Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 134 (2018) 310–315

Procedia Computer Science

www.elsevier.com/locate/procedia

The 2nd International Workshop on Big Data and Networks Technologies

(BDNT 2018)

Detection of DNS DDoS Attacks with Random Forest Algorithm on Spark

Liguo Chen[a,b], Yuedong Zhang[b,*], Qi Zhao[b], Guanggang Geng[b], ZhiWei Yan[b]

[a]School of Computer and Control University of Chinese Academy of Sciences, Zhongguancun Nansijie, Haidian District, Beijing 100190, China
[b]China Computer Network Information Center, Zhongguancun Nansijie, Haidian District, Beijing 100190, China

Abstract

Domain Name System(DNS) is one of the most foundational and essential services on the Internet, the security and robustness of DNS are of great significance. However, the stable operation of DNS has been threatened by Distributed Denial of Service(DDoS) for quite a long time, especially when the number of registered names of .CN are over 20 million on November 11, 2016. According to our observation, the frequency of volume-based DDoS attacks increased rapidly in recent years, and when the attack happened, not only the authoritative servers were affected, servers of Top Level Domain(TLD) also suffered a lot. In this paper, a model based on Random Forest[1] is applied to traffic classification with an accuracy of 99.2% on Spark. The result shows that the model could be used to deal with large-scale DNS query flows, which is fast enough to be used in practice.

Peer-review under responsibility of the scientific committee of the 13th International Conference on Future Networks and Communications, FNC-2018 and the 15th International Conference on Mobile Systems and Pervasive Computing, MobiSPC 2018.

Keywords: DNS; DDoS; Traffic Filtering; Random Forest Algorithm; Spark

1. Introduction

Domain Name System(DNS) plays a significant role on the Internet, which could be used to translate the domain names into Internet Protocol(IP) addresses. It is a great step that the number of registered names of .CN exceeded 20

* Corresponding author. Tel.: +86-189-1087-3500.
E-mail address: zhangyuedong@cnnic.cn

Peer-review under responsibility of the scientific committee of the 13th International Conference on Future Networks and Communications, FNC-2018 and the 15th International Conference on Mobile Systems and Pervasive Computing, MobiSPC 2018.
10.1016/j.procs.2018.07.177

million on November 11, 2016. As the Internet is more and more important to our life, the security and robustness of DNS are of great significance.

1.1. The Progress of DNS DDoS Attack

The Water Torture Attack is a kind of DDoS attacks against the Domain Name System. During the progress of a Water Torture Attack, the attackers could firstly collect a number of zombies, not only personal computers but also devices on the Internet (the number of which has been increased rapidly in recent years).

Secondly, the scanning tools like Z-map could be used by the attackers to find out a huge amount of recursive servers on the Internet. These tools are quite powerful that all the IPv4 addresses could be scanned within only several hours if the bandwidth of scanner is big enough.

Thirdly, the attackers would send attack scripts to the zombies, with spoofed IP addresses in general. Then, these zombies would wait for the commands from a controller server or pull them local from HTTP server actively. After that, the zombies would send random queries which shared the same authoritative server in Water Torture. Massive queries would arrive at recursive servers. As these names are not covered in the local cache, the iterative queries would be sent out to authoritative servers.

Finally, these queries would arrive at each level of DNS which includes root servers, TLD servers, Second Level Domain(SLD) servers and other authoritative servers. According to the mechanism of Time to Live(TTL), all the traffic would come to the attacked authoritative server, and it's usually a big problem to deal with such a large scale of traffic by authoritative servers in a short time.

1.2. Detection and Prevention of DNS DDoS Attack

DDoS attacks are large-scale cooperative attacks launched by compromised hosts. Many researchers are working on the detection and prevention of DNS DDoS attacks in the life time. Some of them focus on botnet detection, a method could be used to prevent DDoS attacks before it happened.

