Available online at www.elixirpublishers.com (Elixir International Journal)

Digital Processing

Elixir Digital Processing 96 (2016) 41767-41770

ISSN: 2229-712X

A Novel Botnet Detection System to Identify Resilient P2P-Botnet

Khalid Sheik and Kameswara Rao

Department of Information Technology, SRKR Engineering College, Bhimavaram, India.

ARTICLE INFO

Article history:

Received: 21 June 2015;

Received in revised form:

17 July 2016;

Accepted: 23 July 2016;

Keywords

Botnet,

Network security, P2P,

Intrusion detection.

ABSTRACT

Peer-to-peer (P2P) botnets are the modern and most resilient bot structures which are harder to take down and stealthier to detect their malicious activities, because of which these are adopted by many of the recent botmasters. In this paper, we propose a novel botnet detection system which is capable to identify resilient P2P botnets. Our system initially identifies the p2p communication hosts present in the network. It then derives p2p traffic and further distinguishes between the botnet generated traffic and legitimate generated traffic. The parallelized computation makes scalability a default feature of our system. High detection accuracy and prodigious scalability are the extra features of our proposed system.

Introduction

A BOTNET is a group of conceded hosts which are called as bots, which are controlled through command and control channel (C&C) by an attacker. Botnets aids as infrastructure to many of the cyber-crimes, such as distributed denial-of-service (DDoS) attacks, spamming, click frauds, identity theft etc. The C&C channel is the main component in botnet because botmasters depends on this channel to send commands to their bots and to receive information from the compromised hosts or machines. The C&C channels used in botnets are of different types. In centralized architecture, all the bots in a botnet will be controlled or contact by one or few C&C servers which are under botmaster. Conversely, a fundamental disadvantage of this architecture is that they have single point of failure. In order to overcome this problem, botmasters have started to build P2P bonnets by using more resilient C&C architecture. Bots belonging to P2P botnet form a network such that any node can be used by botmaster to send commands or receive information from other peers.Some of the examples of P2P botnets are Waledac [1], Strom [2], Nugache [3], and even confiker are some interested botnets because they have used P2P C&C architecture as the primary ways to establish and spread across network. These botnets are more complex and more costly to manage compared to centralized botnets,P2P botnsts are more stealthy against take down efforts (by law and enforcement) so even if large number of bots in P2P are detected and taken down still the remaining bots may be able to communicate with each other and to botmaster.

Detecting botnets is of great importance. Yet, designing an operational P2P botnet detection system is challenged with several constraints. First, we have to capture the P2P traffic but P2P communication and file sharing applications, such as skype, Bittorrent and emule also generates high traffic so traffic of P2P botnets can be easily hide or blend into the background P2P traffic.This challenge is faced by one more constraint as the fact that bot or compromised host can behave both legitimate and botnet because due to coexistence of both P2P sharing application and P2P bot on the same host. Second , modern botnets started using more resilient ways to perform malicious activities that are very hard to capture or observe in the network traffic. Example, some botnets will send spam messages through very popular web services like Hotmail, which is likely transparent and seems legitimate to network detectors because of overlap with legitimate email patterns. Third, the network traffic will increase day by day so our deployed system should be capable to process huge amount of traffic efficiently.

To date, a few approaches have been proposed capable of detecting P2P botnets [4]-[6].However, these approaches cannot solve all the above mentioned challenges. For example, In BotMiner if the bots share similar communication patterns belongs to same botnet and performs similar malicious activities like spamming, exploiting, scanning etc. then it identifies as botnets. Unfortunately, malicious activities of bots may be resilient there by non-observable and making BotMiner ineffective. In addition the scalability of BotMiner is constrained [7]. Yen et al. [5] has proposed an algorithm which will differentiate legitimate P2P application and P2P bot. However, this algorithm did not consider the fact that same host can behave both as legitimate host and bot rendering the algorithm ineffective. BotGrep[6] collects network flows over multiple large networks(e.g., ISP network) ,and try to detect P2P botnets by analyzing the communication graph formed by overlay network. Even though BorGrep do not rely on malicious activities for detection, it requires prior detection results to bootstrap the detection and also needs a global view of internet traffic. Nevertheless, it is very difficult to acquire such prior information in practice.

