DGA Botnet Detection Utilizing Social Network Analysis

Tzy-Shiah Wang
Institute of Computer and Communication Engineering
Department of Electrical Engineering
National Cheng Kung University
Tainan City, Taiwan
tcwang@nsda.ee.ncku.edu.tw

Chih-Sheng Lin
Institute of Computer and Communication Engineering
Department of Electrical Engineering
National Cheng Kung University
Tainan City, Taiwan
xaoslin@gmail.com

Hui-Tang Lin
Institute of Computer and Communication Engineering
Department of Electrical Engineering
National Cheng Kung University
Tainan City, Taiwan
htlin@mail.ncku.edu.tw

***Abstract*—Botnets are one of the major threats to network security. A botnet can launch attacks by stealing information, phishing sites, sending spam mail and setting up distributed denial of service (DDoS). Some botnets called Domain Generation Algorithm (DGA) Botnets apply a domain generation algorithm to avoid being detected by the traditional blacklist detection scheme. Using a domain generation algorithm, a DGA bot periodically generates a huge list of candidate Command and Control server (C&C) domains. The bot then attempts to connect to the C&C server by querying DNS servers for the domains on the list one by one until it connects to an existing C&C server. By doing this, DGA botnets become very elusive and difficult to detect by traditional defending systems and thus have high survivability. To resolve this issue, this study proposes a DGA botnet detection mechanism utilizing the feature-based characteristics of social networks. The effectiveness of this mechanism was measured by implementing it in a campus network environment and observing it over eighteen months. The most interesting finding of this experiment is a new class of DGA botnet with a query pattern that has not been detected before. The results show that the proposed mechanism has the ability to accurately and effectively detect both well-known and new malicious DGA botnets in real-world networks.**

Keywords-Botnet; Domain Flux; Domain Generation Algorithm; Social Networks Analysis

I. INTRODUCTION

Today, the proliferation of various online services, such as cloud computing, social media and multimedia, has attracted more and more users to access the Internet. However, most users surf the Internet with little network security awareness, so they frequently fall victim to network security threats. Among all network security issues, botnets are one of the most major threats. A botnet consists of a set of compromised hosts, called bots, that are remotely controlled by a botmaster. A botmaster controls the bots by giving them orders through a Command and Control server (C&C) to launch malicious attacks or perform illegal activities (e.g., spam, private information stealing, DDoS, etc.). Among the various forms of botnets, Domain Generation Algorithm (DGA) botnets are one of the most disruptive and difficult to detect. All bots in a DGA botnet periodically execute a DGA to generate a list of candidate C&C domains. Each bot then performs DNS queries for the domains in the list one by one until it connects to the C&C server. When a domain is detected and blocked by defending systems, the botmaster simply migrates the C&C server by associating it with a new IP and a new domain name in the list of candidate C&C domains. The bots only need to query the domains in the list sequentially until they connect to the C&C server again. By doing so, DGA botnets can effectively conceal their malicious activities and significantly improve their survivability against detection systems [4][5]. Therefore, how to detect DGA botnets has become a very important research issue.

Currently, there are several well-known DGA botnets, such as Kraken [1], Srizbi [1], Mjuyh [1] Conficker-A/B [2] and Conficker-C [3]. In general, each DGA botnet applies a unique DGA algorithm to generate its own list of candidate C&C domains. For instance, Mjuyh uses the domain name "Mjuyh.com" as the Top-Level Domain (TLD) and Second-Level Domain (SLD) and chooses random alphanumeric characters as the third or the fourth-level domain [1]. Meanwhile, the variants of Conficker botnets employ the current Coordinated Universal Time (UTC) as seeds [2][3] to generate the list.