The prevention of the Water Torture Attack is treated as a classification problem in this paper. The classification model is trained with the attack data of .CN from recursive servers between 2014 and 2016 by Random Forest Algorithm in Spark Mllib[2]. Spark MLlib is an efficient architecture for high-speed data processing on large clusters, and Random Forest Algorithm is an ensemble method of Bagging[3] in which each base classifier has no relationship with others while training.

2. Related Work

In this section, the mainly existing methods to defense DNS DDoS attacks will be introduced.

2.1. Botnet Detection

The clients of DDoS attack are usually hosts infected by viruses in botnets. Many researchers are working on the detection and prevention of DNS DDoS attacks.

Ishibashi et al.[4] introduced a method to discover worm e-mail senders by calculating a score for both quantity and quality of the emails sent by a single host.

Hyunsang Choi et al.[5] proposed a botnet detection mechanism based on IRC protocol by monitoring group activities in query traffic of DNS. It was shown that this mechanism could detect botnets dramatically while the zombies were connecting to their server or migrating to another server.

Junjie Zhang et al.[6] proposed a novel scalable botnet detection system which is capable of detecting P2P botnets. The statistical fingerprints of the P2P communications are used for identifying P2P clients and distinguish between normal hosts and P2P bots. The extensive evaluation shows that the detection system is with both high detection accuracy and scalability.

2.2. DDoS Attack Detection

Weizhang Ruan et al.[7] introduced a method for detecting abnormal patterns in query traffic with sequence mining techniques. The abnormal growth of traffic could be detected by this model. But the amount of DNS queries varies among different time period in a single day. If the traffic volume increased with a normal pattern during unusual hours, this algorithm would fail as it is not time-related.

Hongyuan Cui et al.[8] proposed a novel frequent episode mining algorithm which is a time-related volume trend prediction method, allowing anomalies to be detected in real time.

Spark is suitable for intrusion detections due to its perfect distributed processing capability. Jian Zhang et al.[9] proposed a Spark-Based DDoS attack detection model with an accuracy of 95%.

2.3. Query Traffic Classification

Ron Begleiter et al.[10] introduced a fast and scalable method for threat detection in large-scale DNS logs with 5% false-negative rate and 1% false-positive rate, which means 1% reasonable requests would be rejected.

Yuya Takeuchi et al.[11] proposed a method for detection and prevention of Water Torture Attack in routers of authoritative name servers. They adopted the Naive Bayes Classifier and got a precision rate of 95.59%. However, this algorithm is not useful to reduce the attack traffic of root and TLD DNS servers.

Three aspects differentiated our study from those above. Firstly, the model is applied to filter the traffic on recursive servers, which will reject faked queries from the near source. Secondly, the false-positive rate is at a sufficiently low level by using the proposed training model, which means a small number of normal queries would be failed. Last but not least, Spark is used for performance enhancing which makes the traffic filtering fast and scalable.

3. Our Work

Usually, there is noting could help except making the servers offline when a DDoS flooding attack was detected[12]. Based on the fact that DNS has a tree-like structure, although it is quite hard to release the NIC load on the victim server when DDoS attack is going, the attack traffic could be removed from parent domain if some smart traffic fileters could be applied on its child domain. The model proposed in this paper is finally applied to central recursive DNS servers in ISPs which is closer from the attack sources to reduce the attack traffic on servers of .CN.

The target of the model is not only to point out whether the DNS server is under attack but also distinguish regular queries from abnormal queries. Several statistical features are chosen from the work of Jun Wu et al.[13] and the objective of the statistical features has been switched from the query names to its sub-domain names, which is used for pointing out whether the name is a zombie. Some other features are also added to distinguish abnormal domain names from regular domain names.

In this section, each adopted features and utilizations of Random Forest Algorithm will be introduced.

3.1. Feature Set Selection

Two categories of features are adopted. In the first group, analytical features of sub-domain names during a specific time were taken. In the second category, features of the queried name itself were extracted.

