Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 17 (2013) 852 – 860

Procedia Computer Science

Information Technology and Quantitative Management

(ITQM2013)

A Bigram Based Real Time DNS Tunnel Detection Approach

Cheng Qi[a,c,*], Xiaojun Chen[b,c], Cui Xu[d], Jinqiao Shi[a], Peipeng Liu[b,c]

[a]*National Engineering Laboratory for Information Security Technologies, IIE, Beijing, 100193*
[b]*Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190*
[c]*Graduate University of Chinese Academy of Sciences, Beijing, 100049*
[d]*China Academy of Aerospace Aerodynamics*

Abstract

DNS (Domain Name System) tunnels can provide high-bandwidth covert channels that pose a significant risk to sensitive information inside the company networks. Sensitive data are embedded in DNS query and response packets to exfiltrate and infiltrate the network boundaries. However, traditional Intrusion Detection Systems (IDS) and Firewalls let DNS packets pass without any checking. This paper explores a novel approach to detect in real time whether a DNS packet is in a tunnel by scoring the query domain based on bigram. Experiment shows that the bigrams of domains follow Zipf's law whereas tunnelled traffic is obedient to random distribution. The score mechanism in detecting DNS tunnels is proved to be usable theoretically and is confirmed in the experiment. Our approach can get a high accuracy of 98.74% and low false positive of 1.24%.

Selection and peer-review under responsibility of the organizers of the 2013 International Conference on Information Technology and Quantitative Management

Keywords: DNS Tunnel; Zipf' law; N-Gram; Score Mechanisim

1. Introduction

DNS is a hierarchical distributed system resolving domain names to IP addresses. In recent years, more and more network heretics exploit DNS to conduct illegal activities for it is poorly monitored and typically allowed passed network boundaries [1]. For example, P2P botnets fast-flux the IP and domain to avoid detection and increase resilience [2]; DNS tunnels are used as covert channel to transfer sensitive information by malicious users. A lot of DNS tunnel tools such as DNScat, TCP-over-DNS, Iodine and Ozyman are available on the Internet [3]. Many methods have been proposed detecting DNS tunnels, but there are still massive gaps remaining in detecting them in real time and effectively.

Figure 1 shows the working mechanism of DNS tunnels. Usually many channels between malicious insider and Internet are blocked by firewall or intrusion expect for HTTP, DNS and SMTP etc. The DNScat server is outside of the organization and is the authority server of domain, for example, 'example.com'. Then all query domains like '*.example.com' will be sent to the DNScat server. The DNScat client inside the organization encodes sensitive data in the DNS queries in base32 or base64 format, and generates query domains ended with '.example.com'. Then it

* *Email address: sanqi322@gmail.com*

Selection and peer-review under responsibility of the organizers of the 2013 International Conference on Information Technology and Quantitative Management
doi:10.1016/j.procs.2013.05.109

sends these packets to the local DNS server. The local DNS server cannot resolve the domain, it will recursively forward the DNS packet to the up-level DNS server， finally the DNS query will be routed to the DNScat Server. The DNScat server decodes the client's data from the domain name and processes it, and then it encodes the result data in the response packets and sends it back to the client. So the malicious user finishes an illegal communication [4].

