DGA Botnet detection using Collaborative Filtering and Density-based Clustering

Trung-Duc NGUYEN
Hanoi University of Sience and Technology
1, Dai Co Viet Street
Hanoi, Vietnam
trungduc110492@gmail.com

Tuan-Dung CAO
Hanoi University of Sience and Technology
1, Dai Co Viet Street
Hanoi, Vietnam
dungct@soict.hust.edu.vn

Linh-Giang NGUYEN
Hanoi University of Science and Technology
1, Dai Co Viet street
Hanoi, Vietnam
giangnl@soict.hust.edu.vn

ABSTRACT

In recent years, the botnet phenomenon is one of the most dangerous threat to Internet security, which supports a wide range of criminal activities, including distributed denial of service (DDoS) attacks, click fraud, phishing, malware distribution, spam emails, etc. An increasing number of botnets use Domain Generation Algorithms (DGAs) to avoid detection and exclusion by the traditional methods. By dynamically and frequently generating a large number of random domain names for candidate command and control (C&C) server, botnet can be still survive even when a C&C server domain is identified and taken down. This paper presents a novel method to detect DGA botnets using Collaborative Filtering and Density-Based Clustering. We propose a combination of clustering and classification algorithm that relies on the similarity in characteristic distribution of domain names to remove noise and group similar domains. Collaborative Filtering (CF) technique is applied to find out bots in each botnet, help finding out offline malwares infected-machine. We implemented our prototype system, carried out the analysis of a huge amount of DNS traffic log of Viettel Group and obtain positive results.

CCS Concepts

• Theory of computation~Theory and algorithms for application domains • Computing methodologies~Machine learning • Networks~Network services • Software and its engineering~Software organization and properties

Keywords

Big Data; Botnet; Domain Generation Algorithm; Machine Learning.

1. INTRODUCTION

Botnets are networks of malware-compromised machines, or *bots*, that can be remotely controlled by an attacker through a command and control (C&C) communication channel [8]. Recently, they have been known as the root cause of many kind of Internet attacks. The attacker (*botmaster*) employs them to send spam email, host phishing web-pages, perform DDOS attack, steal user's information or mine Bitcoins, etc...

The first botnet in the world, Global Threat Bot (GTBot) [17], was detected in 2000. GTBot was written by Sony, mSg and DeadKode using the legitimate mIRC program as its main core and a program called HideWindow or a vbs script to make mIRC invisible to the host computer. mIRC is scripted to generate an IRC Bot which replies to remote commands from the BotMaster. When run mIRC loads the scripts and hiding program then connects to the predetermined server and channel to wait for commands. GTBot was used for Denial of Service (DoS) attack and port scanning. After the apperance of GTBot, hackers in the world have created many other botnets, such as Bagle which was detected in 2004 with 230.000 infected-computers, Akbot (1.300.000 bots), Zeus (3.600.000 bots) ...

To create a botnet and implement a malicious activity such as email spamming, hacker will go through following step:

- A botnet operator infect ordinary users' computers by diseminating worms or viruses. A malicious application is set up and turn the victim's computer to bot.
- The bot on the infected PC connect to a particular C&C server.
- A spammer purchases the services of the botnet from the operator.
- The spammer provides the spam messages to the operator, who controls the compromised machines, causing them to send out spam messages.

The differences between botnet and other malwares are in the way they control communication network. Two common botnet architectures that have been known until now are centralized and peer-to-peer (P2P). In the former one, bots connect directly to some special host called C&C server to receive commands from them. Most of botnets now use this architecture because it is easy to deploy and maintain. Bots and C&C server can be connected by protocols such as HTTP or HTTPS. Aside from that, some botnets use IRC protocol. Single-point-of-failure of centralize architecture is that if C&C server is detected and taken down, whole botnet would be removed. The peer-to-peer architecture can overcome this weakness. Due to dynamic architecture, it is hard to find out C&C server. However, it is difficult to deploy, operate and maintain a peer-to-peer botnet.

SoICT 2015, December 03-04, 2015, Hue City, Viet Nam © 2015 ACM.
ISBN 978-1-4503-3843-1/15/12…$15.00
DOI: http://dx.doi.org/10.1145/2833258.2833310.