The statistical features of the queried domain name are:

- Query Rate(QR). This feature denotes query counts of this domain during a certain period.
- Sub-domain Space(SS). SS means the counts of the query names in this domain during a certain period.
- Source IP Space(SIS). This feature denotes the counts of the query source IP addresses in this domain during a certain period.

The features of queried name itself are:

- Bi-gram of Name(BGN). This feature denotes the bi-gram score of the query name. This feature will be introduced in the following paragraph.
- Name Length(NL). NL means the length of the query name, which could be quite large in a Water Torture Attack.

- Name Level(NLVL). This feature denotes the level of the query name. NLVLs are always large when a Water Torture DDoS Attack is taking place.
- Is Name Server(INS). Sub-domains of name servers are usually queried at a high rate which is usually not caused by an attack. This feature denotes whether this name belongs to a name server.
- Is Reverse Query(IRQ). NLVL of a reverse query is usually very high. This feature denotes whether this sub-domain belongs to "in-addr".

N-gram is a language model in which regular sentences have higher probabilities than irregular ones in natural language processing(NLP). Suppose each character in the name is C_i and the length of the name is m, the prior probability of a domain name $C_1C_2\cdots C_m$, according to bi-gram($N=2$), is :

$$P(Name)=P(C_1/BON)*P(C_2/C_1)*\cdots*P(C_m/C_{m-1})*P(EON/C_m) \quad (1)$$

BON denotes the first character of a name, and EON means the last character of the name.

About one million normal names have been collected to train the bi-gram language model in which each $P(C_{i+1}/C_i)$ is stored to Hive[14], which is suitable for feature extraction with SparkSQL[15].

A tiny transformation has been made to realize several real number features. Q means the queries during a certain period (which is one minute by default) and Q_i denotes every single query. Then, the input vector including these features has been formulated as:

$$I(Q_i)=[\log(QR),\ \log(SS),\ \log(SIS),\log(BGN),\ NL,\ NLVL,\ INS,\ IRQ] \quad (2)$$

These features have been resized into the same scale by the log function.

3.2. Utilization of Random Forest Algorithm

The Random Forest Classifier is chosen due to the fact that it doesn't expect features that interact linearly or even linear features.Random Forest is also a bagging method which is easy to scale as each decision tree could be trained at each worker node of Spark cluster.

When Water Torture DDoS Attack happened, the analytical features may interact linearly because the names of the sub-domain under attack are generated randomly, and queried by zombies with a large amount of IP addresses. In this case, QR, SS and SIS would interact linearly. But in an AMP DDoS Attack, the zombies would claim the name record which would lead to a small SS and a large QR. Therefore, tree-based models is better than linear models.

4. Experiments

Three steps are made during the experiment: preprocessing, training, evaluation. In this section, each step will be introduced in detail.

4.1. Preprocessing

Production environment data instead of those generated by algorithms or self-defined experiments was adopted to make the model fit the attack incident in the real world. The queries logged by the site of .CN in Hong Kong on March 12, 2015 is taken for training, because .CN was under Water Torture Attack on that day.

The query logs of DNS servers are usually well-formated as:

$$query=[date,time,client\_ip,client\_port,name,ip,params] \quad (3)$$

in which *date* and *time* denote the query time, *client_ip* means the Internet address of the query client, *client_port* indicates the port of this query, *name* means the name queried, ip denotes the Internet address returned according to authoritative severs and *params* denotes other parameters such as network type etc.

After data preprocessing, the vector generated from the query is:

$$processed\_query=[date,h,m,s,ms,client\_ip,client\_port,name,tld,2ld,3ld,4ld,5ld,ip,params] \quad (4)$$

in which *date* is converted from string to integer, *time* is split into an array(*h*(*hour*), *m*(*minute*), *s*(*second*), *ms*(*millisecond*)) and *name* is split into a name array(*tld*, *2ld*, *3ld*, *4ld*, *5ld*) for features extraction. Finally, the model input extracted from the *processed_query* is as equation (2) showed.

