Industrial Science ISSN 2347-5420
Volume-1 | Issue-10 | April-2015 Available online at www.lsrj.in

SCALABLE SYSTEM CONSTRUCTION FOR STEALTHY PEER-TO-PEER BOTNET DETECTION

Andrea Noreen D silva
PG Scholar , Department of Information Science Engineering,
Dr Ambedkar Institute of Technology,Bangalore.

Short Profile

Andrea Noreen D silva is PG Scholar at Department of Information Science Engineering,Dr Ambedkar Institute of Technology in Bangalore.

Co- Author Details :

Vidyarani H. .J
Assistant Professor , Department of Information Science Engineering,Dr Ambedkar Institute of Technology,Bangalore.

ABSTRACT:
Peer-to-peer (P2P) botnets have recently been adopted by botmasters for their resiliency against take-down efforts. Besides being harder to take down, modern botnets tend to be stealthier in the way they perform malicious activities, making current detection approaches ineffective. In addition, the rapidly growing volume of network traffic calls for high scalability of detection systems. In this paper, we propose a novel scalable botnet detection system capable of detecting stealthy P2P botnets. Our system first identifies all hosts that are likely engaged in P2P communications. It then derives statistical fingerprints to profile P2P traffic and further distinguish between P2P botnettraffic and legitimate P2P traffic. The parallelized computation with bounded complexity makes scalability a built-in feature of our system. Extensive evaluation has demonstrated both high detection accuracy and great scalability of the proposed system.

KEYWORDS
Scalable System Construction , communications , parallelized computation .

I.INTRODUCTION

As the fast development of information technology, the population of using the Internet increases rapidly. Due to the prevalence of social networks, a huge amount of data are transmitted via networks in every second, causing the potential risks of disclosing personal information. One can see on television or from newspapers about the news of hackers stealing confidential data for illegal usage, and they may use a variety of methods such as Distributed Denial of Service (DDoS), Spam and Trojan. These methods require the cooperation of many computers, so hackers often spread out malicious software to achieve the goal of attacks. Therefore, it is important to enhance network security while developing and applying new information technologies to prevent from illegal access to confidential information.A botnet is a collection of software agents, or robots, that run autonomously and automatically.

The term is most commonly associated with IRC botnets and more recently malicious software, but it can also refer to a computer network using distributed computation software. Botnets are usually named after their malicious software, such as Peacomm and Waledac. Basically, the composition of a botnet includes: the server programs used to control the infected computers, the client programs installed on the infected computers waiting for the control instructions, and the malicious software to infect normal computers as zombie computers to steal confidential data. The above programs often use a unique encryption system to communicate with each other to prevent from detection.

The data-transmission power of P2P architecture usually depends on its number of connections; a larger connection number means a larger network space and shared bandwidth. The ability of P2P networks in bearing workload and its high transmission speed facilitates network sharing, but it also causes serious network security problems. To provide a number of users with network connections and shared resources, the P2P architecture usually adopts open-source software, and some extensive and fault-tolerant programs can easily be revised as virus programs for extending the coverage of botnets.

The nodes in a P2P network are usually connected through an ad hoc network , and the main idea is to form a logical network through the existing physical network, rather than reconstructing a new physical network. No matter what kind of logical network structure is adopted, the clients still have to transfer data through the physical layer. For simplicity, the installation of P2P software is usually done by clicking on a button. When the users are not familiar with its setting, their personal information can easily be exposed to others on the Internet. Besides, their computers may be infected by botnet viruses if they don't have the sense of network security or there is no anti-virus software installed.

1.1 Existing System

In the existing system,a few approaches capable of detecting P2P botnets have been proposed . Compared with the existing methods , the design goals of our approach are different in that: 1) our approach does not assume that malicious activities are observable, unlike in the existing system; 2) our approach does not require any botnet-specific information to make the detection, unlike 3) our approach needs to detect the compromised hosts that run both P2P bot and other legitimate P2P applications at the same time, unlike in the existing system different from the existing system, our approach has high scalability as a built-in feature. Other methods use machine learning for detection, which require labeled P2P botnet data to train a statistical classifier. Unfortunately, acquiring such information is a challenging task, thereby drastically limiting the practical use of these methods.