Over time, attackers developed a new botnet generation which not only has the simplicity of centralized architecture but also make C&C server be harder to take down as it is in the P2P botnets. It means, instead of connecting to somes C&C server address, bots periodically execute a Domain Generation Algorithm (DGA) to create a list of candidates C&C server domain. DGAs usually use current system time as random seed to ensure that output set will be different each time bot execute algorithm. With a collections of domains, bots will send DNS queries to resolve IP address of domains. Usually, when hackers want to send command to bots, they will register one or some domains in group of domains which will be created by DGA. Therefore, most of created domains will not be resolved to ip address, these domains is called non-existent domain (NXDomain). This strategy improves the robustness of botnet because even though one or more C&C servers are located and taken down, the bots will finally get the relocated C&C server via DNS queries to the next set of automatically generated domains.

The number of domains generated by DGA is large, for example: Conficker-A create 250 domains per 3 hour; at the same time, in Conficker-C edition, each bot create 50000 domains. Because of this fact, from system administrator's point of view, appling a filter rules or using black list domain for C&C server is impossible. Existing approach to detect this kind of botnet is based on reverse engineering in hopes of knowing how the domain generation algorithm works. However, this task is difficult, tedious and ineffective in many cases.

In this paper, we present a large-scale botnet detection system – GOTS which is dedicaded to DGA based botnets. This system has the following functionalities and characteristics:

- Detecting centralized architecture botnets by analyzing DNS traffic logs of a monitored computer network.
- Detecting bots in each botnet.
- Having ability to detect new editions of botnet which hard to find by reverse malware binaries.
- Based on Big Data platform, processes a huge amount of data.

For archiving this goal, we propose a novel method to detect DGA botnets without reversing. The event occurs when multiple computers send DNS queries with non-existent domain in a same time is a important signal for detect DGA botnet. By recording log of DNS traffic from user's computers to DNS server, we gain datasets of NXDomains created by DGA Botnet and connection requests from users to NXDomains. Based on the similarity in characteristic distribution of domain names, we combine clustering and classification algorithms to remove noise and group similar domains to corresponding cluster. In the next step, we apply Collaborative Filtering (CF) technique to find out offline malwares infected-machine in each botnet, which do not create access log to NXDomains.

The rest of the paper is organized as follows. In Section 2 we give the literature review. In Section 3, we present an overview of GOTS system and our detection method. The domain filtering task is described in Section 4. Sections 5 and 6 describe the DGA clustering and Botnet detection processes. The experimental results are introduced in Section 7 before concluding in Section 8.

2. RELATED WORK

Botnet detection is an active research topic in recent years. Botminer [4] detects botnet by analyzing log flow traffic between client and server computer. This method tries to find anomalous flow that is created by botnet. Flow features such as number of flows per hour, number of packets per flow, average bytes per packet, average bytes per second were exploited. Yadav and Reddy [11,12] used DNS traffic log to detect botnet. They analyze statistical features of NXDomain to detect which domain is created by human and which is generated by DGA botnet. Manos and Roberto [8] method also works on DNS traffic log. Besides using statistical features, they created a connection matrix between user and NXDomain, then used Spectral Clustering [21] to cluster the NXDomain. After that, Hidden Markov Model [20] was applied to detect C&C server domain. However, this method does not show good performance when it is applied on our datasets because of noising and failing in active domain classification. Furthermore, Spectral clustering requires a given number of output clusters to perform, however with real dataset we do not know how many botnets are there.

Zhou, Li, Miao, and Yim [15] present a detection method that is based on calculating the similarity of live time span and visit pattern. Their insight is that in a certain period of time, a DGA-domain is queried by a group of clients, but in next period of time, this domain has no query at all. The query characteristic of DGA-domain is different from legitimate domain, requests to legal domain appear randomly but for DGA-domain it is queried only in a short time. The domain set active time distribution feature was calculated to filter out suspected DGA-domains.

Leyla Bilge, Engin Kirda, Christopher Kruegel, and Marco Balduzzi [7] introduced EXPOSURE, a system that employs large-scale, passive DNS analysis technique to detect domains that are involved in malicious activity. They use 15 features extract from the DNS traffic that allow them to characterize different properties of DNS names and the ways that domains are queried.

Stefano Schiavoni, Federico Maggi, Lorenzo Cavallaro, Stefano Zanero [7] proposed Phoenix, which leverages publicly available passive DNS traffic to find AGDs, characterize the generation algorithms, isolate logical groups of domains that represent the respective botnets, and produce novel knowledge about the evolving behavior of each tracked botnet.

