Artificial Neural Network Based DGA Botnet Detection

Jiaxuan Wu*

Wuxi Big Bridge Academy, Jiangsu Wuxi, China, 214000

*E-mail: flamingo918@163.com

Abstract. In the recent years, botnet has become a serious threat to network security. As the result, the botnet detection solution is becoming an important topic for network security. DNS request is usually the first step to contact the C&C server of the bots controlled by the bot master and the detection of the DNS request domains is an effective way in detecting the bots. However, most botnets based on DNS protocol adopt Domain Generation Algorithm (DGA), which can change the domain randomly to hide themselves. Therefore, the traditional signature-based approach is rendered ineffective. Compared with the conventional ways of detection, the detection based on machine learning can obtain better detection result. In this work, we propose a botnet detection architecture based on Artificial Neural Network. We implement and evaluate the practicability of this solution with real datasets.

Keywords: Artificial Neural Network, Botnet Detection

1. Introduction

Because of the quickly development of internet, botnet is becoming a more prevailing style of net attacking. According to the Spamhaus Botnet Threat Report 2019, their team blocked 10,263 malware botnet controllers (C&C) hosted on 1,121 different networks in 2018, which is 8 percent more than botnets found in 2017. This kind of destructive network was first emerged in the late 1980s in the Internet Relay Chat[1]. From then on, the botnet is keep developing. In the attack chain of botnet, there are mainly five stages for a typical botnet to form and maintain:

1.1. Initial infection
As the first stage of the botnet lifecycle, the victims are infected in several different ways. For instance, when the victims are fooled into clicking the malicious hyperlinks in the websites or emails, the program will be downloaded and automatically installed onto the victim's computer. Once the botnet program is installed, it will run malicious code to get the root access and turn on simultaneously with the system as the computer is rebooted[2].

1.2. Connection
In the second stage, the botnet program on the victim's computer will try to contact with the command
and control (C&C) servers set up by the attacker. After the bots are connected with the C&C server, the attackers can remotely control the bots. It is in this stage that the botnet masters get control of the victim's computers[3]. Using C&C servers is not the only way of connection, instead the creators of the botnets have multiple ways of communication between botnet master and bots.

1.3. Further expansion
In this stage the attacker will use the bots to enlarge the botnet by infecting more and more bots. One common strategy is to scanning known vulnerabilities of the computers in the same local network, especially those 0day vulnerabilities. If vulnerabilities are found the bots will try to infect that computer to expand the botnet.

1.4. Malicious activities
The botnet master can send malicious orders to the bots with the C&C servers or by P2P protocol. Since botnets are made up a great numbers of controlled computers, any attacks conducted by them will become destructive.

1.5. Maintenance and upgrade
During this phase, the attackers upgrade the botnet programs on the computer he or she controlled from time to time in order to avoid being blocked by the authority[4].

Among these the stages above, the stage 2 in which the bots communicate with the C&C server is the relatively easy monitoring point. If we can confirm the domains are DGA domains we can identify the computer which send the DNS request to be one of the bots controlled by the botnets and take the further step. In the connection stage, a wide variety of the botnets use the domain-generation algorithms (DGA), botnet Mirai is suspected to use the DGA. In DGA-based botnets, the bots can use this algorithm to generate a list of DNS domains, which can be generated by the attacker using the same algorithm on the C&C server. The controllers of the botnets can register some domains in the list as the address of C&C servers. The bots will try to access the list of the domains and finally they will reach the real domains registered by the attackers. The attackers can change the domains used by the C&C servers from time to time to avoid the domains used being blocked by the authority. In addition, algorithm used to generate domains can be changed from time to time through the updates. Which means DGA can nullify many traditional computer security techinics like blacklist blocking and attribute network behavior detection. Therefore, the usage of DGA can increase the cybersecurity force's difficulty of stopping the botnet. One study about blacklisting by Marc Kührer and his team members have shown that only one blacklist among all the blacklists they tested can provide sufficient protection against the DGA botnets they tested[5].

In order to resolve this problem, we proposed the use of Artificial Neural Network in botnet detection. Compared with existing works, the usage of machine learning technology can provide a more accurate detection result after training. We implement a prototype to evaluate this solution. With real-world dataset of DGA domains from 360 network security lab, it is shown that the system can achieve an accuracy of more than 95 percent of distinguishing the DGA domains from normal domains after the model is trained.

The main contribution of our work is that we evaluate and compare the difference between the traditional Naive Bayesian Model and Artificial Neural Network model, the statistical features and the feature of task with N-gram model. Moreover, we built a practical system which can be deployed and
help detecting the DGA domains\cite{6}.

