Detection Method of Fast Flux Service Network Based on Decision Tree Algorithm

Jiajia Wang*, Yu Chen

1 School of Information Technology, Taizhou Polytechnic College, Taizhou, Jiangsu, 225300, China

*Corresponding author’s e-mail: wangjiajia_99@163.com

Abstract. Fast Flux service network is the main threats to network security at present. By quickly changing IP address, the location of servers can be hidden and agents can be provided for botnet. In this paper, a detection method of fast flux service network is proposed, which narrows the detection range through pretreatment, then constructs the detection features, and accurately distinguishes FFSN from legitimate network. Finally, decision tree algorithm is used to obtain the ideal results.

1. Introduction

Botnet is a collection of captured hosts, and botmasters use command and control infrastructure to hold these captured hosts. Of all malicious acts, the easiest to be exploited is botnet-based proxy services, which can redirect normally users to malicious, illegal content. This strategy ensures the high anonymity of criminal behavior and makes malicious content easier to be used and centralized management. But botnet is made up of hosts around the world, with a variety of resources and network connections, at any time offline is also expected. In order to provide stable and reliable services in the unstable connection botnet, botmaster must replace the offline IP address in malicious domain name record at any time. Therefore, a FF(fast flux) domain name system technology emerges as the times require. FF technology can quickly replace the mapping between domain name and IP address. By recycling the address in the address pool and constantly replenishing the new IP through query. Botnet can realize that the result of each malicious domain name resolution is the available online IP addresses, which ensures high reliability and stability. Botnet supported by fast flux technology has become the main threat in the network, and how to eliminate it is an urgent problem to be solved.

Fast flux technology is also used in CDN(Content Distribution Network) and RRDNS(Round Robin Domain Name System), so CDN and RRDNS have some similar characteristics with FFSN, which makes it difficult to detect FFSN accurately.

The structure of this paper is as follows: section 2 describes the related work in the current field; section 3 abstracts the relevant features of data packets according to the characteristics of FFSN; section 4 summarizes the detection flow of FFSN; section 5 is the detection algorithm used in this paper, a good algorithm can further improve the detection efficiency; section 6 is experiment and results, and finally, the prospect.

2. Related Work

Salusky[1] proposed the Honeynet project, which included an overview of the FF attack technology, and described how the botnet is hidden from the single FF and the double FF mechanisms. The former can
realize the fast conversion of the DNS resolution record of A type, and the latter can realize the double fast conversion of the DNS resolution record of A type and the DNS resolution record of NS type. There are currently a lot of work to detect FFSN and distinguish FFSN from CDN and RRDNS. Most of these required the active detection of the DNS data stream and allowed a domain name to correspond to many IP addresses, as in the literature [2-4]. These methods simplified the detection of FFSN, but had to resolve the domain name, including the domain name associated with the malicious activity. The monitoring of the corresponding network behavior would result in a large workload, which is not conducive to the busy network equipment.

Part of the FFSN detection work is based on passive DNS packet analysis. Perdisci[3] analyzed the DNS data in the whole network, which was a considerable workload, and then abstracted the relevant attributes for analysis. Berger[6] and Stevanovic[7] both used DNSMap tools to distinguish the diagrams composed of all domain names and related IP addresses. Undifferentiated analysis of all domain names and IP addresses would overburden the original busy network.

Soltanaghaei[8] proposed a passive detection method of DNS data stream based on historical data, and achieved ideal results in the experiment. Using historical empirical data can reduce the burden of detection algorithm, but the storage needs a lot of space, which is still a challenge to the server.

## 3. FFSN Detection Characteristics

Because the end user has to access the server through IP address, in FFSN, the IP address corresponding to the malicious domain name should be available as much as possible, could not be forged, and the legal network would not fake the IP address, too. Therefore, it is feasible to use IP address as a judgment parameter.

### 3.1. IP address distribution $d_{IP}$

Legal network domain names are often distributed on stable and reliable servers, with limited number and little change. FFSN is inclined to use global terminals (more often less protected personal hosts) for domain name hopping. These hosts are scattered and distributed without a fixed geographical limit. The distribution attribute of IP address can be used to distinguish legitimate network from FFSN. First, collect the set of IP addresses returned by a single domain name $\{X_1, X_2, ..., X_n\}$, where $n$ is the number of IP addresses returned. In this paper, we define the distribution of IP address as $d_{IP} = \sum_{i=2}^{n} X_i - X_{i-1} / n$. If the IP address is in a limited range, then $d_{IP}$ tends to zero, and if the geographical location of IP address is scattered, then $d_{IP}$ is always a large value.