GOTS differs from the approaches described above in the following ways. (**A**) By clustering NXDomains with DBSCAN algorithm, our method does not require to vectorize domain and specify number of output cluster. (**B**) We use Collaborative Filtering algorithm to finding out offline malware infected-machine, which accounted for a large number in real world botnet.

Unlike previous work, the purpose of our system is not a real-time botnet detection, so we do not build a model for malware domain. GOTS collects DNS traffic log of every two days and perform a batch-processing task to detect botnet in monitored network. The proposed method require to maintain a blacklist domains of well-known botnet, but it not only identify published botnet, but GOTS also can detect new DGA botnet which generate randomly domain name for C&C server. We suggest using DBSCAN [19] algorithm to cluster NXDomain, its advantage is do not need to specify the number of output clusters. Lastly, it is noted that our solution works on big data platform, so it has scalability. GOTS can be performed on large scale monitored network which produce huge amount of internet traffic log everydays.

3. SYSTEM OVERVIEW

In this section, we provide an overview of our DGA botnet detecting system - GOTS.

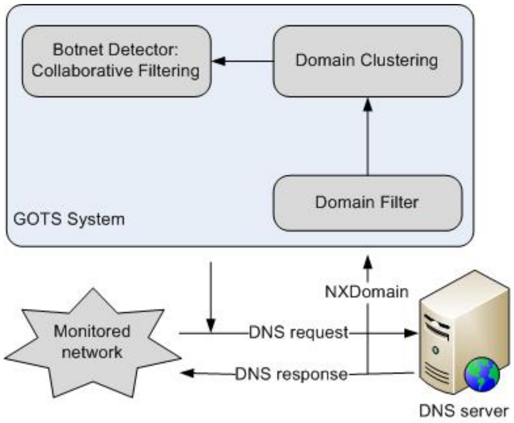


Figure 1. GOTS system overview.

In Figure 1, all DNS traffic from a monitored network to DNS server are collected. All request and response packets shall be logged and stored in our data warehouse (we use Hadoop cluster to store this large dataset). After that, GOTS system analyze this log through three steps: *Domain Filtering* to remove benign domain in collected dataset, *Domain clustering* to cluster NXDomain with each cluster of domains is generated by a DGA, *Botnet detection* to find bots in botnets.

In Domain Filtering step, we try to remove all domains which are known as not being generated by DGA botnet such as www.google.com, www.facebook.com, www.amazon.com, ... Because of some objective reason such as faulty network connection or connection time out, ... that domains is not resolvable and it will be logged in our dataset as NXDomain. Because we want to collect only domains generated by DGA botnet, we will apply clustering-classification technique to remove benign domain in our dataset to reduce noise for later analysis.

In Domain Clustering step, we group all NXDomains in our dataset using DBSCAN algorithm [19]. The purpose is to group similar NXDomains to the same cluster, each cluster should correspond to a particular DGA botnet type. Domain Clustering module takes as input the following information: (1) list of NXDomain from Domain Filtering step and (2) user connected to each NXDomain in that list. The distance between two domains used in clustering algorithm is a function of the number of users connected to both of them.

The final step in our GOTS system is Botnet Detection. In this phase, the NXDomain clusters which are generated from Clustering step, will be used as input data for botnet detection. All users connected to clustered domain are classified as bot in corresponding botnet. With remaining users, we used Collaborative Filtering method to estimate a probability that user may be connected to a clustered domain. In other word it is the probability that user is a bot in botnet. A user will be classified as a bot if the value of probability is greater than a predefined threshold. This idea comes from the fact that user may offline at a time bots need connect to C&C server and it do not create DNS queries.

4. DOMAIN FILTERING

While recording DNS traffic log, we notice that collection of domains received contains a lot of benign domains such as www.google.com, www.facebook.com, amazon.com, 24h.com.vn, etc ... When network connection is failed or timed-out, user can not connect to these domains. Then, these domains are recorded into group of NXDomains, and will produce deviation to the later analyzing of NXDomains created by DGA. Therefore, in domain filtering step, all of the benign domains have to be removed and the remaining domains are all generated by DGA botnet and will be used for further analysis.

4.1 Using white-list filtering

A simple method to remove benign domain is using a white-list of top-query domain. In GOTS system, we use a list of top 1.000.000 domains extract from alexa.com. Simultaneously, we apply wild card filtering to remove all domain have pattern like *.google.com, *.facebook.com, *.amazon.com, etc. At this phase, approximately 80% of benign domains in collected dataset will be removed.