1.2 Proposed System

In the proposed system, we present a novel scalable botnet detection system capable of detecting stealthy P2P botnets. We refer to a stealthy P2P botnet as a P2P botnet whose malicious activities may not be observable in the network traffic. Particularly, our system aims to detect stealthy P2P botnet even if P2P botnet traffic is overlapped with traffic generated by legitimate P2P applications (e.g., Skype) running on the same compromised host and ii) achieve high scalability. To this end, our system identifies P2P bots within a monitored network by detecting the C&C communication patterns that characterize P2P botnets, regardless of how they perform malicious activities in response to the botmaster's commands. Specifically, it derives statistical fingerprints of the P2P communications generated by P2P hosts and leverages them to distinguish between hosts that are part of legitimate P2P networks (e.g., filesharing networks) and P2P bots. The high scalability of our system stems from the parallelized computation with bounded computational complexity.

2.SYSTEM DESIGN

System Overview: A P2P botnet relies on a P2P protocol to establish a C&C channel and communicate with the botmaster.Therefore P2P bots exhibit some network traffic patterns that are common to other P2P client applications (either legitimate or malicious). Thus, we divide our systems into two phases.

In the first phase, we aim at detecting all hosts within the monitored network that engage in P2P communications. We analyze raw traffic collected at the edge of the monitored network and apply a pre-filtering step to discard network flows that are unlikely to be generated by P2P applications. We then analyze the remaining traffic and extract a number of statistical features to identify flows generated by P2P clients. In the second phase, our system analyses the traffic generated by the P2P clients and classifies them into either legitimate P2P clients or P2P bots. Specifically, we investigate the active time of a P2P client and identify it as a candidate P2P bot if it is persistently active on the underlying host. We further analyze the overlap of peers contacted by two candidate P2P bots to finalize detection.

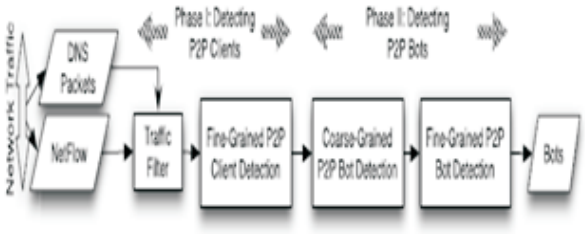


Fig 1:System Overview

2.1.Traffic Filter

The Traffic Filter component aims at filtering out network traffic that is unlikely to be related to P2P communications. This is accomplished by passively analysing DNS traffic, and identifying network flows whose destination IP addresses were previously resolved in DNS responses.

Specifically, we leverage the following feature: P2P clients usually contact their peers directly by looking up IPs from arouting table for the overlay network, rather than resolving a domain name.

Since most non-P2P applications (e.g., browsers, email clients, etc.) often connect to a destination address resulting from domain name resolution, this simple filter can eliminate a very large percentage of non-P2P traffic, while retaining the vast majority of P2P communications.

2.2.Fine-Grained Detection of P2P Clients

This component is responsible for detecting P2P clients by analyzing the remaining network flows after the Traffic Filter component. For each host h within the monitored network we identify two flow sets, denoted as Stcp(h) and Sudp(h), which contain the flows related to successful outgoing TCP and UDP connection, respectively. We consider as successful those TCP connections with a completed SYN, SYN/ACK, ACK handshake, and those UDP (virtual) connections for which there was at least one "request" packet and a consequent response packet.

In order to detect P2P clients, consider the fact that each P2P client frequently exchanges control messages(e.g., ping/pong messages) with other peers. Besides, we notice that the characteristics of these messages, such as the size and frequency of the exchanged packets, are similar for nodes in the same P2P network, and vary depending on the P2P protocol and network in use. As a consequence, if two network flows are generated by the same P2P application and they carry the same type of P2P control messages, they tend to share similar flow size. In addition, a P2P client will exchange control messages with a large number of peers distributed in many different networks. Consequently, the destination IP addresses of network flows that carry these control messages will spread across a large number of networks where each network can be represented by its BGP prefix.

To identify flows corresponding to P2P control messages, we first apply a flow clustering process intended to group together similar flows for each candidate P2P node h.