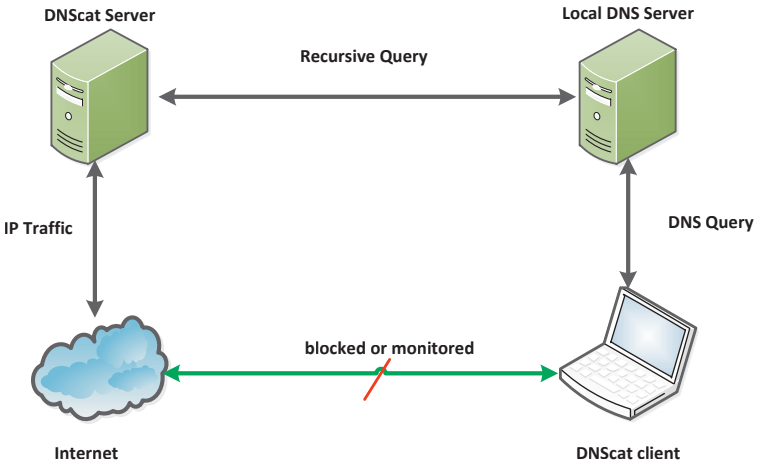

Figure 1 DNS Tunnel workflow

Some features can distinguish DNS tunnels from conventional packets. Those include DNS traffic volume, the length of domain in DNS queries and the randomness of the domain name. However, DNS traffic volume is a statistic feature so it cannot detect DNS tunnels in real time; DNS tunnels such as DNScat can set the length of domain names to escape detection based on domain's length; Existing method uses Shannon Entropy [5] to identify the randomness of a domain name for detecting DNS tunnel, but it maybe treat normal domains such as 'baidu.com' as an abnormal case as it could get a high entropy.

In this paper, we propose such an approach that detects DNS tunnels based on bigram in real time. The key features of the approach are as follows:

- Real-time: we use bigram character frequency to identify the randomness of a monitored domain name and the feature can be detected and calculated in real time. So our approach is real time.
- Effective: DNS tunnel domains are obedient to random distribution. Our approach uses the frequency of bigram to identify the randomness of the domain names to decrease false positive. Experiment shows the approach can get 98.74% accuracy and less than 1.24% false positive.

The remainder of this paper is organized as follows. In section 2, we discuss existing DNS tunnel tools and existing solutions for detecting DNS Tunnels. In section 3, we introduce our score mechanism and prove the effectiveness of the approach. Section 4 introduces the details of system implementation, and then evaluates the efficiency of our approach in experiments. Section 5 gives a conclusion and discusses some future improvement of our approach.

2. Background and Related Work

To the best of our knowledge, existing DNS Tunnels includes time-interval based tunnels and data-embedded based tunnels [6]. The former one use the interval time of packets as one bit to exfiltrate or infiltrate information. The capacity of time-interval based tunnels is very small, so the related tools are very few. Our target focuses on detecting data-embedded based tunnels.

Due to the growing proliferation of DNS tunnel tools, many research activities have been focused on network flow analysis and randomness detection of domain. Existing network flow analysis methods detect DNS tunnels by

analyzing DNS channel volume and packet size features [7]. The feature of DNS channel volume needs a period of time to detect so it cannot give the alarm in real time [8]. Wenke Lee proposed to use Shannon Entropy to test the randomness of domain in Next-Generation DNS Monitoring Tools on Cyber Security Division 2012 Principal Investigators' Meeting. But this scheme cannot resolve the problem of high false positive that some normal domain names such as 'baidu.com' have high Shannon Entropy since the theory ignoring different characters having different frequencies and longer domains having higher entropies.

Kenton Born and Dr. David Gustafson proposed using character frequency analysis to detect DNS tunnels and their approach is the most related to ours [9]. Both the approaches are based on that normal domain names follow Zipf's laws just like natural languages and DNS tunnel domain names are obedient to random distribution. Their approach detects DNS tunnels by analyzing a lot of domains such as a local area network's character frequencies distribution and comparing the distribution to the character frequency fingerprint of legitimate domain traffic. The key idea of our approach is the score mechanism that based on bigram character frequency which can distinguish normal domain names and random ones. The main difference from ours is that their approach uses statistical feature and cannot indicate the malicious DNS channel in real time. Furthermore, their approach will lose efficacy when the background network flow is big enough. While our approach proposed in this paper can detect DNS tunnels in real time. Even only one DNS tunnel packet passes our system, we can detect it at a high probability.

3. Score Mechanism

The score mechanism is the key idea of our approach and its performance determine the result of our approach. In this section we will introduce the score mechanism, and then prove the effectiveness under certain assumption.