Although DGA botnets are very elusive and difficult to detect, they still leave several clues behind. First of all, the bots of a DGA botnet typically conduct a large number of DNS queries on the same set of domains. Thus, the bots associated with the same DGA botnet exhibit a high degree of similarity between their query behaviors. Secondly, the domain queries from the DGA bots tend to generate lots of Name Error responses [6], called NXDOMAIN responses. This is because at any moment in time, only a small subset of domains in the list is actually associated with the C&C server [7]. When DGA bots sequentially query the domains in their list in an attempt to connect to the C&C servers, they inadvertently generate NXDOMAIN responses because many domain names they query are not bound to any IP address. The domain names that generate NXDOMAIN responses are referred to as NXDomains for the rest of this paper. Figure 1 shows an example that illustrates the query

978-1-5090-3071-2/16 $31.00 © 2016 IEEE
DOI 10.1109/IS3C.2016.93

behaviors on NXDomains from DGA and normal hosts. The circles and squares in Fig.1 represent the hosts and NXDomains, respectively. An edge indicates that the host has queried the NXDomain and received an NXDOMAIN response. Note that the queries generated from DGA bots for NXDomains are highly overlapped if not completely the same, rendering a complete or almost complete bipartite graph as shown in Fig. 1(a). This is intuitive since the bots compromised by the same DGA-based malware tend to query the same set of NXDomains in a short period of time. For normal hosts, the query pattern on NXDomains is quite different from that of the DGA bots. Usually, the query to an NXDomain from a normal user is either due to typos or websites temporarily becoming unavailable. The probability of two different normal users entering the same typo for the same domain name within the same time frame is very small. Although a popular website which temporarily goes down can create a huge amount of NXDOMAIN responses to many hosts accessing that website, such an event occurs with a very low probability, is generally short-lived, and can be detected and ruled out easily. Thus, in general, the queries on NXDomains from normal hosts form a sparser bipartite graph as shown in Fig. 1(b). By contrasting the behaviors of normal and botnet NXDomain queries, it becomes apparent that bots belonging to the same DGA botnet are far more similar in their query behaviors compared to a group of normal hosts. In other words, the graph constructed from the queries of the same group of DGA bots exhibits a unique pattern and can be used to distinguish DGA bots from normal hosts.

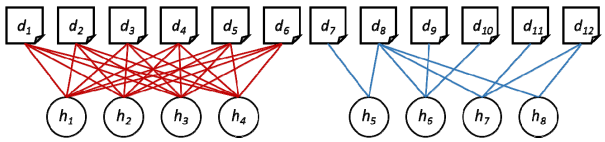

Figure 1. Examples of query behavior on NXDomains generated by (a) DGA bots and (b) Normal hosts.

To detect groups of DGA bots based on the two clues discussed above, social network analysis (SNA) is applied to evaluate the structure of the graph generated by each group. SNA has been considered as a novel measurement tool in networking research [8] for assessing the existence and strength of network structures, node centralities, and network robustness upon node removal. In the past, graphs have been widely used for representing complex objects in many research areas [9], and their structures were often then analysed using SNA. However, to the best of our knowledge, this paper is the first attempt to demonstrate that SNA can be a key tool in effectively detecting DGA botnets. The remainder of this paper is organized as follows: Section II describes the background and the SNA concept that we used in this research. Section III describes the assumptions and presents the proposed algorithm. Section IV shows the simulation results using real world DNS traffic data. Section V draws some brief conclusions.

II. PROPOSED METHOD

This section describes the detection process of the proposed method. The goal here is to analyse a group of hosts to determine whether they are normal or compromised hosts. As shown in Fig. 2, the proposed method consists of a filtering module, a clustering module and a group identification module. The details of each module are discussed in the following paragraphs.

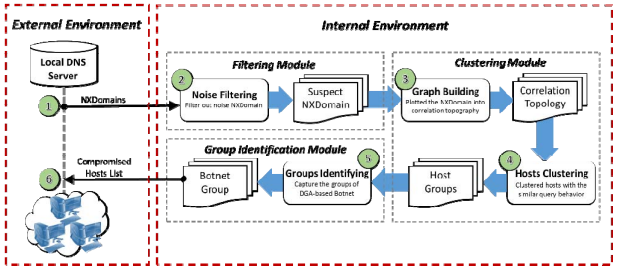

Figure 2. System architecture of the proposed method.

A. Filtering Module

Upon receiving DNS traffic from DNS servers, the proposed scheme commences by discarding the queries of active domains in the DNS traffic using the filtering module. The filtering module then further weeds out any queries on "normal" NXDomains according to a whitelist of normal NXDomains, which is compiled based on four common block list datasets: Spamhaus [10], BRBL [11], SpamCop [12] and AHBL[13]. By removing both active domain queries and queries of normal NXDomains from the log, the number of false alarms and the computation time in the subsequent steps is significantly reduced. (Note that while AHBL is no longer in operation, some mail servers still attempt to use its services.) The remaining NXDomains and hosts are regarded as "Suspected NXDomains and hosts" and are thus passed to the clustering module for further analysis.