Given sets of flows Stcp(h) and Sudp(h), we characterize each flow using a vector of statistical features v(h) =[Pkts , Pktr , Bytes , Byter], in which Pktsand Pktrrepresent the number of packets sent and received, and Bytes and Byterrepresent the number of bytes sent and received, respectively. The distance between two flows is subsequently defined as the euclidean distance of their two corresponding vectors. We then apply a clustering algorithm to partition the set of flows into a number of clusters. Each of the obtained clusters of flows, Cj (h), represents a group of flows with similar size. For each Cj (h), we consider the set of destination IP addresses related to the flows in the clusters, and for each of these IPs we consider its BGP prefix (using BGP prefix announcements).

Finally, we count the number of distinct BGP prefixes related to destination IPs in a cluster bgpj= BGP(Cj (h)), and discard those clusters of flows for which bgpj< _bgp.We call fingerprint clusters the remaining cluster of flows. Therefore,each host h can now be described by a set of fingerprint clusters FC(h) = {FC1, . . . , FCk}. We label h as P2P node if FC(h) _= ∅, namely if h generated at least one fingerprint cluster.

Figure 2 illustrates an example of the flow clustering process for a P2P node. Flows corresponding to ping/pong and peer-discovery share similar sizes, andhence they are grouped into two clusters (FC1 and FC2), respectively. Since the number of destination BGP prefixes involved in each cluster is larger than _bgp, we take FC1 and FC2 as its fingerprint clusters. A fingerprint cluster summary, (Pkts,Pktr, Bytes , Byter, proto), represents the protocol and the average number of sent/received packets/bytes for all the flows in this fingerprint cluster.

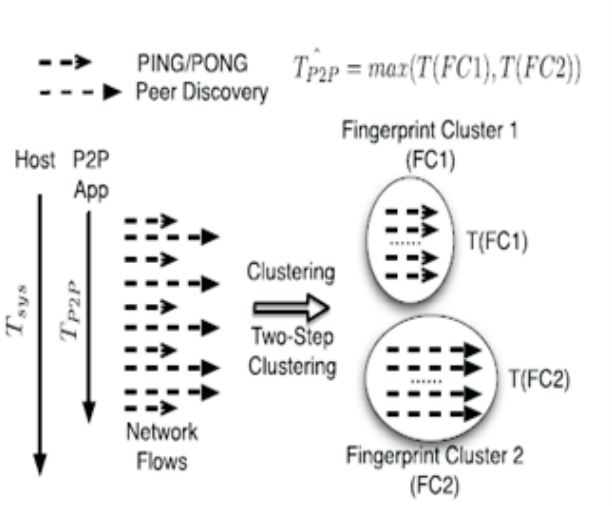


Fig 2:Example of flow clustering to identify p2p hosts

3 SYSTEM IMPLEMENTATION

The implementation objective is to integrate high scalability as a built-in feature into our system.

3.1 Performance Bottleneck

Out of four components in our system, “Traffic Filter” and “Coarse-Grained Detection of P2P Bots” have linear complexity since they need to scan flows only once to identify flows with destination addresses resolved from DNS queries or calculate the active time. Other two components, “Fine-Grained Detection of P2P Clients” and “Fine-Grained P2P Detection of P2P Bots”, require pairwise comparison for distance calculation. Specifically, if we denote the number of flows generated by a host as n and the number of hosts as S, the time complexity of Fine-Grained Detection of P2P Clients approximates O(S∗n2). Comparably, if we denote the number of persistent P2P clients as l, the time complexity of Fine- Grained P2P Bot Detection approximates O(l2). Since the number of flows generated by network applications (i.e., n)

could be enormous (e.g., more than hundreds of thousands of flows are generated by a single P2P client in our experiments), the computation overhead of Fine-Grained Detection of P2P Clients may become prohibitive. On contrary, the percentage of P2P clients in the ISP network is relatively small. Fine- Grained P2P Bot Detection is unlikely to introduce huge performance overhead.

3.2 Two-Step Flow Clustering

We use a two-step clustering approach to reduce the time complexity of “Fine-Grained P2P Client Detection”. For thefirst-step clustering, we use an efficient clustering algorithm to aggregate network flows into K sub-clusters, and each subcluster contains flows that are very similar to each other. For the second-step clustering, we investigate the global distribution of sub-clusters and further group similar sub-clusters into clusters.