### 3.2. Changes in the IP address set $IP_c$

The IP address of the legal network is within a limited range and is highly available, even if to be replaced, the other that already among the existing address pools tends to be selected. FFSN can be offline at any time due to the instability of personal host. The change of IP address set is $IP_c = \frac{IP_{cset}}{IP_{set}}$.

$IP_{set}$ is the existing address set for domain name resolution and $IP_{cset}$ is the address set after domain name resolution. The $IP_{cset}$ of legal domain name is always smaller than that of $IP_{set}$, so $IP_c$ tends to 0 as time goes by. Because a large number of new IP are added constantly, $IP_{cset}$ is always larger than $IP_{set}$, so $IP_c$ of FFSN is a large value.

### 3.3. Number of networks to which IP belongs $N_{IP}$

Compared with the legitimate network, the host used by FFSN is widely distributed in different networks, so the number of network segments resolved by FFSN domain name is much larger than that resolved by legitimate network domain name[9].

The above three points are used in our FFSN detection method, the detection feature $F = \{d_{IP}, IP_c, N_{IP}\}$.
4. FFSN Detection Flow
The detection flow covered in this paper is shown in figure 1.

```
pretreatment

DNS data source

Port=53 and
Protocol=UDP

Yes

forward

No

Blacklist item

Yes

discard

No

Whitelist item

Yes

Feature extraction

Detection and calculation

Classification results

Figure 1. Detection flow chart
```

The detection process of this paper mainly includes two parts: pretreatment and specific detection. The pretreatment part is mainly used to extract network data, filter non-DNS packets, trusted domain names and known malicious domain names.

By checking port 53 and the transport layer protocol type, most of the non-DNS packets in the network can be filtered, lighten the burden on the detection system.

Blacklist is consisited of www.malwaredomains.com website published information, whitelist is consisited of www.Alexa.com website published information. If network data and blacklist item match, then directly filter, do not detect. If network data and whitelist item match, then forward directly, do not detect. The use of blacklist and whitelist can further reduce the burden of detection system and improve the detection efficiency. The remaining packets will enter the specific detection section.

Firstly, the feature vectors entering the specific detection section of the packet are extracted, and then the C4.5 algorithm mentioned in section 5 is used to detect, and finally the results are obtained.

5. FFSN Detection Algorithm
Using decision tree to deal with classification problem is divided into two steps usually: the first step is to form a decision tree classification model by training set learning; the second step is to use the generated decision tree model to classify samples with unknown types. In this paper, the most widely used C4.5 decision tree algorithm is selected, which selects the test attributes according to the information gain rate, which is described as follows:

It is assumed that the network data sample set $S = \{X_1, X_2, ..., X_n\}$, where each sample may be represented by a property vector containing the m-term network flow properties $(A_1, ..., A_m)^T$, it is assumed that the class attribute $A_m$ has k different values, then according to different values of $A_m$, the sample set $S$ may be divided into k subset $C_1, C_2, ..., C_k$, whereby the average amount of information for the classification of the sample set $S$ can be obtained: $H(S) = -\sum_{p=1}^{k} p(C_p) \log_2 p(C_p)$, where $p(C_p) = |C_p|/|S|(1 \leq p \leq k)$.

The construction process of decision tree is the process of gradually decreasing the post-division uncertainty. Taking the arbitrary discrete attribute $A_i(1 \leq i \leq m - 1)$ as an example, assuming that there are t different values $a_q(1 \leq q \leq t)$ in $A_i$, it is not only possible to divide the S into t subset $S_1, S_2, ..., S_t$, according to the value of $A_i$, but also divided the k subset $C_1, C_2, ..., C_k$ into a subset of k*t, each of which $C_{pq}$ represents a set of samples belonging to the p class under the condition of $A_i = a_q$.
where $1 \leq p \leq k$, $1 \leq q \leq t$. Thus, the discrete non-class attribute $A_i$ is selected for division. The average information of the sample set $S$ is $H(S/A_i) = -\sum_{p=1}^{k} p(C_q) \left[ -\sum_{q=1}^{t} p(C_{pq}) \log_2 p(C_{pq}) \right]$, where $p(C_q) = \sum_{p=1}^{k} |C_{pq}| / |S|$, $p(C_{pq}) = |C_{pq}| / |S|$, then the amount of information gain divided by $A_i$ to $S$ is: $f_G(S, A_i) = H(S) - H(S/A_i)$.