4.2. Training and Detection Rate Evaluation

The samples are divided into two sets: training set(70%) and validation set(30%). The training set is presented to the Random Forest Classifier for training, and the validation set is for checking the classifier performance. The test set is chosen from queries of several other Water Torture attack events of .CN for the measurement of accuracy and performance.

Two criteria are chosen to evaluate the performance of the classifier: False Positive Rate(FPR) and False Negtive Rate(FNR). FPR means the rate of normal queries that are labeled as attacks which would be filtered by the firewall and FNR means the rate of attacks which could be marked as normal.FPR is more important than FNR because only few failures of common queries could be tolerated in a production environment.

The number of decision trees is an important hyper parameter of Random Forest Classifier. Figure 1 (a) shows that FPR decreases with the growth of the amount of decision trees and figure 1 (b) indicates that FNR increases smoothly and keeps steady at 0.04.

When one decision tree exists, Random Forest Classifier becomes a Decision Tree which would have a problem of over-fitting. With the increasement of the count of trees, this issue would be overcome. Figure 1 (c) shows the confusion matrix when there are 20 decision trees. FPR decreases to 0.0, and FNR keeps steady at around 0.04 which means that about 96% attack traffic would be filtered without common queries denied.

4.3. Performance Evaluation

The query samples adopted for performance evaluation is the log information of recursive server(1.2.4.8) on November 11, 2016.

The experiment is performed on a desktop PC, and the relative information is shown in Table 1.

There are mainly two steps to label each query. Features extraction converts the raw query into $I(Q_i)$, and the classifier gives the label according to Q_i.

Features extraction costs more time than classification. As is shown in table 2, the time cost of features extraction increases linearly with the queries count while the time cost of classification keeps steady.

The main innovation of Spark was to introduce an in-memory caching abstraction. This makes Spark ideal for workloads where multiple operations access the same input data. What's more, Spark splits the jobs that users submit to several stages, and each stage contains many tasks which can be executed in parallel at each worker node.

Servers in Recursive Resolution Cloud cover DNS queries from 20 provinces in China, and about 0.71 million queries per minute is received. It would take only 20.34 seconds to label these queries in such scale by the firewall on a single desktop PC, that means this model meets the performance requirement in practice.

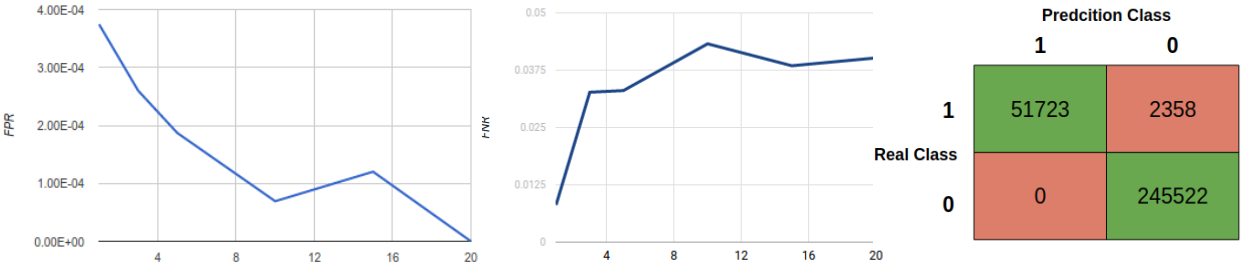

Fig. 1. (a) FPR and trees count; (b) FNR and trees count; (c) the confusion matrix.