In this paper, we present a novel detection system to identify the resilient (stealthy) P2P botnets. We refer to a resilient P2P botnets to those whose malicious activities may not be observable in network traffic. Principally, our system main aim is to detect resilient P2P botnets even if botnets generated traffic is overlapped with legitimate P2P applications(e.g. Skype) running on same compromised host and to also aims to achieve high scalability. Our system, regardless of how bots perform malicious activities in response to botmaster's commands, It identifies P2P bots from monitored network with the help of C&C communication patterns that characterize P2P botnets. Basically, it derives statistical fingerprints from all P2P applications and leverage them to differentiate between hosts that are part of legitimate P2P networks (e.g. file sharing

Tele:

E-mail addresses: reach2khalid@yahoo.com

networks) and P2P bots. To summarize, our work makes the following contributions:

1. Identifying hosts that interact in P2P communication by using a new flow-clustering based analysis.
2. Estimating active time of various P2P applications by using an efficient algorithm for P2P traffic profiling, where we build a statistical fingerprints.
3. A P2P botnet detection system that notices stealthy P2P bots even though the P2P botnet traffic is overlapped with traffic generated through professional P2P purposes (e.g., Skype) running on same compromised machine.
4. A scalable design based on parallelized computation and efficient detection algorithm.
5. A prototype system, which has demonstrated great scalability and high detection accuracy.

We have made these improvements in current system compare to some other P2P detection systems published earlier. First, we removed coarsgrained analysis component(two level P2P client detection component) to simplify the design and replaced with single level P2P client detection to reduce storage cost by 60% because it eradicates the necessity of keeping failed connections as shown in figure1filteing In-active P2P client. Second, decreasing the processing time by at least 50% by redesigning the clustering based P2P client detection algorithm. Third, parallelizing our system to boost its efficiency and scalability are few important improvements we made in current system compared to older exiting systems.

Related work

The Design Goals of our approach are different when compared to existing approaches of detecting P2P botnets: 1) our approach does not require any previous botnet information to make detection, unlike [6]; 2) our approach does not assume that malicious activities that botnets performs are observable, unlike [4]; 3) our approach should be able to detect compromised hosts that run both legitimate P2P application and P2P bot at same time, unlike[5]: 4) different from [4]-[6], our approach is having high scalability as built-in feature. Other methods [9]-[11] use machine learning for detection, which require previous or labeled P2P botnet data to train statistical classifier. Regrettably, acquiring such information is challenging and may not possible, thereby significantly limiting the practical use of these methods.

To achieve the abovementioned design goals, our system has multiple components. The first one is flow clustering –based analysis approach to identify hosts which are running P2P applications. In divergence to existing approaches of identifying hosts running P2P applications[12]-[16] our approach differs in fallowing ways: 1) our approach does not rely on fixed source port used by [12],[14], which can be easily violated by P2P applications; 2) unlike [13], our approach does not need content signature because encryption makes content signature useless; 3) our approach do not need training data set to build a machine learning based model as used in[15], because it is very challenging to get such information before they are detected; 4) in divergence to [16] identifying a specific P2P application, or approach can detect and profile various P2P applications; 5) our approach can estimate active time of a P2P application which is very important in botnet detection, active time of a bot should be comparable with the active time of the underlying compromised system If this was not the case, the botnet overlay network would risk degenerating into a number of disconnected sub networks.

System design

System Overview: P2P bots exhibits some network traffic patterns that are common to other P2P client applications because a P2P botnet relies on P2P protocol to establish a channel with botmaster through command and control C&C channel. Thus, we divide our system in two phases. In the first phase, we aim to detect all hosts in our monitored network which are engaged in P2P communication as shown in fig. 1, we analyze raw traffic collected at edges of monitored network and apply pre-filtering to discard the network flows that are unlikely to be generated by non P2P applications. We then examine and mine a number of statistical features to identify flows generated by p2p clients from remaining traffic. In the second phase, our system will differentiate from legitimate P2P clients to P2P bots by analyzing traffic generated by P2P clients. Specifically, we identify as a candidate P2P bot if it is persistently active on the underlying host by considering the active time of p2P clients. We further analyze the overlap of peers contacted by two candidate P2P bots to finalize detection.