The distance of two flows is defined as the Euclidean distance of their corresponding vectors, where each vector [Pkts ,Pktr , Bytes , Byter] represents the number of packets/ bytes that are

sent/received in a flow.For the second-step clustering, we use hierarchical clustering with DaviesBouldin validation to group sub-clusters into clusters. Each sub-cluster is represented using a vector ([Pkts ,Pktr , Bytes , Byter]), which is essentially the average for all flow vectors in this sub-cluster. Hierarchical clustering is used to build a dendrogram.

4 .WORKING

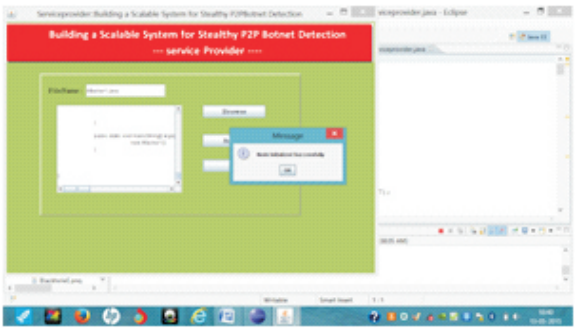
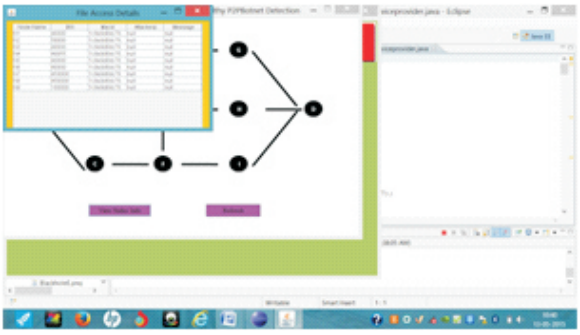