4.2 Using clustering-classification

Besides using white-list filtering, we also analyze statistical features of malicious domain generated by DGA to remove benign domain. To ensure that domains generated by DGA have not been registered, attacker often tries to generate long and insignificant domain. This is an important feature to detect which domain is generated by DGA botnet.

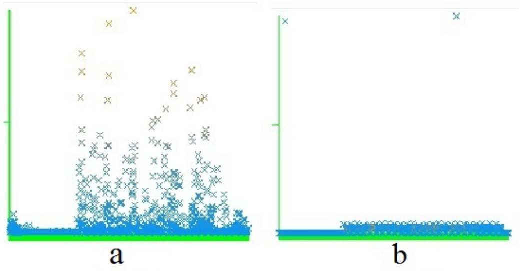


Figure 2. Frequency of 2-gram: a. Domains from alexa.com b. Domains of Conficker botnet.

We will apply clustering-classification technique to decide whether a domain is generated by DGA or not and analyze important statistical features such as distribution of n-gram and length of domain name using training dataset of 1.000.000 top-query domains from alexa.com.

While domain is created by normal people often use some characters, domain generated by DGA tend to use all characters. In our GOTS system, we compute frequency of n-gram with n = 1, 2, 3. With training dataset of 1.000.000 benign domain from alexa.com and 5.000 domain of published botnet, we have distribution of 2-gram frequency value in Figure 2. Cryptolocker botnet generated domain with equal frequency of 2-gram, opposed to domain in list 1.000.000 domains from alexa.com.

Assume that domain d = 'facebook.com', probability of the domain can be calculated by:

P(facebook.com) = P(fa) + P(ac) + ... + P(om) = 0.0154

For other domain d = 'awetfbxzr.ru':

P(awetfbxzr.ru) = P(aw) + P(we) + ... + P(ru) = 0.000031

After analyzing length and n-gram features, we form two-dimensional vector for each domain. With input is 1.000.000 domains from collected dataset, we applying k-means clustering algorithm with k = 2, then we have two centroids corresponding to two clusters: botnet and non-botnet domains. We use these two centroids to classify new input domain. With each new input domain, we translate it to a two-dimension vector and then compute Euclide distance to two centroids, the closer centroid will be the cluster that domain belong to. Finally, we remove all domains that belong to benign cluster

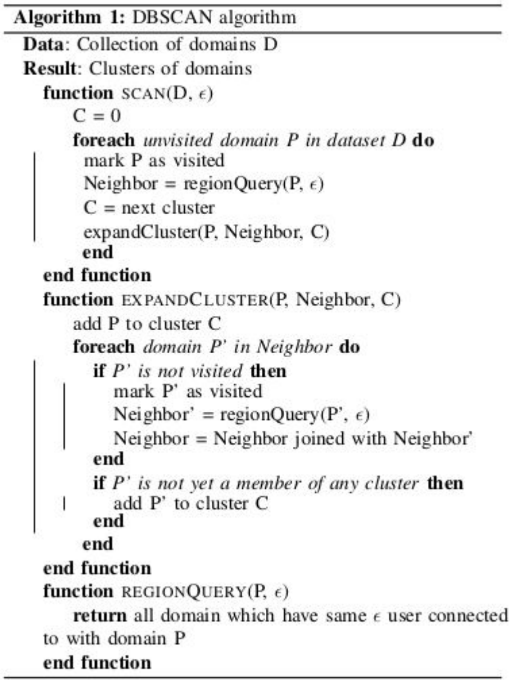
5. DOMAIN CLUSTERING

The Domain Clustering module receives NXDomain from the Filtering module. This step clustering NXDomains to clusters, each cluster contains NXDomains which are generated by a DGA. The idea of clustering step comes from the fact that the probability of event that two distinct user connect to same NXDomain is very low. In other words, two NXDomain will be clustered to same cluster if they have same set of connected user. Manos and Roberto [4] clustered NXDomain by constructing a matrix of connection between user and NXDomain. After translating each NXDomain to a vector by doing eigen decomposition, they cluster NXDomains by X-means algorithm. Based on K-means, it runs K-means multiple times and uses Bayesian Information Criterion (BIC) to obtain optimal cluster k. With large scale dataset, X-means algorithm is not effective because if k is large, the number of iterative K-means is high so we need more time to do clustering.