3.1. Character Frequency Analysis

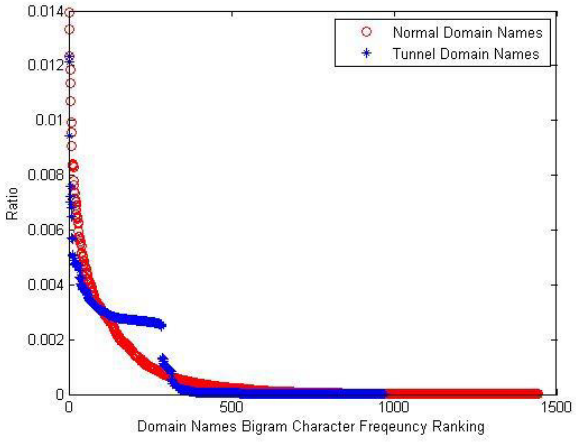

Figure 2 Domain Names Bigram Character Frequency Distribution

The character frequency analysis is the foundation of our approach and the method has successfully been used in cipher text detection [10]. Normal domain names are strings carefully selected by humans and should be recognizable and memorable by humans. Therefore domain names closely follow the natural language characteristics or at least follow Zipf's law which means most normal domain names character frequency will concentrate on a small part of high frequency characters. While DNS tunnel domain names are embedded in data and the embedded data are encoded with Base32 or Base64 to present as a normal DNS packet, they are obedient to random distribution the same as cipher text.

The bigram character frequency distributions of normal domain names and DNS tunnel domain names are shown in Figure 2.The dataset of normal domain names is collected from Alexa [11] that publish one million most popular

website and we use the top 100,000 websites' domain names. The dataset of DNS tunnel are produced by DNScat. Bigram character frequency of normal domain names approximate the curve of function f(x)=1/x and they absolutely follow Zipf's law. The character frequency of DNS tunnel domain names contains three stages. The first one is a small part of sharp declining with high frequency stage for the domain names produced by DNScat having tags of 'dnscat' and 'tcp-over-dns'. The second and last stages are uniform distributed but the last one has a much lower frequency because of DNScat encoding schema using less numbers. Therefore DNS tunnels domain names closely follow random distribution.

3.2. Score Mechanism Design

The approach we propose use the expect value of bigram character frequency as the score of a domain, as

$$Score = \frac{\sum_{levels}\sum_{j} freq\left(gram_j\right)}{\sum_{levels}(len+n-1)} \quad (1)$$

Where $gram_j$ is the j_{th} gram of one level of a domain name, *freq(g)* is the frequency of the gram *g*, *levels* is the $level_{th}$ of the domain name, *len* is the length of a single level domain, *n* is the meta number of gram. In this paper, we use bigram to calculate the character frequency, so *n* is 2.

Let's take 'baidu.com' for example. We omit the top level domain (TLD) such as 'com' because they don't provide much information about a domain. Then it is a domain of one level. All of the bigrams are '$b', 'ba', 'ai', 'id', 'du', 'u$'. The next is to get the frequencies of these bigrams from a bigram character frequency table generated in the off-line training stage. Finally we sum these frequencies and divide it by the number of bigrams to calculate the score of a domain.

3.3. Score Mechanism Analysis

The character frequencies of normal domains follow Zipf's laws [9] and we verify it in our examination. That is

$$freq(i) = \frac{freq(1)}{i} \quad (2)$$

Where *i* is the rank of bigram ordered by the character frequencies descending, *freq(i)* is the character frequency of i_{th} bigram.

We assume that the character frequency of DNS tunnel domain names follow random distribution. Then the character frequency values follow uniform distribution. So we can get their expect value namely their average scores:

$$\overline{Score_{random}} = \frac{\sum_{i} freq(i)}{\sum_{i} 1} \quad (3)$$

Where *freq(i)* is the frequency of the i_{th} bigram.

Whereas normal domain names follow Zipf's laws, most of normal domain names bigram will locate the small part of high character frequencies. So the expected value of normal domains scores will be a little higher. It is

$$\overline{Score_{nomal}} = \sum_{i} freq^2(i) \tag{4}$$

The characters allowed in a domain name are a subset of the ASCII character set, and includes the characters *a* through *z*, *A* through *Z*, digits *0* through *9*, and the *hyphen* [12]. We use '*$*' to indicate the start and end of a domain. So the bigram number is 64*64=4096. Then we can calculate that $\overline{Score_{random}}$ is 0.000251952 and $\overline{Score_{nomal}}$ is 0.0209344. The latter is 83 times of the former. Therefore we can conclude that the score mechanism can be used to distinguish DNS tunnels and conventional channels.

4. System & Evaluation

The system includes two parts: Off-Line Training Mode and On-Line Classifying Mode. As shown in Figure 3 , character frequency table and classifier are produced in the Off-Line Training part and are used to classify in the On-Line Classifying part.