B. Clustering Module

The aim of the clustering module is to group hosts compromised by the same DGA algorithm together. Before performing the clustering, an undirected bipartite graph $G = (V, E)$ is constructed from the output of the filtering module, where V is the set of nodes and E is the set of edges. Note that V consists of the sets of suspected NXDomains $D = \{d_i\}_{i=1...M}$ and hosts $H = \{h_j\}_{j=1...N}$ (i.e., $|V|=M+N$). The set of edges $E = \{e_{ij}\}$ denotes the query pattern from the hosts in H to NXDomains in D. Edge e_{ij} represents the host h_j querying the domain d_i and is defined as:

$$e_{ij} = \begin{cases} 1/\alpha & \text{if } h_j \text{ ever queries } d_i \\ 0 & \text{otherwise} \end{cases}, \qquad (1)$$

where α denotes the number of times that the domain d_i is queried by host h_j. The intuition behind this is that the

NXDomains generated by the DGA algorithm are queried only once by each bot in most cases. By contrast, the NXDomains generated as a result of typos or temporarily unavailable websites, for example, are generally queried multiple times since most users either repeat the typos or attempt to reconnect to the chosen website. Thus, we set the value of edge e_{ij} higher when the number of times that host h_j queries domain d_i is smaller.

The clustering process is performed using the Chinese Whispers (CW) clustering algorithm [14] since the algorithm is time-linear with respect to the number of edges and is thus capable of handling very large graphs in a reasonable time. Having performed the clustering process on graph G, the node set V would now be grouped into a number of clusters. The output of the clustering module has the form $H=\{h_k\}_{k=1...K}$, where H is a set of K clusters. The clustering results are then output to the group identification module for final detection.

C. Group Identification Module

The group identification module receives the output of the clustering module and identifies whether or not each candidate cluster is a DGA botnet group by performing a statistical analysis on the cluster. To this end, based on the clustering results, the graph G is partitioned into K subgraphs, one subgraph per cluster. For each subgraph $\tilde{G}_k$, a score function $S(k)$ is computed as follows:

$$S(k) = w \cdot S_D(k) + (1-w) \cdot S_C(k), \quad (2)$$

where $w \geq 0$ is a weighting coefficient. $S_D(k)$, and $S_C(k)$ are the social analysis scores derived from the standard deviation of *Degree centrality* and *Closeness centrality*, respectively. For all the NXDomains and hosts that belong to the subgraph $\tilde{G}_k$, $S_D(k)$ is computed as

$$S_D(k) = \sqrt{1/MN \sum_{i=1}^{M} \left(\frac{A(i,j)}{m_D^k(i)} - 1\right)^2 \sum_{j=1}^{N} \left(\frac{A(i,j)}{m_D^k(j)} - 1\right)^2}, \quad (3)$$

where $m_D^k(i)$ and $m_D^k(j)$ are the mean degrees of all the NXDomain nodes and host nodes in subgraph $\tilde{G}_k$, respectively. The adjacent matrix G, i.e., $A(G)$, has an entry of 1 if two nodes are connected; otherwise, it is 0. $S_C(k)$ is computed as

$$S_C(k) = \sqrt{1/MN \sum_{i=1}^{M} \left(\frac{l(i,j)}{m_C^k(i)} - 1\right)^2 \sum_{j=1}^{N} \left(\frac{l(i,j)}{m_C^k(j)} - 1\right)^2}, \quad (4)$$

where $l(i,j)$ denotes the length of the shortest path between node i and node j while $m_C^k(i)$ and $m_C^k(j)$ represent the mean length of every pair of NXDomain nodes and the mean length of every pair of host nodes. Having determined the value of $S_D(k)$ and $S_C(k)$, the score function $S(k)$ for cluster C_k can then be obtained from Eq.(2). A smaller value of $S(k)$ implies a greater degree of similarity in terms of the query pattern for all the hosts in that cluster. Thus, a lower score indicates that there is a higher probability that a cluster is compromised. In the final classification process, a threshold θ is used to determine whether a cluster is compromised or not. If the score $S(k)$ of a cluster is lower than θ, the cluster is classified as a compromised cluster; otherwise, it is not. In the present study, a threshold value of $\theta = 0.1$ is found to yield an effective detection performance.