To solve this problem, we use idea of DBSCAN [19] - the density-based clustering algorithm. This algorithm has advantage that it does not require to specify the number of output clusters as opposed to K-means. Another advantage is DBSCAN can find arbitrary shaped clusters. To determine distance between two elements in dataset, namely the distance between two domains and, we suggest that it can compute by the number of user connect to both domains *k*, if *k* is greater than *eps* with *eps* is a threshold, we decide that two domain and is neighborhood.

Algorithm is described in details in algorithm 1.

With our collected dataset contain domains and clients connected to, DBSCAN algorithm iterates each domain and get a list of neighbor domain (have same eps user connected to) then added it to current cluster. The process repeats until the cluster is not extends, then turn to the new cluster. Output of clustering step is clusters of NXDomains and each of them includes NXDomains that are suspected be generated by same DGA algorithm. Because our collected dataset has many noises that are benign domains (i.e *.googlevideo.com, *.fbcdn.com...), so many of output clusters contain only benign domain and should be removed. To obtain expect result, after running DBSCAN clustering, output clusters will be checked by security expert to remove all benign cluster, finally we have only NXDomain clusters that are generated by DGA botnet.



```
Algorithm 1: DBSCAN algorithm
 Data: Collection of domains D
 Result: Clusters of domains
   function SCAN(D, $\epsilon$)
     C = 0
     foreach unvisited domain P in dataset D do
       mark P as visited
       Neighbor = regionQuery(P, $\epsilon$)
       C = next cluster
       expandCluster(P, Neighbor, C)
     end
   end function
   function EXPANDCLUSTER(P, Neighbor, C)
     add P to cluster C
     foreach domain P' in Neighbor do
       if P' is not visited then
         mark P' as visited
         Neighbor' = regionQuery(P', $\epsilon$)
         Neighbor = Neighbor joined with Neighbor'
       end
       if P' is not yet a member of any cluster then
         add P' to cluster C
       end
     end
   end function
   function REGIONQUERY(P, $\epsilon$)
     return all domain which have same $\epsilon$ user connected
   to with domain P
   end function
```

6. BOTNET DETECTION

We use Collaborative Filtering [18] (CF) to find bots in botnets. CF is a technique used by some recommender systems. Our idea is using CF to find all users whose behavior is similar with a user that is considered to be a bot in botnet. If two computers are compromised with same DGA malware, they will generate same collection of NXDomains. Assume that user A has same set of NXDomain with user B and user A is a bot in botnet, but user B did not connect to any domain in that botnet. Because the behavior of user A and B are similar, we can calculate the probability that user B connect to the domain in common botnet through CF or otherwise the probability that user B is component of botnets.

Collaborative Filtering requires three conditions to perform. Firstly, to predict the future behavior of a users B, we need to build a model of his past behavior. In this case, the domains that he has accessed to are basis for determining other users who have similar behavior in the system. Secondly, it is necessary to find a simple way to represent users' interests to the system. Thirdly, we need an algorithm that is able to match users with similar behaviors.

In common systems which employ collaborative filtering such as recommender systems, there is only one score when user buy item and review or rate it. In our GOTS system, we use two kinds of score, the first is domain score represent the probability that domain is generated by DGA botnet, the second score is user score represent the probability that user is a bot in botnet.

Our GOTS system using CF to detect user who is bot in botnet through following steps:

- Collect the set of domains that is accessed by users in monitored network and give each domain a score.

- Analyze access log to find users who have same behavior and same set of connected domains.
- List all users who have similar behavior with user who is considered as a bot in botnet and compute a probability that each user of group is a bot.

To calculate the similarity between two users A and B, there are many measures, such as Pearson correlation and vector cosine based.

The Pearson correlation similarity of two users x, y is defined as:

$$simil(x,y)=\frac{\sum_{i\in I_{xy}}(r_{x,i}-\overline{r_x})(r_{y,i}-\overline{r_y})}{\sqrt{\sum_{i\in I_{xy}}(r_{x,i}-\overline{r_x})^2\sum_{i\in I_{xy}}(r_{y,i}-\overline{r_y})^2}}$$

The cosine-based approach defines the cosine-similarity between two users x and y as:

$$simil(x,y)=\cos(\vec{x},\vec{y})=\frac{\vec{x}.\vec{y}}{\|\vec{x}\|\times\|\vec{y}\|}=\frac{\sum_{i\in I_{xy}}r_{x,i}r_{y,i}}{\sqrt{\sum_{i\in I_x}r_{x,i}^2}\sqrt{\sum_{i\in I_y}r_{y,i}^2}}$$