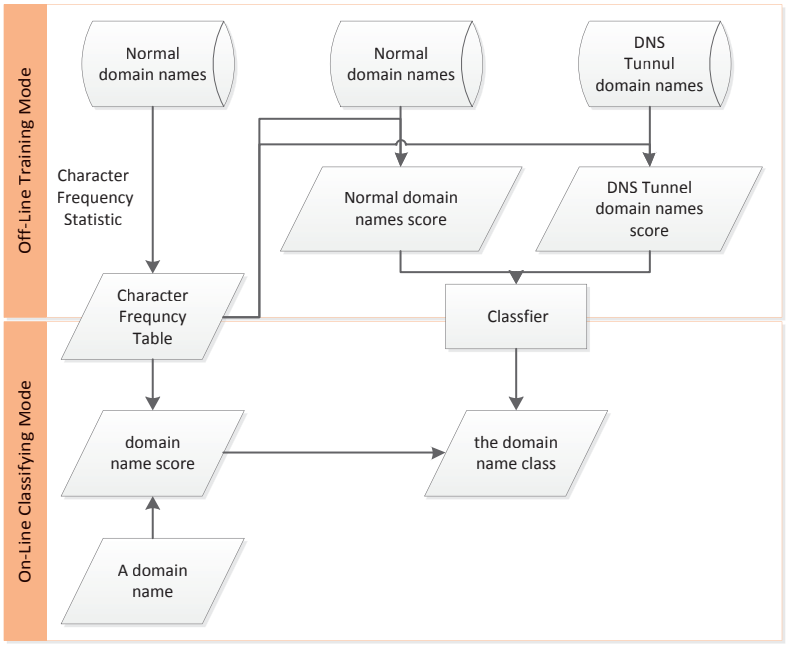

Figure 3 System Overview

4.1. Data Collection

In the evaluation experiment we used 5 data sets: the normal domain names used to get character frequency table, the normal domain names for off-line training, DNS Tunnel domain names for off-line training, the normal domain names used for evaluation and the DNS Tunnel domain names used for evaluation.

The data sets are from two resources. All the normal domain names are from the one million domain names published by Alexa. The first normal domain names are top 100,000, the second ones are the top 10,000 and the last ones are the 10,001~20,000. All the DNS tunnels domain names are produced by DNScat and one half is used as

training data and the left half is used as evaluation data.

4.2. Off-Line Training Mode

The Off-Line Training Mode contains four steps.

- Preprocessing

In this step we extract each level domain and ignore the top level domain which may confuses the classifier. For example, 'news.google.com' will be split into 'news' and 'google'.

- Character Frequency Statistic

After the preprocessing of the top 100,000 normal domain names published by Alexa, each bigram appeared in the domains counts once. Then we can calculate the character frequency of each bigram to form a character frequency table. The top 10 bigram characters in character frequency table are list in Table 1. For example, if all domains are just 'news' and 'google', then '$n', 'ne', 'ew', 'ws', 's$', '$g', 'go', 'oo', 'og', 'gl', 'le', 'e$' are all of the bigrams and the character frequency table is each bigram frequency equaling 1/12.

Table 1 Top 10 bigram character in character frequency table

Ranking	Bigram	Occurrence Number	Frequency
1	s,$	14692	0.013936
2	i,n	14033	0.013311
3	$,c	12997	0.012328
4	e,$	12478	0.011836
5	e,r	11993	0.011376
6	c,o	11305	0.010723
7	a,n	10479	0.00994
8	r,e	10460	0.009922
9	a,r	10089	0.00957
10	e,s	9580	0.009087

- Score Calculating

In this step we first preprocess the normal domain names and DNS tunnel domain names used for training. And then we use the character frequency table mentioned above to calculate the score of each domain name with the score mechanism described in subsection 3.2.