2. Related work

To avoid being detected by internet security authorities, the most common technic used by the cyber criminals is the Domain Generating Algorithm (DGA). First used by the botnet named Kraken, DGA became more famous by the spread of conficker-A, which generates 250 domains every three hours to make it difficult for security experts to find the real domain of the server used by the attacker to communicate with the bots. The main reason of using DGA is that malwares which use the static IP addresses can be quickly blocked by the authority thus the botnet masters cannot control their botnets since they are not able to use their C&C server to communicate with all the bots they controlled. So, instead of setting up totally new servers, the attackers will just change the IP address of their server from time to time to avoid being blocked. As a result, they will register a domain name of the server and then they can change the IP address without changing the domain. In order to hide from detection, the attackers use the DGA to generate multiple domains and change from time to time. Signature-based detection, Anomaly-based Detection and DNS-based Detection are the most common three ways of detection. Signature-based detection mainly uses the already known signatures of the malicious botnets to detect the botnets. This kind of detection can only react on those already known botnet types. For the new botnets, signature-based detection will not work. So, in order to use this way to detect the botnet, we need to maintain an up-to-date database of the malicious botnet's different signatures and this is hard to achieve. Anomaly detection is based on finding the communicating behavior of the hosts, then use them to distinguish the botnets. Unlike the signature-based detection, anomaly-based detection can be used to detect the new botnets. The anomaly detection mainly includes the analysis of the network anomalies such as the high latency of network, the high volumes of traffic and also usage of special ports to communicate, the unusual function of the system behaviors. However, this kind of anomaly detection is not able to detect those botnets which have not yet used to conducted attacks will not be detected by the anomaly-based detection since there are no anomalies in traffic of these hidden botnets. In order to connect to the C&C server, the bots will send DNS queries to locate the C&C server and this is typically hosted by the DNS provider. Therefore, it is workable for us to monitor the DNS traffic and detect DNS traffic anomalies. Those domains have abnormally high and concentrated for short time query rate are likely to be the domain of botnet server. However, this kind of detection will be dodged if the botnet master use DGA and the bots will perform queries to many fake domains\cite{7}.

3. System design and details

3.1. The design of the system

To make it easier to update some particular parts in the system, we divide the whole system into different modules. Our system is mainly made up of two main parts: the domain detecting part and the domain analysis part. In this section, we will talk about the function and realization of each modules. The design of the whole system is shown\cite{8}. 


Figure 1. Design module
3.1.1. The traffic capture module. This module is mainly established on libpcap, which is an open source library provides a high-level interface for traffic capture systems. We use the promiscuous mode of the network card to sniffer all the packages pass though. With this module we can capture the traffic from the bots to the local network servers.

3.1.2. The decoding module. In this module, we will analysis the packages we caught by the traffic capture module we mentioned before. As the only traffic we need to monitor are those DNS requests, we need to filter the other traffic. We analysis the package header of traffic to distinguish the DNS traffic from others. We define structures of the ethernet, IP, UDP, DNS protocols. Each of these structures include information of different protocols and each aspect contained in the protocols' package header is given a different integer or string variable. Some of the aspects such as the flag, need to be defined with the fixed length integer. Moreover, all these variables are defined in the same order as their counterparts of data in the real packages. The next step we do is to assign the proper places of these packages to the structures' pointers according to the length of the package headers. For example, we will assign the package contents from certain place to the pointer of the DNS package. The place from which to start is determined by the size of the ethernet protocol header, IP protocol header, and UDP header, the contents within the range of the size of these irrelevant protocols will not be include in the content assigned to the pointer of the DNS package structure[9].

3.1.3. The domain name extraction module. In this module we extract the names of the domains from the package we captured. As we have already determined the identity of the packages in the decoding module (3.1.2) and filter all the other packages which do not contain the DNS protocol. Then we will read the name of the domain in the content of the packages according to the length of the package header. After getting the domain names, we vectorized these names in order to be handled by the machine learning model which we will mention next.

3.1.4. The machine learning module. To have a usable machine learning module, we must train the module with enough data which have different characteristics. After the machine learning module is trained with sufficient date, it can be used to determine whether the domain inside the packages are normal domains or the DGA domains. In our system we mainly tested the Naive Bayesian Model and the Artificial Neural Network (ANN) Model trained with the normal features of the DGA or the N-gram Feature of the DGA.More details of these machine learning models and features we tested will be included in 3.2.

3.2. Details of the different models and features we tested
In this section, we will discuss the different models in the machine learning module. We mainly focus on the features of these models and the mathematic bases of these models.

3.2.1. Features of DGA. We can use features of DGA to effectively distinguish DGA domains from normal ones because they have many differences linguistically. For example, the DGA domains are more likely to have less vowels and less unique chars, and tend to use more numbers in the domain. Overall, we distinguish the normal domains from the DGA domains according to the percent of vowel, the percentage of unique chars and also the length of the domains.

3.2.2. The N-gram feature. N-gram is a language model, which makes a presupposition that all the appearance of words depends on the words which have already appeared before the word.While calculating feature of DGA date by the N-gram model, we are finding the probability of each alphabet or other numbers or symbols showing in the domain. When a domain is put into the model, we will get the probability of its existence. In our system, the N-gram model mainly serves to help get feature of our date.
3.2.3. *The naive bayesian model.* Naive Bayesian Model makes two assumptions which makes it a good model suitable for machine training. First, all the predictive attributes are independent. Second, there is no hidden or latent attributes which may also influence the prediction.