III. EVALUATION

In this section, the performance of the proposed scheme is evaluated using real-world DNS traffic. The DNS traffic was generated by more than 10000 network users (including students, faculty and employees) in the education network of Tainan city in Taiwan from March 2014 to August 2015. An initial inspection of the traffic reveals that the volume of queries is about 2.8 million queries per day on weekdays and around 1.6 million per day on weekends or holidays. However, on average, there are only around 600 NXDomains per hour after the DNS traffic is processed by the filtering module. That is, the proposed botnet detection scheme only needs to process about 0.05% of the original traffic.

From the experimental results, a cluster of seven Conficker bots are identified. Examining the NXDomains queries by these bots, it is found that each domain is composed of a random string and a TLD (i.e., net, com, cc, cn, ws, info, biz and org), which is consistent with the characteristic of a Conficker-based DGA botnet. Figure 3 shows a bipartite graph of query behavior which was generated by the Conficker cluster (to protect user privacy, the middle bytes of IP addresses are replaced with alphanumerical letters). Fig. 3 gives a graph constructed using the queries from the seven Conficker bots to their NXDomains within an hour. It is observed that the graph forms a complete bipartite graph in which all the bots query all the NXDomains.

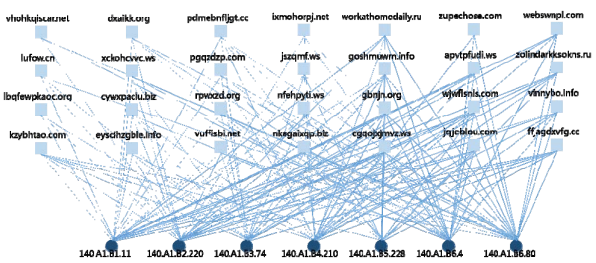

Figure 3. A bipartite graph constructed for the seven Conficker bots and their NXDomains using one-hour sample data.

In the present study, we capture a cluster of new DGA bots and find that their generated NXDomains have similar characteristics to that of Mjuyh bots, but with variations in their subdomains. Recall that the Mjuyh botnet [1] uses the domain name "Mjuyh.com" as the SLD and TLD and uses random alphanumeric characters as the third-level domain. As shown in Fig. 4, the NXDomains generated by the

"Mjuyh-like" hosts comprised a random string with various 3LDs, e.g., yt.liebiao.800fy.com and ef.www.qiyue98.net. This Mjuyh-like botnet was first detected in March 2014. In addition, "liebiao.800fy.com" was also involved in a systematic DDoS attack detected by Nominum in 2014 [15].

mn.liebiao.800fy.com	yt.liebiao.800fy.com	yd.liebiao.800fy.com
sf.liebiao.800fy.com	wl.liebiao.800fy.com	ef.www.qiyue98.net
gt.www.qiyue98.net	yn.liebiao.800fy.com	wx.xin.lyaux.com
sl.liebiao.800fy.com	wx.xin.lyaux.com	cp.xin.lyaux.com
qt.liebiao.800fy.com	cp.xin.lyaux.com	wh.xpiaopiao.com
gr.777.521wooolf.com	wh.xpiaopiao.com	qs.xpiaopiao.com

Figure 4. A sample of domains queried by a cluster of Mjuyh-like hosts

Furthermore, we track the daily query behavior of these Mjuyh-like hosts for three consecutive days, as shown in Fig. 5. As we can see, there is a sudden increase in query volume at 11am-12pm and a marked decline at 10pm-11pm for all three days. A possible explanation for the observed trend is that the bot's query pattern was set to mimic that of real world users. Since most people start working around 8-10am, we suspect that the time zone of the botmaster may be two or three hours later than Taiwan's (which is UTC +8) if the botmaster wrote the algorithm based on the time zone where he/she resides. Based on this theory, the botmaster may reside in the timezones of UTC +5 or +6. However, it is also possible that the botmaster's choice of time zone was arbitrary.