Table 1. Hardware and software of the detection machine

hardware and software	information
OS	64-bit Ubuntu 16.04.1 LTS
CPU	intel i5-3570 CPU @ 3.40GHz * 4
Memory	12GB
Spark Version	2.0.0

Table 2. time cost with the increment of queries count

queries count	feature extraction (s)	classification (s)
2000	1.88924	0.292806
20000	2.94668	0.320008
200000	8.45116	1.024076
2000000	57.84464	1.701591

5. Conclusion

In this paper, a novel method to reduce the DDoS traffic on TLD servers is introduced, in which traffic filter based on machine learning algorithms is applied to major recursive DNS servers on the Internet. The classification model is built on spark and performs with 0.0% FPR and 4.36% FNR, which means both the accuracy and performance demands in practice is met. In future work, the features will be extracted and the model will be applied in a streaming-based way which is more suitable for real-time rules making by the firewall. A real-time detection and prevention system with this traffic filtering model will also be studied in the near future.

References

[1] A. Liaw, M. Wiener, Classification and regression by randomforest, R news 2 (3) (2002) 18–22.

[2] Y. Takeuchi, T. Yoshida, R. Kobayashi, M. Kato, H. Kishimoto, Detection of the dns water torture attack by analyzing features of the subdomain name, Journal of Information Processing 24 (5) (2016) 793–801.

[3] X. Meng, J. Bradley, B. Yuvaz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen, et al., Mllib:Machine learning in apache spark, JMLR 17 (34) (2016) 1–7.

[4] L. Breiman, Bagging predictors, Machine learning 24 (2) (1996) 123–140.

[5] K. Ishibashi, T. Toyono, K. Toyama, M. Ishino, H. Ohshima, I. Mizukoshi, Detecting mass-mailing worm infected hosts by mining dns traffic data, in: Proceedings of the 2005 ACM SIGCOMM workshop on Mining network data, ACM, 2005, pp.159–164.

[6] H. Choi, H. Lee, H. Lee, H. Kim, Botnet detection by monitoring group activities in dns traffic, in: Computer and Information Technology, 2007. CIT 2007. 7th IEEE International Conference on, IEEE, 2007, pp. 715–720.

[7] J. Zhang, R. Perdisci, W. Lee, X. Luo, U. Sarfraz, Building a scalable system for stealthy p2p-botnet detection, IEEE transactions on information forensics and security 9 (1) (2014) 27–38.

[8] W. Ruan, Y. Liu, R. Zhao, Pattern discovery in dns query traffic, Procedia Computer Science 17 (2013) 80 – 87. doi:http://dx.doi.org/10.1016/j.procs.2013.05.012.

[9] H. Cui, J. Yang, Y. Liu, Z. Zheng, K. Wu, Data mining-based dns log analysis, Annals of Data Science 1 (3-4) (2014) 311–323.

[10] J. Zhang, Y. Zhang, P. Liu, J. He, A spark-based ddos attack detection model in cloud services, in: International Conference onInformation Security Practice and Experience, Springer, 2016, pp. 48–64.

[11] R. Begleiter, Y. Elovici, Y. Hollander, O. Mendelson, L. Rokach, R. Saltzman, A fast and scalable method for threat detection in large-scale dns logs, 2013 IEEE International Conference on, IEEE, 2013, pp. 738–741.

[12] S. T. Zargar, J. Joshi, D. Tipper, A survey of defense mechanisms against distributed denial of service (ddos) flooding attacks, IEEE communications surveys & tutorials 15 (4) (2013) 2046–2069.

[13] J. Wu, X. Wang, X. Lee, B. Yan, Detecting ddos attack towards dns server using a neural network classifier, in: International Conference on Artificial Neural Networks, Springer, 2010, pp. 118–123.

[14] A. Thusoo, J. S. Sarma, N. Jain, Z. Shao, P. Chakka, S. Anthony, H. Liu, P. Wyckoff, R. Murthy, Hive: a warehousing solution over a map-reduce framework, Proceedings of the VLDB Endowment 2 (2) (2009) 1626–1629.

[15] M. Armbrust, R. S. Xin, C. Lian, Y. Huai, D. Liu, J. K. Bradley, X. Meng, T. Kaftan, M. J. Franklin, A. Ghodsi, et al., Spark sql:Relational data processing in spark, in: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, ACM, 2015, pp. 1383–1394.