Table 1. Notations and Descriptions

Notation	Description
Tp2p	The active time of P2P application
NO-DNS peers	The percentage of flows associated with no domain names
Nclust	The number of clusters left by enforcing Θbgp and Θp2p
Nbgp	The largest number of unique bgp prefixes in one cluster
Tp2p	The estimated active time for p2p application

Identifying P2P Clients

Traffic Filter

The traffic Filter component aims at filtering out network traffic that's unlikely to be related to P2P communications. This is entire by means of passively inspecting DNS traffic, and identifying network flows whose destination IP addresses have been beforehand resolved in DNS responses. Specially, we leverage the following features: P2P clients on the whole contact their peers immediately with the aid of looking up IPs from a routing table for the overlay network, as an alternative than resolving a domain name. This option is supported by using table II (No-DNS peers), which illustrates that the sizeable majority of flows generated by P2P applications do not have destination IPs resolved from domain names. The remaining small fraction of flows are akin to a viable exception that a peer bootstraps into a P2P network by looking up domain names that resolve to stable super-nodes) when you consider the most non-P2P applications (e.g., browsers, email clients ettc.)Often connect with a destination address resulting from domain name resolution, this simple filter can eradicate very huge fraction of non-P2P traffic, while recollecting the vast majority of P2P communication.

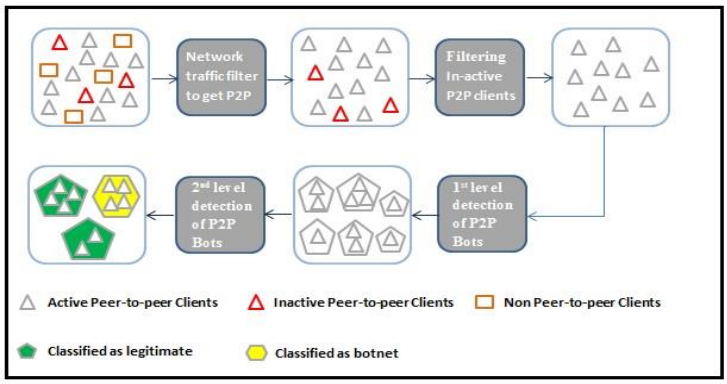

Figure 1. System overview

Identification of active P2P Clients

After traffic Filter component remaining flow will be analyzed to detect P2P clients. For every host h within the monitored network we establish two flow sets, denoted as Stcp(h) and Sudp(h), which contain the flows related to successful outgoing TCP and UDP connection, respectively. We recall as triumphant those TCP connections with a accomplished SYN, SYN/ACK, ACK handshake, and those UDP (virtual) connections for which there used to be at least one "request" packet and a consequent response packet.

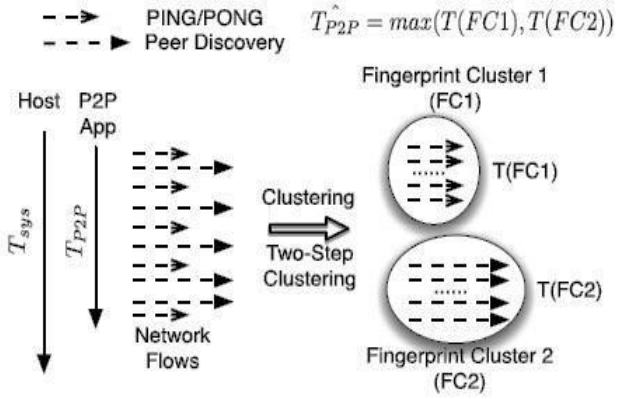

Figure 2. Example of flow clustering to identify P2P hosts

In order to detect P2P clients, we first bear in mind the actual fact that every P2P clients most often exchanges control messages (e.g., ping/pong messages) with other peers. Besides, we discover that the properties of these messages, corresponding to the size and frequency of the exchanged packets, are an identical for nodes in the same P2P network, and differ depending on the P2P protocol and network in use. As a outcome, if two network flows are generated by same P2P application and they carry same type of P2P control messages, they tend to share identical flow size. Additionally, a P2P client will exchange messages with a colossal number of peers disbursed 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 equivalent to P2P control messages, we first apply a flow clustering process envisioned to group together related flows for each candidate P2P node h. Given sets of flows Stcp(h) and Sudp(h), we distinguish each flow using a vector of statistical features v(h) =[Pkts , Pktr , Bytes , Byter], in which Pkts and Pktr mean the number of packets sent and received, and Bytes and Byter mean the number of bytes sent and received, similarly. The distance between two flows is consequently 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). In the end, we rely the number of designated BGP prefixes associated to destination IPs in a cluster bgpj = BGP(Cj (h)), and discard those clusters of flows for which bgpj < Θbgp.We call fingerprint clusters the rest cluster of flows. Hence, each and every host h can now be described by means of 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.