Table 2 DNS Tunnel domain names scores V.S. Normal domain names scores for training data

	DNS Tunnel domain names scores	Normal domain names scores
min	0.0013	4.3700e-004
max	0.0029	0.1743
mean	0.0019	0.0303
median	0.0018	0.0239
mode	0.0018	0.0504
standard deviation	2.4148e-004	0.0232
range	0.0016	0.1739

- Classifier Training

After calculating the score of normal domain names and DNS tunnel ones, we get the distribution of them and calculate the threshold which will make the classifier produce the least false positive. Normal domain name score

distribution and DNS tunnel domain name score distribution are shown in Figure 4. Some parameters about the two datasets are shown in Table 2. The scores of DNS tunnel domain names are focus on a small range with a low score while the scores of normal domain names have a wide distribution. The mean number and medium number of the former is much lower than the latter's. Both the figure and table show that points of DNS tunnel domain name scores are so dense and the points of normal domain name scores is sparse relatively. The relationship between the threshold of score and accuracy is shown in Table 3. Therefore the threshold of score is set 0.0027 to get the highest accuracy.

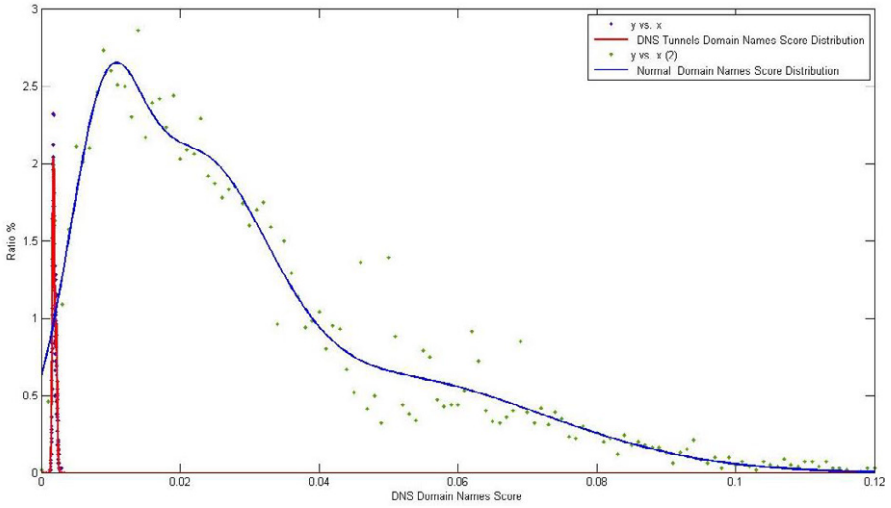

Figure 4 Normal domain name score distribution VS DNS tunnel domain name score distribution

Table 3 the threshold of the classifier selecting

Threshold Score	False Positive	False Negative	Accuracy
0.0025	0.0106	0.0037	0.9857
0.0026	0.0111	0.0015	0.9874
0.0027	0.0118	0.0006	0.9875
0.0028	0.0124	0.0001	0.9874
0.0029	0.0131	0	0.9869

4.3. On-Line Classifying Mode

The workflow of on-line classifying is shown in the below half part of Figure 3. We first do the same preprocessing as do in the off-line training part. Next we calculate the score of a monitored domain names with the same character frequency table. Then we compare the score with the threshold and finally classify the domain name.

4.4. Evaluation Results

In this subsection, we present the experiment results of our evaluation. We show that the approach can distinguish DNS Tunnel domain names and normal ones in real time. The executing time of classifying 20,000 domain names is shown in Table 4 and the experiment is done on a computer with a 2.33GHz CPU and 16G memory. The average total executing time for classifying 20,000 monitored domain names is 751.2 milliseconds. Therefore the system can process 26624 domain names per second which is sufficient for most LAN. The classification result is shown in Table 3. It shows our approach can get a high accuracy of 98.74% and low false positive of 1.24%.