The detection module takes as input the following information: *(i)* a list of domain that is known as domain of botnet, *(ii)* a list of domain clusters receive from clustering module, *(iii)* a list of domain generated by host in monitored network. With each domain d in input sets, we assign a score s_d represent the probability that domain *d* is generated by DGA botnet. With domain in list (i), we assign s_d = 1.0 that means domain is generated by botnet certainly. With domain in list (ii) we assign s_d value in range from 0.6 to 0.8 depend on result of clustering step. With each domain *d* in list (iii), if *d* is found in (i) or (ii) then value of s_d is unchanged, otherwise s_d = 0.0.

All domain have s_d = 0 will be update score by formula:

$$s_d=1-\prod_{i=1}^{n}(1-s_{u_i})$$

with $\{u_i\}_{i=1,n}$ is list of user connected to domain *d*, each user *u* have a score s_u can compute by

$$s_u=\alpha\times\max(s_{d_i})$$

with $\{d_1,d_2,...,d_n\}$ is list of domain that user u connected to.

After calculate s_d for all domain *d* in input dataset, we can calculate the similarity between two user by formula (2). Then, the probability that user *u* connect to domain *d* in future can compute by:

$$r_{u,\mathrm{d}}=k\sum_{u'\in U}simil(u,u')r_{u',\mathrm{d}}$$

with U is top N users connect to domain *d* with highest *simil(u, u')*, *k* is normalize factor is defined as:

$$k=\frac{1}{\sum_{u'\in U}|simil(u,u')|}$$

Finally, after calculate all $r_{u,i}$ value, with domain *d* was considered a domain generated by DGA botnet, we can detect all user *u* that can be a bot in this botnet if user *u* have $r_{u,\mathrm{d}}>\theta_r$ with θ_r is a threshold will be optimize for best result during system operation.

Because our dataset is very large so we use Map-Reduce programming model to compute *simil(x, y)*. That value is calculated through 4 Map-Reduce jobs as detailed in figure from 3 to 6.

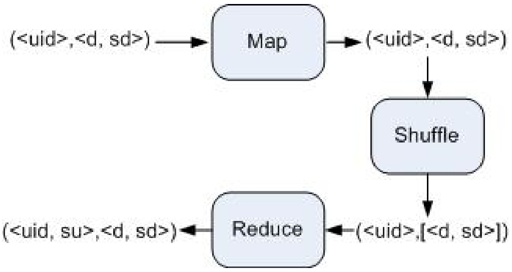


Figure 3. First map-reduce job to compute similarity.

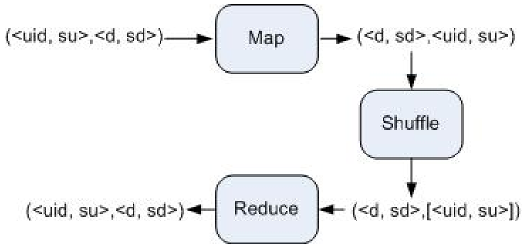


Figure 4. Second map-reduce job to compute similarity.

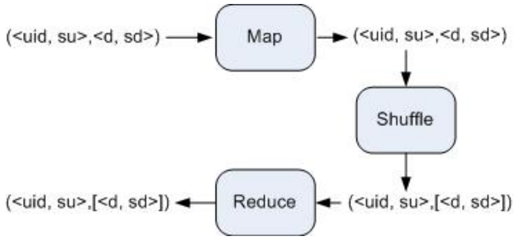


Figure 5. Third map-reduce job to compute similarity.

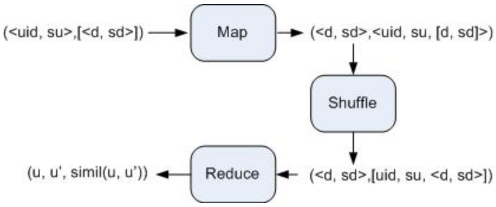


Figure 6. Final map-reduce job to compute similarity.

7. EXPERIMENTS

In this part, we describe dataset collected by recording DNS traffic log of about 18.000 users in a area in 2 weeks. Then, we show the results of detecting some botnets in the dataset. The DNS traffic is collected from 1/4/2015 to 15/4/2015. Each record in our dataset is in following format: *<timestamp>, <uid>, <domain>* in which *timestamp* is timestamp when a user having *uid* connect to *domain*. Each user in monitored network will be assigned a *user id* value to identify with others, we do not use user IP address to identify user because it may be change over time.