Because the information gain rate of using attribute $A_i$ to divide $S$ is equal to the ratio of information gain to segmented information, then we can get: $f_{GR}(S, A_i) = f_G(S, A_i) / f_{sp}(S, A_i)$, where the segmented information $f_{sp}(S, A_i) = -\sum_{l=1}^{t} (|S_l| / |S|) \log_2 (|S_l| / |S|)$. By selecting the attribute with the maximum information gain rate as the test attribute, the C4.5 decision tree method completes the decision tree building process from top to bottom. In order to remove the branch anomalies caused by noise points and isolated points, the C4.5 decision tree method uses the remaining samples in the training data set to pruning the initial generated decision tree, and then obtains the final C4.5 decision tree.

6. Experiments and Results

6.1. Training dataset and experimental dataset

This paper selects training data from www.Alexa.com site and www.malwaredomains.com site, selects 5000 legitimate domain names from www.Alexa.com site, divides them into 5 groups as training data, and selects 2000 malicious domain names from www.malwaredomains.com website, which are divided into 5 groups as training data. Five groups of training data $\{D_{T1}, D_{T2}, D_{T3}, D_{T4}, D_{T5}\}$ include 1000 legitimate domain names and 400 malicious domain names, respectively. The training data does not need to go through the pretreatment stage, directly enter the specific detection stage for feature extraction calculation, the training time of 5 groups of data is about 4h in total. The experimental data are derived from the real operation data of campus network. This paper uses the domain name data collected by campus network within 30 days as the experimental data, and the specific description is shown in Table 1.

| Data Name | Statistics Collected Within 30 Days |
|-----------|-----------------------------------|
| No of hosts | 392 |
| No of A type DNS item | 4Mb |

6.2. Experiments

This article uses the Windows 7 operating system to run the detection algorithm. The experimental data were divided into 5 groups, the time interval of each test was 10 min, and the whole experiment took about 8 hours. The results of 5 groups of experiments are shown in Table 2, 3, 4, 5 and 6.

| Type | Domain Name |
|------|-------------|
| Malicious domain names that are correctly identified | autosegurancabrasil.com |
| | tonyyeo.com |
| | dicrophani.com |
| | airtyrant.com |
| | and so on |
| Malicious domain names that are identified as legal domain names | amazon.co.uk.security-check.ga |
| Legitimate domain names that are identified as malicious domain names | ai.sodoos.com |
| | mzkong.com |
| | tqzyb.com |
| | qtravel.cn |

Table 3. Results of the second experiment
Malicious domain names that are correctly identified
autosegurancabrasil.com
tonyyeo.com
dicrophani.com
airtyrant.com
and so on

Malicious domain names that are identified as legal domain names
amazon.co.uk.security-check.ga

Legitimate domain names that are identified as malicious domain names
ai.sodoos.com
mzkong.com
tqzyb.com

Table 4. Results of the third experiment

Malicious domain names that are correctly identified
autosegurancabrasil.com
tonyyeo.com
dicrophani.com
airtyrant.com
and so on

Malicious domain names that are identified as legal domain names
none

Legitimate domain names that are identified as malicious domain names
ai.sodoos.com
mzkong.com

Table 5. Results of the fourth experiment

Malicious domain names that are correctly identified
autosegurancabrasil.com
tonyyeo.com
dicrophani.com
airtyrant.com
and so on

Malicious domain names that are identified as legal domain names
none

Legitimate domain names that are identified as malicious domain names
ai.sodoos.com
mzkong.com

Table 6. Results of the fifth experiment

Malicious domain names that are correctly identified
autosegurancabrasil.com
tonyyeo.com
dicrophani.com
airtyrant.com
and so on

Malicious domain names that are identified as legal domain names
none

Legitimate domain names that are identified as malicious domain names
ai.sodoos.com
mzkong.com

The following parameters are mainly considered in the experimental process, as shown in Table 7. The experimental data are tested and the results are shown in Table 8.
### Table 7. Main parameter

| Parameter | Description                              |
|-----------|------------------------------------------|
| $T_P$     | Number of FFSN correctly identified       |
| $F_N$     | Number of FFSN identified as legitimate domain names |
| $F_P$     | Number of legitimate domain names identified as FFSN |
| $R$       | Detection rate \( R = \frac{T_P}{T_p + F_N} \) |
| $P$       | Correct rate \( P = \frac{T_p}{T_P + F_P} \) |