Fig 4.1At the service provider



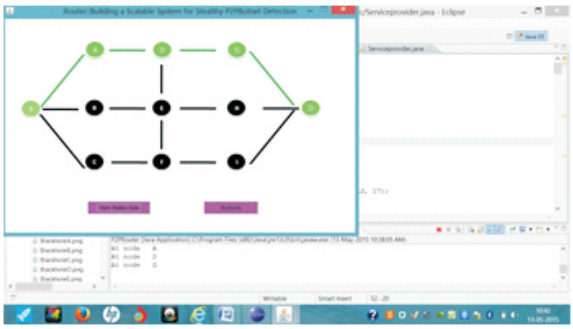


Fig 4.2 At the router

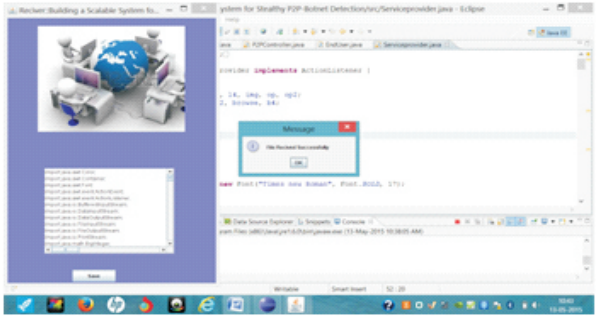


Fig 4.3 At the end user

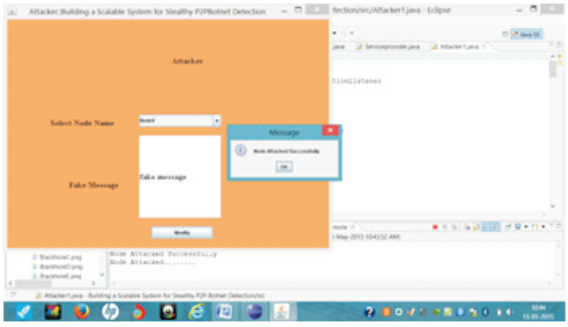


Fig 4.4 Attacker 1

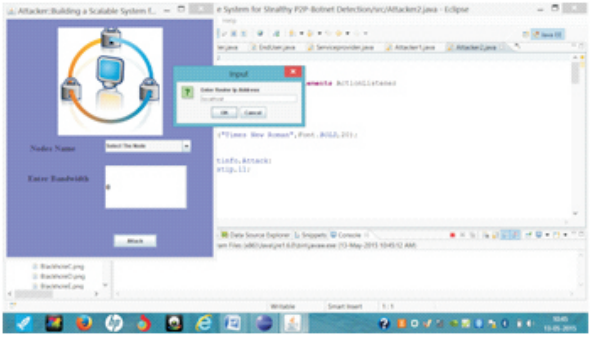


Fig 4.5 Attacker 2

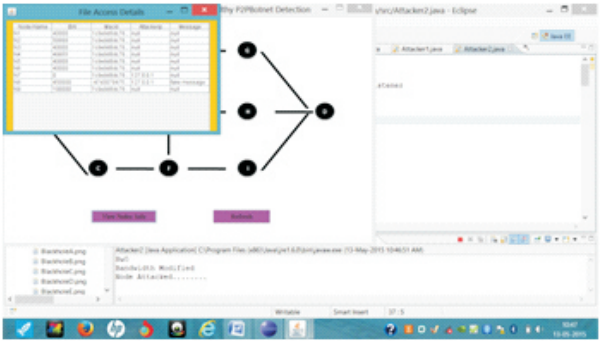


Fig 4.6 Modified details at router

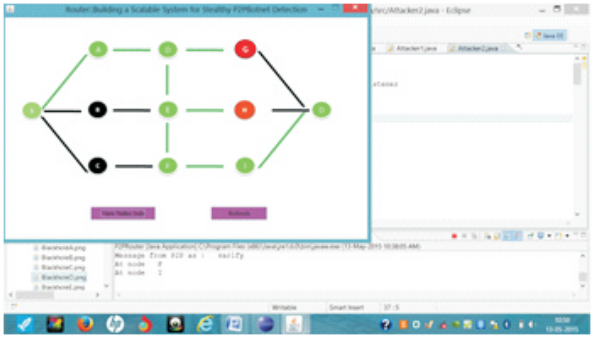


Fig 4.7 At the router

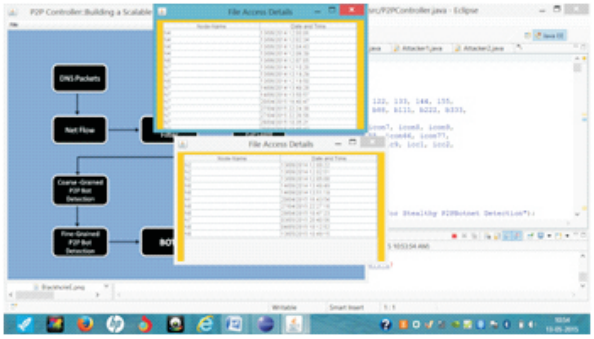


Fig 4.8 At the Peer to Peer Controller

5 POSSIBLE EVASIONS AND SOLUTIONS

If botmasters get to know about our detection algorithm, they could attempt to modify other bots' network behavior to evade detection. This situation is similar to evasion attacks against other intrusion detection systems.

Evasion by Misusing the Traffic Filter

Botmasters may exploit our traffic filter for evasion. For example, a botmaster may set up a DNS server and then instruct each bot to query this server before contacting any peer. The (malicious) DNS server can respond the bot with a DNS response that contains the IP address of that peer.

In this case, our traffic filter could eliminate bot flows. To solve this problem, we can only filter out network flows whose destination IP addresses are resolved from popular domains, i.e., domains queried by a non-negligible fraction of hosts in the monitored networks.

Evasion by Joining Legitimate P2P Applications

Bots may "blend" with legitimate P2P clients to evade detection by directly joining legitimate P2P networks, in which botmasters can propagate commands. In fact, the initial versionof Storm adopted this "blending" strategy, while recent P2P botnets, including the most recent version of Storm, Waledac, and Zeus, build their own P2P networks. Nevertheless, the "blended" P2P bots may make our detection more difficult because bots' fingerprint clusters will be identical to those of legitimate P2P applications. However, since two bots will search for the same commands and thus their fingerprint clusters are still likely to have a large overlap of contacted peers, which may serve as a discriminative feature by itself. In spite of that, we acknowledge that "blended" P2P bots make our detection very challenging in certain circumstances. For example, when two P2P bots relay queries from legitimate peers, their fingerprint clusters are unlikely to have large peeroverlaps.