Then we can simply use the Bayes' rule to calculate the probability with the vector of each predictive attributes. This model is relatively fast while training and predicting compared to other machine learning models.

3.2.4. *The artificial neural network (ANN) model.* This model is also called the Multilayer Perceptron (MLP) model, as a bionic network, it has a number of hidden layers. All of the hidden layers will affect the result of the model. Moreover, each layers inside the Multilayer Perceptron model have an activation function. (shown in Figure 2) Activation function(θ) is a function which serves to filter the useless information for the model. So, only when the date is inside the domain of the layer's activation function, the layer will start to work. One Multilayer Perceptron network (shown in Figure 3) is made up of mainly three layers: the input layer, the hidden layer and the output layer. There can be more than one neuron in each layers, and there can be more than one hidden layer in the middle. The result of prediction depends on all these neurons together. Each of these neuron can be adjusted individually while training, thus the whole system can produce more accurate prediction result.

![Figure 2. Architecture of a single Neuron](image)

![Figure 3. Artificial neural network model](image)

4. Experimental evaluation

4.1. *The datasets used in the experiment*

To train these different machine learning models we use the dataset of the DGA domains released by the 360 net lab and the top one million domains by Alexa. Since the more date we use the better result we will get, but we need to have some date to test, as a result we will use sixty percent of all the date to train our model and the rest to test the accuracy of our detection.

4.2. *Ways used to measure the effectiveness of models*

Among all the ways to determine the effectiveness of different prediction models, we mainly focused on the confusion matrix, precision, recall, and the f1 score of these models in our system.
4.3. Result of our experiment

The final result of the experiment is shown by Table 1. From these date we can find our new model with the Artificial Neural Network can distinguish DGA domains better than traditional models based on Naive Bayesian model, as more domains are treated correctly by the Artificial Neural Network than its opponent. Additionally, the use of N-gram model as the feature of the domains can increase the accuracy of more than ten percent.

| feature& NB | precision | recall | f1 score | confusion matrix |
|-------------|-----------|--------|----------|------------------|
|             | 0.79      | 0.78   | 0.77     | 42596 5494       |
|             |           |        |          | 16092 31811      |
| feature& ANN| 0.85      | 0.85   | 0.85     | 43933 4157        |
|             |           |        |          | 7966 39937       |
| N-gram& NB  | 0.86      | 0.85   | 0.95     | 44511 3472        |
|             |           |        |          | 10879 37131      |
| N-gram& ANN | 0.96      | 0.96   | 0.96     | 46450 1561        |
|             |           |        |          | 2656 45326       |

5. Conclusion and future works

We propose a DNS domain-based botnet detection solution in this work. We implement and evaluate the system. The proposed solution can help those online security guards to monitor the network traffic to distinguish the DGA domains among the normal domains. There still exist some issues of this system. For example, it can only distinguish the DGA domains in the traffic, but it cannot trace back to find the true IP address of the C&C server. Moreover, this system cannot stop those botnets which do not utilize C&C server and DNS protocol, such as the P2P based botnets. In the future, we are going to study the ways to detect the P2P botnets and to determine the real addresses of the C&C servers. Additionally, there are more different machine learning models which we may compare their performance in the system to make the prediction more precise.

References

[1] A survey of botnet and botnet detection[C]/2009 Third International Conference on Emerging Security Information, Systems and Technologies. IEEE, 2009: 268-273.

[2] Spamhaus Malware Labs. Spamhaus Botnet Threat Report 2019 https://www.spamhaus.org/news/article/793/spamhaus-botnet-threat-report-2019

[3] Abusix. A Brief History of Bots and How They've Shaped The Internet Today https://www.abusix.com/blog/a-brief-history-of-bots-and-how-theyve-shaped-the-internet-today.

[4] Garcia L M. Programming with libpcap-sniffing the network from our own application[J]. Hakin9-Computer Security Magazine, 2008, 2: 2008.

[5] John G H, Langley P. Estimating continuous distributions in Bayesian classifiers[C]/Proceedings of the Eleventh conference on Uncertainty in artificial intelligence. Morgan Kaufmann
Publishers Inc., 1995: 338-345.

[6] Zhang H. The optimality of naive Bayes[J]. AA, 2004, 1(2): 3.

[7] Basile A, Dwyer G, Medvedeva M, et al. N-gram: New groningen author-profiling model[J]. arXiv preprint arXiv:1707.03764, 2017.

[8] Guresen E, Kayakutlu G, Daim T U. Using artificial neural network models in stock market index prediction[J]. Expert Systems with Applications, 2011, 38(8): 10389-10397.

[9] Sonawane M S. A Survey of Botnet and Botnet Detection Methods[J]. International Journal of Engineering Research & Technology, 2018, 7(12).