Identification of P2P Bots

First level Detection of P2P Bots

Since bots are malicious programs used to perform profitable malicious activities; they represent valuable assets for the botmaster, who will intuitively try to maximize utilization of bots. This is particularly true for P2P bots because in order to have a functional overlay network (the botnet), an enough number of peers needs to be always online. In other words, the active time of a bot should be comparable with the active time of the underlying compromised system. If this was not the case, the botnet overlay network would risk degenerating into a number of disconnected sub networks due to the short life time of each single node. In contrast, the active time of legitimate Hence, the first component in the "Phase II" of our system ("Coarse-Grained Detection of P2P Bots") aims at identifying P2P clients that are active for a time TP2Pclose to the active time Tsys of the underlying system they are running on. While this behavior is not unique to P2P bots and may be representative of other P2P applications (e.g., Skype clients that run for as long as a machine is on), identifying persistent P2P clients takes us one step closer to identifying P2P bots.

Second level detection of P2P bots

This component is used to identify P2P bots from all persistent P2P clients (i.e. set P). We leverage one feature: the overlap of peers contacted by two P2P bots belonging to the same P2P botnet is much larger than that contacted by two clients in the same legitimate P2P network. Assume that two hosts in the monitored network, say hA and hB, are running the same legitimate P2P file-sharing application (e.g., Emule).). Users of these two P2P clients will most likely have uncorrelated usage patterns. It is realistic to assume that in the general case the two users will search for and download different content (e.g., different media files or documents) from the P2P network. This translates into a deviation between the set of IP addresses contacted by hosts hA and hB. The reason is that the two P2P clients will tend to exchange P2P control messages (e.g., ping/pong and search requests) with different sets of peers which "own" the content requested by their users, or peers that are along the path towards the content. On the contrary, if hA and hB are compromised with P2P bots, one very common characteristic of bots is that they need to periodically search for commands published by the botmaster. This typically translates into a convergence between the set of IPs contacted by hA and hB

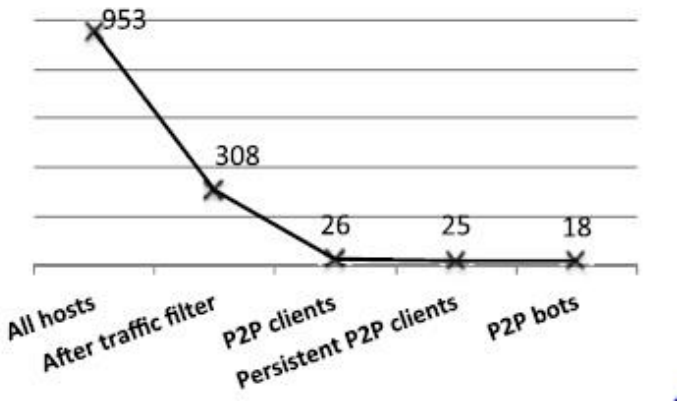

Figure 3. Number of hosts identified by each component

System implementation

The implementation goal is to integrate high scalability as a constructed-in feature into our procedure. To this end, we first identify the performance bottleneck of our approach after which mitigate it utilizing complexity reduction and Parallelizing of system.

Performance Bottleneck

Out of four components in our system, "Traffic Filter" And "First Level Detection of P2P Bots" have linear Complexity seeing that they have got to scan flows only once to identify flows with destination addresses resolved from DNS queries or calculate the active time. Other two components, "Detection of P2P clients" and "second-level Detection of P2P Bots", require pair wise assessment for distance calculation. Above all, if we denote the number of flows generated through a bunch as n and the number of hosts as S, the time complexity of Detection of P2P clients approximates $O(S*n2)$. Comparably, if we denote the number of persistent P2P clients as l, the time complexity of second-level Bot Detection approximates $O(l2)$.

Two-Step Flow Clustering

We use a dual clustering process to scale back the time complexity of "P2P client Detection". For the first-step clustering, we use an effective clustering algorithm to aggregate network flows into k sub-clusters, and each sub cluster contains flows which might be very similar to each and every other. For the second-step clustering, we examine the global distribution of sub-clusters and extra workforce an identical sub-clusters into clusters.

In the current design, we employ K-means as the first step clustering. For the second-step clustering, we use hierarchical clustering with DavisBouldin validation [10] to group sub-clusters into clusters.

