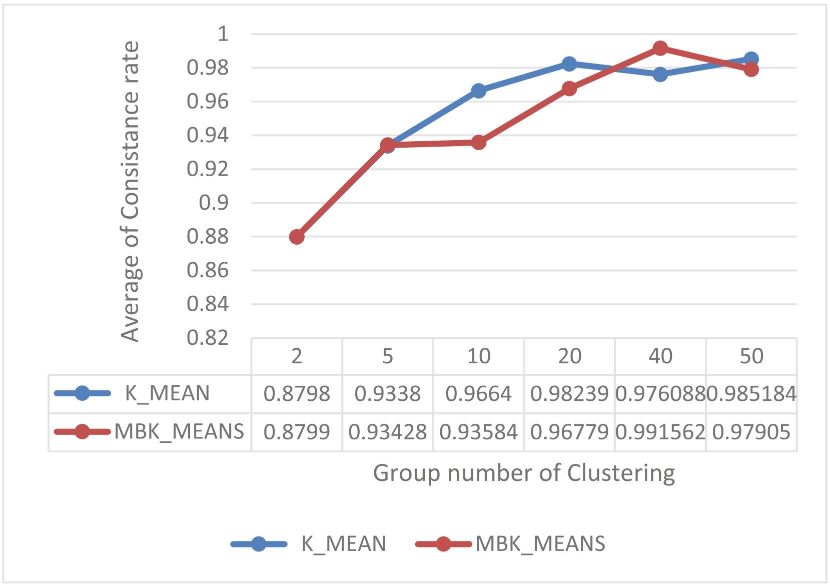
Domain Names Without Labels. The dataset include 103118 domain names collected from real-time DNS traffic.

In the result, some domain names which looks similar in the same group like: as.city8.com, as.com, as.ebz.io, as.eqxiu.com.

4.4 Validation and Evaluation

Domain Names with Labels. For analyzing average of consistence rate in group of different number, we can find out in Table 2 that the consistence rate increase as the group number goes up. Plus, the consistence rate of this method is relatively high. We can conclude that this method is efficient.

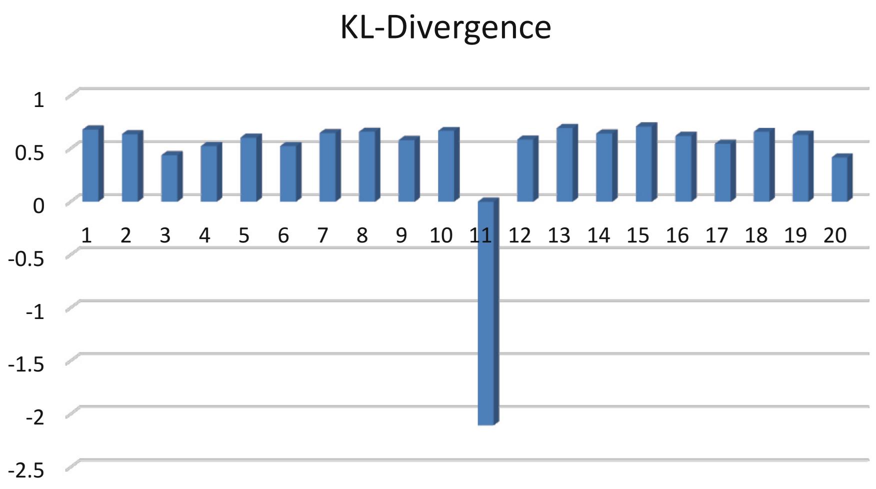
Table 2. Average value of consistence rate



	2	5	10	20	40	50
K_MEAN	0.8798	0.9338	0.9664	0.98239	0.976088	0.985184
MBK_MEANS	0.8799	0.93428	0.93584	0.96779	0.991562	0.97905

Domain Names Without Labels. In this paper, we use KL-Divergence or Internet site to label the group. We cluster 20 groups. Then we use JAVA to calculate K–L divergence. The only point we consider is that if the number is positive or negative. We can see in Table 3, group 11 may be malicious group.

Table 3. KL-Divergence



5 Conclusion

In these paper, we propose a novel method to detect DGA domains, which has following advantages:

(1) Don't need to keep an updated blacklist.
(2) Detect unknown DGAs by analyzing domain names instead of reverse-engineering technology.

It also suffer from some shortages such as that K should be chosen by expert information or search all possible K, which is time consuming work. In the next work, we will consider expand our method on detecting domain names in same type. For example. Domain names of university may be similar with each other, for having the same second level domain, .edu.

References

1. Porras, P., Saidi, H., Yegneswaran, V.: An analysis of conficker's logic and rendezvous points. SRI International Technical report, March 2009
2. Twitter API still attracts hackers. Unmask Parasites blog (2009). http://blog.unmaskparasites.com/2009/12/09/twitterapi-still-attracts-hackers/

3. Stone-Gross, B., Cova, M., Cavallaro, L., Gilbert, B., Szydlowski, M., Kemmerer, R., Kruegel, C., Vigna, G.: Your botnet is my botnet: analysis of a botnet takeover. In: Proceedings of ACM CCS, pp. 635–647, November 2009
4. Antonakakis, M., Perdisci, R., Nadji, Y., Vasiloglou, N., Abu-Nimeh, S., Lee, W., Dagon, D.: From throw-away traffic to bots: detecting the rise of DGA-based malware. In: The 21st USENIX Security Symposium, Bellevue, WA, 8–10 August 2012
5. Yadav, S., Reddy, A.K.K., Reddy, A.L.N., Ranjan, S.: Detecting algorithmically generated domain-flux attacks with DNS traffic analysis. IEEE/ACM Trans. Netw. **20**, 1663–1977 (2012)
6. Alexa: http://www.alexa.com/
7. Universität Bonn. http://net.cs.uni-bonn.de/wg/cs/applications/containing-conficker/
8. Kraken. https://www.damballa.com/downloads/r_pubs/KrakenWhitepaper.pdf
9. McAfee site advisor. http://www.siteadvisor.com
10. Web of trust. http://mywot.com
11. Li, Q., Chen, Z.: Detection of domain-flux Botnet domain names. Comput. Eng. Des. **33**(8), 2915–2919 (2012)
12. Zhang, X., Xu, X., Li, Q.: A real time detect method of malicious domains. Mod. Sci. Technol. Telecommun. **7**(7), 3–8 (2013)
13. Li, X.: The Research on Botnet detection. PLA Information Engineering University, P.R. China, April 2013
14. Jiang, J., Zhuge, J.-W., Duan, H.-X., Wu, J.-P.: Research on Botnet mechanisms and defenses. J. Softw. **23**(1), 82–96 (2012)