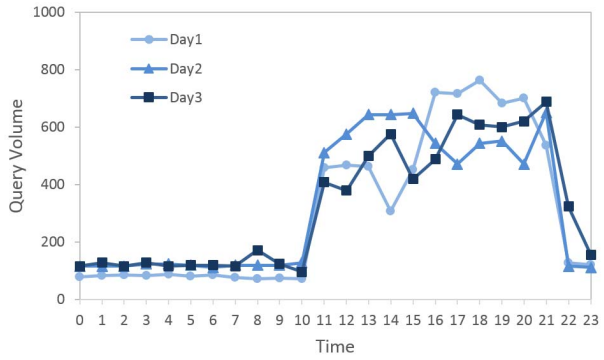

Fig. 5 A sample of daily query behavior generated by a cluster of Mjuyh-like bots over three consecutive days.

IV. Conclusion

In this paper, we propose a defending mechanism for detecting DGA botnets which algorithmically generate domain names to improve their survivability. In the proposed method, the hosts and NXDomains are grouped into clusters in accordance with the query behaviors between them, and each cluster is then identified as either malicious or benign depending on social network analysis. The performance of the proposed method has been evaluated using the traffic obtained over an eighteen-month period from a real-world university campus network. The results have confirmed the ability of our proposed method to detect DGA-based botnets in realistic network environments. Notably, a new botnet has been identified which exhibits Mjuyh-like behavior, using multiple subdomains when generating a list of candidate domains. The results show that our method can actually capture new and known DGA botnets.

Acknowledgment

This work was supported in part by Taiwan Information Security Center (TWISC), Academia Sinica, and Ministry of Science and Technology, R.O.C., under Grant No. MOST 103-2221-E-006-147-MY3 and MOST 104-2218-E-001-002.

References

[1] S. Yadav, A. K. K. Reddy, and S. Ranjan, "Detecting algorithmically generated domain-flux attacks with DNS traffic analysis," IEEE/ACM Transactions on Networking, vol. 20, pp. 1663-1677, 2012.

[2] P. Porras, "Inside risks reflections on Conficker," Communications of the ACM, vol. 52, pp. 23-24, 2009.

[3] P. Porras, H. Saidi, and V. Yegneswaran, "An analysis of Conficker's logic and rendezvous points," in USENIX Workshop on Large-Scale Exploits and Emergent Threats, 2009.

[4] M. Antonakakis, R. Perdisci, Y. Nadji, N. Vasiloglou Ii, S. Abu-Nimeh, W. Lee, and D. Dagon, "From Throw-Away Traffic to Bots: Detecting the Rise of DGA-Based Malware," in USENIX Security Symposium, pp. 491-506, 2012.

[5] R. Sharifnya and M. Abadi, "A novel reputation system to detect dga-based botnets," in 2013 3th International eConference on Computer and Knowledge Engineering (ICCKE), 2013, pp. 417-423.

[6] P. V. Mockapetris, "Domain names-concepts and facilities," [Online]. Available: http://www.ietf.org/rfc/rfc1034.txt

[7] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna, "Your botnet is my botnet: analysis of a botnet takeover," in Proceedings of the 16th ACM conference on Computer and communications security, pp. 635-647, 2009.

[8] D. Katsaros, N. Dimokas, and L. Tassiulas, "Social network analysis concepts in the design of wireless ad hoc network protocols," IEEE Network, vol. 24, pp. 23-29, 2010.

[9] R. Giugno and D. Shasha, "Graphgrep: A fast and universal method for querying graphs," in Proceedings of the IEEE 16th International Conference on Pattern Recognition, pp. 112-115, 2002.

[10] The Spamhaus Project, [Online]. Available: http://www.spamhaus.org/

[11] Barracuda Reputation Block List, [Online]. Available: http://www.barracudacentral.org/

[12] SpamCop, [Online]. Available: https://www.spamcop.net/

[13] The Abusive Hosts Blocking List, [Online]. Available: http://www.ahbl.org/node

[14] C. Biemann, "Chinese whispers: an efficient graph clustering algorithm and its application to natural language processing problems," in Proceedings of the first workshop on graph based methods for natural language processing, pp. 73-80, 2006.

[15] B. Van Nice. DNS-Based DDoS: Diverse Options for Attackers. [Online]. Available: http://www.circleid.com/posts/20150415_dns_