The number of domains collected in the dataset was 1.190.000 and the number of users in our monitored network is approximated 18.000. There are some users connected to published C&C domains. List of domains is showed in table 1.

Table 1. Known C&C domain in dataset

Domain	**Botnet name**
imgirmyddbsniuh.pw	necurs
kfncxvayakmb.com	necurs
ultrttvbvjaanrj.jp	necurs
oiijxplrnmvgskxwaye.ru	necurs
tx.mostafaaljaafari.net	palevo
mst.com.ua	palevo
kuspehu.ru	vawtrak
ketoshi.com	vawtrak

Table 2. Domain Filtering result

	Benign domain	Botnet domain
Decide as benign	523	114
Decide as botnet	148	215

At the filtering phase, for removal of benign domains, we will partly classify domains into two classes with centroids corresponding with two kind of domains: malicious and benign domain. This two centroids are estimated using K-means clustering algorithm with k = 2 with training dataset is list of 1.000.000 domains from alexa.com and 4.000 NXDomains from the real dataset we collected. After having 2 centroids corresponding with two kind of domain: botnet and not-botnet, we evaluate precision of this model with testing dataset of 1.000 NXDomains extracted from the real data collected. Experiment data was verified by experts and we gain that there are 329 domains generated by botnet and 671 benign domains in this 1.000 NXDomains. After filtering this 1.000 NXDomains with the filtering tool, we have a result presenting in table 2. The precision and recall of our model are:

$$Precision = \frac{523}{523+114} = 82\%$$

$$Recall = \frac{523}{523+148} = 78\%$$

In clustering step using DBSCAN algorithm, the output are 20 clusters including 2 clusters of botnet that is checked by experts. The list of some domains of this 2 cluster is presented in table 3.

To evaluate performance of this step, we also try both K-means and DBSCAN algorithm. While DBSCAN takes 131 minutes to give results, K-means takes 15 minutes with k = 20. But in fact, we do not know parameter k to input to K-means.

Table 3. DBSCAN result

DGA-1	**DGA-2**
zolffidccad.info	3330.com
ojuqyzt.biz	7820.com
dwjdjsokddl.org	8030.com
tsgiuxm.biz	2882.com
ramreegv.info	6024.com
usvltnueo.biz	1067.com
oymbfohtioq.cc	5239.com
fwjiu.cn	6899.com
hughoyvpml.info	4158.com
qvfhzxq.net	8902.com

Table 4. Collaborative Filtering result

uid	score	domain
user_id_123	0.026285	1280.com
user_id_735	1	devicesta.ru
user_id_142	0.027088	restlesz.su
user_id_896	1	sso.anbtr.com
user_id_573	0.123983	sso.anbtr.com

The result of botnet detection module is presented in table 4. Score value is probability that user have *uid* value shall connect to *domain*, in other word, it is probability *uid* is bot of botnet. In our system, if score >= 0.05 then we decide *uid* is bot. Figure 7 illustrates several bots that we detected with and without applying Collaborative Filtering algorithm. The results show that with CF the proposed method find out approximate 50% offline malware infected-computer that does not created access log to NXDomains.

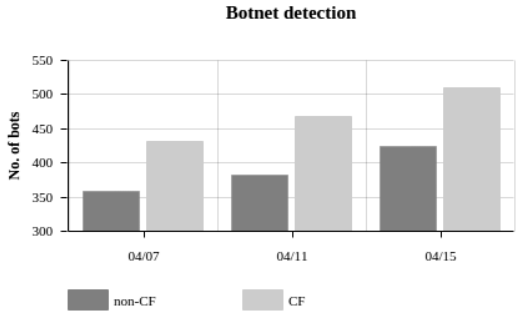


Figure 7. Botnet detection result with CF and non-CF.

8. CONCLUSIONS

In this paper, we introduce a DGA-botnet detection system – GOTS. After deploy and run with real-world dataset, we find that the system is capable of detecting several botnets, not only known botnet based on blacklist but also previously unseen botnet.

We propose a distance metric that was applied in DBSCAN algorithm that successfully clusters domains. Our system used Collaborative Filtering algorithm to finding out offline infected-machine in each botnet, this is new approach in our system.