Evasion by Randomizing Communication Behaviours

Bots could randomize their P2P communication patterns to prevent our system from getting accurate fingerprint clusters for P2P protocols. In this case, we can measure the number of failed connections to perform coarse-grained detection of P2P clients and also adopt more general features, such as the distribution of flow/packet sizes and flow/packet intervals, to profile P2P applications.

To summarize, although our system greatly enhances and complements the capabilities of

existing P2P botnet detection systems, it is not perfect. In this project we should definitely strive to develop more robust defense techniques, where the aforementioned discussion outlines the potential improvements of our system.

6 CONCLUSION

In this system, we presented a novel botnet detection system that is able to identify stealthy P2P botnets, whose malicious activities may not be observable. To accomplish this task, we derive statistical fingerprints of the P2P communications to first detect P2P clients and further distinguish between those that are part of legitimate P2P networks (e.g., file sharing networks) and P2P bots. We also identify the performance bottleneck of our system and optimize its scalability. The evaluation results demonstrated that the proposed system accomplishes high accuracy on detecting stealthy P2P bots and great scalability.

REFERENCES

[1] S. Stover, D. Dittrich, J. Hernandez, and S. Dietrich, "Analysis of thestorm and nugachetrojans: P2P is here," in Proc. USENIX, vol. 32.2007, pp. 18–27.
[2] P. Porras, H. Saidi, and V. Yegneswaran, "A multi-perspective analysisof the storm (peacomm) worm," Comput. Sci. Lab., SRI Int., MenloPark, CA, USA, Tech. Rep., 2007.
[3] P. Porras, H. Saidi, and V. Yegneswaran. (2009). Conficker C Analysis[Online]. Available: http://mtc.sri.com/Conficker/addendumC/index.html
[4] G. Sinclair, C. Nunnery, and B. B. Kang, "The waledac protocol: Thehow and why," in Proc. 4th Int. Conf. Malicious Unwanted Softw.,Oct. 2009, pp. 69–77.
[5] R. Lemos. (2006). Bot Software Looks to Improve Peerage [Online].Available: http://www.securityfocus.com/news/11390
[6] Y. Zhao, Y. Xie, F. Yu, Q. Ke, and Y. Yu, "Botgraph: Largescale spamming botnet detection," in Proc. 6th USENIX NSDI, 2009,pp. 1–14.
[7] G. Gu, R. Perdisci, J. Zhang, and W. Lee, "Botminer: Clusteringanalysis of network traffic for protocol- and structure-independent botnetdetection," in Proc. USENIX Security, 2008, pp. 139–154.
[8] T.-F. Yen and M. K. Reiter, "Are your hosts trading or plotting?Telling P2P file-sharing and bots apart," in Proc. ICDCS, Jun. 2010,pp. 241–252.
[9] S. Nagaraja, P. Mittal, C.-Y. Hong, M. Caesar, and N. Borisov, "BotGrep:Finding P2P bots with structured graph analysis," in Proc. USENIXSecurity, 2010, pp. 1–16.
[10] J. Zhang, X. Luo, R. Perdisci, G. Gu, W. Lee, and N. Feamster,"Boosting the scalability of botnet detection using adaptive trafficsampling," in Proc. 6th ACM Symp. Inf., Comput. Commun. Security,2011, pp. 124–134.
[11] J. Zhang, R. Perdisci, W. Lee, U. Sarfraz, and X. Luo, "Detectingstealthy P2P botnets using statistical traffic fingerprints," in Proc.IEEE/IFIP 41st Int. Conf. DSN, Jun. 2011, pp. 121–132.
[12] S. Saad, I. Traore, A. Ghorbani, B. Sayed, D. Zhao, W. Lu, et al.,"Detecting P2P botnets through network behavior analysis andmachine learning," in Proc. 9th Annu. Int. Conf. PST, Jul. 2011,pp. 174–180.