### Table 8. Experimental result

| No | $T_P$ | $F_N$ | $F_P$ | $R$ | $P$   |
|----|-------|-------|-------|-----|-------|
| 1  | 46    | 1     | 4     | 97.9\% | 92.0\% |
| 2  | 46    | 1     | 3     | 97.9\% | 93.9\% |
| 3  | 47    | 0     | 2     | 100\%  | 95.9\% |
| 4  | 47    | 0     | 2     | 100\%  | 95.9\% |
| 5  | 47    | 0     | 2     | 100\%  | 95.9\% |

From the analysis of the above, it can be seen that the detection rate and correct rate of this method will be optimized as time goes by, the better results can be achieved. The computational complexity is related to the number of leaf nodes in C4.5 algorithm, in this paper, the number of leaf nodes is 2, and the complexity is small. The IP address of every packet collected needed to be calculated in reference [10], if the more complex algorithm was used, the computational complexity was obviously increased, however if the simpler algorithm was used, the number of Bloom Filter conflicts, that was, the false alarm rate, would increase obviously. Reference [10] needed to store a large amount of historical data for comparison and consume a lot of storage space. The distribution of IP addresses per packet needed to be analyzed in reference [11], which was a matter of high complexity in itself, and some countries do not allow to query the address distribution. Reference [11] also required a lot of storage space to track every IP address. Therefore, the computational complexity and spatial complexity of our method are small, and suitable for practical application.

### 7. The Prospect

In this paper, a FFSN detection method is proposed and verified, which uses several types of detection features. In this paper, the packet is pretreatment and only the DNS packet is detected. Filtering known types of domain names with blacklist and whitelist can reduce the burden of detection system, select three simple but effective detection features, and use C4.5 decision tree algorithm for FFSN detection. The experimental results show that the proposed method can effectively identify FFSN, and distinguish it from other legal types of networks, and the detection rate and correct rate are ideal. The next job is to study a more accurate detection system.

### Acknowledgments

This work was financially supported by Taizhou Polytechnic College Boshuo Fund(TZYBS-17-6), and Taizhou Polytechnic College Students Innovation and Entrepreneurship Training Program (YJDC2018017).

### References

[1] Salusky, W., Danford, R. (2007) Know your enemy: fast-flux service networks. The Honeynet Project. pp. 1-24

[2] Chahal, P.S., Khurana, S.S. (2016) Temprr: application of stucture dependent intelligent classifier for fast flux domain detection. International Journal of Computer Network and Information Security 8(10), 37.

[3] Jiang, C.B., Li, J.S. (2017). Exploring global IP-usage patterns in fast-flux service networks. JCP 12(4), 371-379

[4] Lin, H.T., Lin, Y.Y., Chiang, J.W. (2013). Genetic-based real-time fast-flux service networks detection. Computer Networks 57(2), 501-513
[5] Perdisci, R., Corona, I., Giacinto, G. (2012). Early detection of malicious flux networks ia large-scale passive DNS traffic analysis. IEEE Transactions on Dependable and Secure Computing 9(5), 714-726

[6] Berger, A., DAlconzo, A., Gansterer, W.N., Pescap’e, A. (2016). Mining agile DNS traffic using graph analysis for cybercrime detection. Computer Networks 100, 28-44

[7] Stevanovic, M., Pedersen, J.M., DAlconzo, A., Ruehrup, S. (2017). A method for identifying compromised clients based on DNS traffic analysis. International Journal of Information Security 16(2), 115-132

[8] Soltanaghaei, E., Kharrazi, M. (2015). Detection of fast-flux botnets through DNS traffic analysis. Scientia Iranica. Transaction D, Computer Science & Engineering, Electrical 22(6), 2389

[9] Zhou, S. (2015). A survey on fast-flux attacks. Information Security Journal: A Global Perspective 24(4-6), 79-97

[10] Soltanaghaei E , Kharrazi M .(2015). Detection of Fast-Flux Botnets Through DNS Traffic Analysis[J]. Scientia Iranica, 22(6).

[11] Lombardo P , Saeli S , Bisio F , et al. (2018). Fast Flux Service Network Detection via Data Mining on Passive DNS Traffic[J].