GOTS has the advantages that can be performed automated and discovered botnet faster than existing methods. However, our system can only detect botnet which have centralized architecture, can not detect P2P botnet. Another disadvantage is that it can not figure out C&C server address of each botnet because resolvable domain may not be the C&C address, therefore we can only find bots in botnet, C&C server address still must be checked by security expert. In this paper, we only collect DNS traffic log and analyze the characteristics of the generated domain, so we miss others features of log flow traffic. In the future, we will improve the system by analyzing the characteristics of each communication session by using features of log flow traffic.

9. ACKNOWLEDGMENTS

This research is supported by the Vietnam Ministry of Education and Training research project "Development of DDoS attack prevention and botnet detection system".

10. REFERENCES

[1] A. Ramachandran and N.Feamster. Understanding the Network-level Bahavior of Spammers. In ACM SIGCOM, 2006.

[2] Antonakakis, M., Perdisci, R., Lee, W., Vasiloglou, N., Dagon, D. Detecting malware domains at the upper DNS hierarchy. In USENIX Security, vol. 11, 2011.

[3] Bilge, L., Balzarotti, D., Robertson, W., Kirda, E., Kruegel, C. Disclosure: detecting botnet command and control servers through large-scale netflow analysis. In ACSAC, ACM. 2012.

[4] Guofei Gu, Roberto Perdisci, Junjie Zhang and Wenke Lee. BotMiner: Clustering Analysis of Network Traffic for Protocol and Structure Independent Botnet Detection. SS'08 Proceedings of the 17th conference on Security symposium, pages 139-154. USENIX Association Berkeley, CA, USA 2008. ACM.

[5] Holz T., Gorecki C., Rieck K., Freiling F.C. Measuring and detecting fast-flux service networks. In NDSS. 2008.

[6] J.Leyden. Conficker zombie botnet drops to 3.5 million. http://www.theregister.co.uk/2009/04/03/conficker_zombie_count/. 2009.

[7] Leyla Bilge, Engin Kirda, Christopher Kruegel, and Marco Balduzzi. EXPOSURE: Finding Malicious Domains Using Passive DNS Analysis. In Proceedings of NDSS, 2011.

[8] Manos Antonakakis, Roberto Perdisci, Yacin Nadji, Nikolaos Vasiloglou and Saeed Abu-Nimeh, Wenke Lee and David Dagon. From Throw-Away Traffic to Bots: Detecting the Rise of DGA-Based Malware. USENIX Security Conference '12, 2012.

[9] P.Kleissner. Analysis of Sinowal. http://web17.webbpro.de/index.php?page=analysis-of-sinowal. 2008.

[10] S. Yadav, A. K. K. Reddy, A. N. Reddy, and S. Ranjan. Detecting algorithmically generated malicious domain names. In Proceedings of the 10th annual Conference on Internet Measurement, IMC '10, pages 48–61. New York, NY, USA, 2010. ACM..

[11] S. Yadav, A. K. K. Reddy, A. N. Reddy, and S. Ranjan. Detecting algorithmically generated malicious domain names. In Proceedings of the 10th annual Conference on Internet Measurement, IMC '10, pages 48–61. New York, NY, USA, 2010. ACM..

[12] S. Yadav and A. N. Reddy. Winning with dns failures: Strategies for faster botnet detection. In 7th International ICST Conference on Security and Privacy in Communication Networks, 2011.

[13] Stefano Schiavoni, Federico Maggi, Lorenzo Cavallaro, Stefano Zanero. Tracking and Characterizing Botnets Using Automatically Generated Domains

[14] 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: CCS. ACM (2009).

[15] Yong-lin Zhou, Qing-shan Li, Qidi Miao and Kangbin Yim. DGA-Based Botnet Detection Using DNS traffic. Journal of Internet Services and Information Security (JISIS), volume 3, number 3/4, pp. 116-123.

[16] The Spamhaus Project. ZEN. http://www.spamhaus.org/zen/.

[17] GTBot History. http://golcor.tripod.com/gtbot/gtbot_history.htm

[18] Collaborative Filtering. http://en.wikipedia.org/wiki/Collaborative_filtering.

[19] DBSCAN algorithm.

http://en.wikipedia.org/wiki/DBSCAN

[20] Hidden Markov Model. http://en.wikipedia.org/wiki/Hidden_Markov_model

[21] Spectral Clustering Algorithm. http://en.wikipedia.org/wiki/Spectral_clustering.