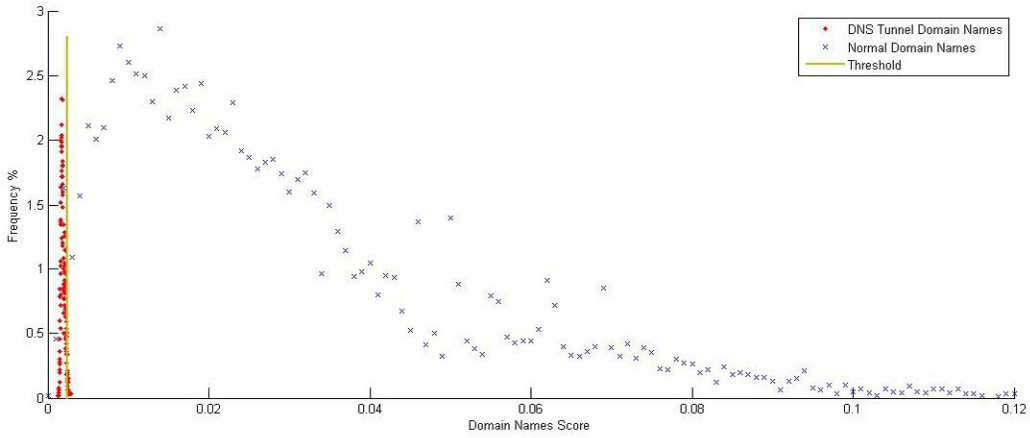

Figure 5 Domain Name Classification Result

Table 4 Executing time of classifying 20,000 domain names

Test number	Preprocessing and Scoring time(ms)	Classifying time(ms)	Total time (ms)
1	770	4.9	774.9
2	770	2.7	772.7
3	660	3.0	663.0
4	840	2.8	842.8
5	700	2.5	702.5
Average	748	3.18	751.2

5. Discussion & Conclusion

There are still some potential way for evasion of our approach in real networks The tags in DNS tunnels domain names may be used to evade detection if they are consist of high character frequency bigram and long enough. But more features could be used to classify domain names to promote the correct rate and avoid the evasion. For example, the length of domain names could be added to character frequency analysis in real time detection.

In this paper, we present an approach that can detect DNS tunnel in real time. We design a score mechanism that can distinguish DNS tunnel domain names and normal domain names based on bigram character frequency. Our experiment uses real-world data to evaluate our approach, which includes normal domain names published by Alexa and DNS tunnel domain names produced by DNScat. The results demonstrate that our approach get higher accurate in identifying DNS tunnel domain names compared with other existed methods.

Acknowledgements

This work is supported by National Natural Science Foundation of China (Grant No.61100174, 61272500)，National High Technology Research and Development Program of China, 863 Program (Grant No.2011AA010701 and 2012AA013101), and Strategic Priority Research Program of the Chinese Academy of Sciences (Grant No. XDA06030200)

References

1. Qiu, L., Zhang, Y., Wang, F., Kyung, M.Mahajan, H.R., Trusted computer system evaluation criteria, National Computer Security Center, 1985.

2. Stalmans, E., Irwin, B., A framework for DNS based detection and mitigation of malware infections on a network, Information Security South Africa (ISSA), 2011.

3. Merlo, A., Papaleo, G.,Veneziano, S. and Aiello, M., A comparative performance evaluation of DNS tunneling tools, Computational Intelligence in Security for Information Systems, 2011.

4. van Leijenhorst, T., Chin, K.W., Lowe, D., On the Viability and Performance of DNS Tunneling, The 5th International Conference on InformationTechnology and Applications (ICITA 2008), Cairns, Australia, 2008.

5.Romaña, D.A.L. and Musashi, Y., Entropy Based Analysis of DNS Query Traffic in the Campus Network, Proceedings for The 4th International Conference on Cybernetics and Information Technologies, System and Applications (CITSA 2007), Orlando, FL, USA, 2007.

6. Mehta, Y., Communication over the Internet using covert channels, 2005.

7. Casas, P., Mazel, J. and Owezarski, P. , MINETRAC: mining flows for unsupervised analysis & semi-supervised classification, Proceedings of the 23rd International Teletraffic Congress, 2011.

8. Hsu, C.H., Huang, C.Y., Chen, K.T., Fast-flux bot detection in real time, Recent Advances in Intrusion Detection, 2010.

9. Born, K. and Gustafson, D., Detecting dns tunnels using character frequency analysis, arXiv preprint arXiv:1004.4358, 2010.

10. van der Heide, H. and Barendregt, N., DNS Anomaly Detection, WWW-Dokument, staff. science, 2011.

11.Alexa.com. Alexa top 1,000,000 global sites.http://www.alexa.com/topsites, June 2012.

12. P. Mockapetris, DNS RFC 1035, http://www.ietf.org/rfc/rfc1035.txt, 1987