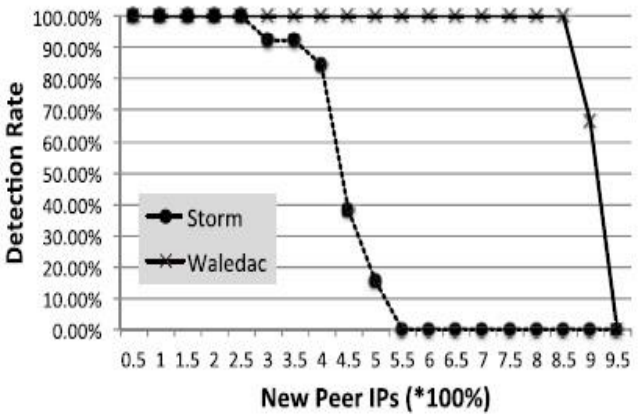

Figure 4. Challenges for attackers to instruct bots to contact different peers to

Conclusion

In this paper, we provided a novel botnet detection process that's able to identify resilient P2P botnets, whose malicious activities may not be identifiable. To achieve this challenge, we derive statistical fingerprints of the P2P communications to first discover P2P clients and further distinguish between those that are part of legitimate P2P networks (e.g., file sharing networks) and P2P bots. We also establish the efficiency bottleneck of our method and optimize its scalability. The analysis results established that the proposed approach accomplishes high accuracy on detecting stealthy P2P bots and satisfactory scalability.

References

[1]G. Sinclair, C. Nunnery, and B. B. Kang, "The waledac protocol: The how and why," in Proc. 4th Int. Conf. Malicious Unwanted Softw., Oct. 2009, pp. 69–77.

[2]P. Porras, H. Saidi, and V. Yegneswaran, "A multi-perspective analysis of the storm (peacomm) worm," Comput. Sci. Lab., SRI Int., Menlo Park, CA, USA, Tech. Rep., 2007.

[3]R. Lemos. (2006). Bot Software Looks to Improve Peerage [Online]. Available: http://www.securityfocus.com/news/11390

[4] G. Gu, R. Perdisci, J. Zhang, and W. Lee, "Botminer: Clustering analysis of network traffic for protocol- and structure-independent botnet detection," in Proc. USENIX Security, 2008, pp. 139–154.

[5]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.

[6]S. Nagaraja, P. Mittal, C.-Y. Hong, M. Caesar, and N. Borisov, "BotGrep: Finding P2P bots with structured graph analysis," in Proc. USENIX Security, 2010, pp. 1–16.

[7] J. Zhang, X. Luo, R. Perdisci, G. Gu, W. Lee, and N. Feamster, "Boosting the scalability of botnet detection using adaptive traffic sampling," in Proc. 6th ACM Symp. Inf., Comput. Commun. Security, 2011, pp. 124–134.

[8]M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "On clustering validation techniques," J. Intell. Inf. Syst., vol. 17, nos. 2–3, pp. 107–145, 2001.

[9]S. Saad, I. Traore, A. Ghorbani, B. Sayed, D. Zhao, W. Lu, et al., "Detecting P2P botnets through network behavior analysis and machine learning," in Proc. 9th Annu. Int. Conf. PST, Jul. 2011, pp. 174–180.

[10]D. Liu, Y. Li, Y. Hu, and Z. Liang, "A P2P-botnet detection model and algorithms based on network streams analysis," in Proc. IEEE FITME, Oct. 2010, pp. 55–58.

[11]W. Liao and C. Chang, "Peer to peer botnet detection using data mining scheme," in Proc. IEEE Int. Conf. ITA, Aug. 2010, pp. 1–4.

[12]T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "BLINC: Multilevel traffic classification in the dark," in Proc. ACM SIGCOMM, 2005, pp. 229–240.

[13] S. Sen, O. Spatscheck, and D. Wang, "Accurate, scalable in-network identification of P2P traffic using application signatures," in Proc. 13th ACM Int. Conf. WWW, 2004, pp. 512–521.

[14] T. Karagiannis, A. Broido, M. Faloutsos, and K. Claffy, "Transport layer identification of P2P traffic," in Proc. 4th ACM SIGCOMM Conf. IMC, 2004, pp. 121–134.

[15]A. W. Moore and D. Zuev, "Internet traffic classification using Bayesian analysis techniques," in Proc. ACM SIGMETRICS, 2005, pp. 50–60.

[16]M. P. Collins and M. K. Reiter, "Finding peer-to-peer file sharing using coarse network behaviors," in Proc. 11th ESORICS, 2006, pp. 1–17.