Botnet Detection Based on Traffic Monitoring

Hossein Rouhani Zeidanloo, Azizah Bt Manaf
Centre for Advanced Software Engineering
University of Technology Malaysia
54100 Kuala Lumpur, Malaysia
h_rouhani@hotmail.com, Azizah07@ ic.utm.my

Payam Vahdani, Farzaneh Tabatabaei, Mazdak Zamani
Centre for Advanced Software Engineering
University of Technology Malaysia
54100 Kuala Lumpur, Malaysia
vahdani@gmail.com

***Abstract*— Botnet is most widespread and occurs commonly in today's cyber attacks, resulting in serious threats to our network assets and organization's properties. Botnets are collections of compromised computers (Bots) which are remotely controlled by its originator (BotMaster) under a common Commond-and-Control (C&C) infrastructure. They are used to distribute commands to the Bots for malicious activities such as distributed denial-of-service (DDoS) attacks, spam and phishing. Most of the existing Botnet detection approaches concentrate only on particular Botnet command and control (C&C) protocols (e.g., IRC,HTTP) and structures (e.g., centralized), and can become ineffective as Botnets change their structure and C&C techniques.**

In this paper, we proposed a new general detection framework. This proposed framework is based on finding similar communication patterns and behaviors among the group of hosts that are performing at least one malicious activity. The point that distinguishes our proposed detection framework from many other similar works is that there is no need for prior knowledge of Botnets such as Botnet signature.

Keywords- Botnet; bot; P2P; detection; malicious activity

I. INTRODUCTION

Nowadays, the most serious manifestation of advanced malware is Botnet. To make distinction between Botnet and other kinds of malware, we have to comprehend the concept of Botnet. For a better understanding of Botnet, we have to know two terms first, Bot and BotMaster and then we can properly define Botnet. *Bot* – Bot is actually short for robot which is also called as Zombie. It is a new type of malware [1] installed into a compromised computer which can be controlled remotely by BotMaster for executing some orders through the received commands. After the Bot code has been installed into the compromised computers, the computer becomes a Bot or Zombie [2]. Contrary to existing malware such as virus and worm which their main activities focus on attacking the infecting host, bots can receive commands from BotMaster and are used in distributed attack platform.

BotMaster – BotMaster is also known as BotHerder, is a person or a group of person which control remote Bots. *Botnets*- Botnets are networks consisting of large number of Bots. Botnets are created by the BotMaster to setup a private communication infrastructure which can be used for malicious activities such as Distributed Denial-of-Service (DDoS), sending large amount of SPAM or phishing mails, and other nefarious purpose [3, 4, 5,6].

The main difference between Botnet and other kind of malwares is the existence of Command-and-Control (C&C) infrastructure. The C&C allows Bots to receive commands and malicious capabilities, as devoted by BotMaster. The first generation of Botnets utilized the IRC (Internet Relay Chat) channels as their Common-and-Control (C&C) centers. The centralized C&C mechanism of such Botnet has made them vulnerable to being detected and disabled. Therefore, new generation of Botnet which can hide their C&C communication have emerged, Peer-to-Peer (P2P) based Botnets. The P2P Botnets do not suffer from a single point of failure, because they do not have centralized C&C servers [6, 7]. Recently researches have proposed some approaches and techniques [8,9,10,11,12,13] for detecting Botnets. Majority of these approaches are developed for detecting IRC or HTTP based Botnets[8,9,13]. For instance, BotSniffer[11] is designed especially for detecting IRC and HTTP based Botnets. Rishi[9] is also designed for detecting IRC based Botnets with using well-known IRC bot nickname patterns as signature. But recently we have witnessed that structure of Botnets moved from centralized to distributed (e.g., using P2P [14,15]). Consequently, the detection approaches designed for IRC or HTTP based Botnets may become ineffective against the new P2P based Botnets. Therefore, we need to develop a next generation Botnet detection system, which is effective in the face of P2P based Botnets as well

We proposed a new general framework for detection of Botnets that currently targets P2P based Botnets, however the framework has the capability of adding another component for centralized Botnets. This framework at one stage monitors the group of hosts that perform at least one malicious activity and then try to find the hosts that show similar communication and behavior patterns.

The rest of the paper is organized as follows. In Section 2, we review the related work. In section 3, we describe our proposed detection framework and all its components and finally conclude in section 4.

II. RELATED WORK

There are many techniques for detection of Botnets. However, there are two essential techniques for Botnet detection: setting up honeynets and passive network traffic monitoring [16]. Many papers discussed about using honeynets for Botnet detection [5,3,17,21]. But we have to take into consideration that honeynets cannot detect bot infection most of the times and is just good for understanding Botnet characteristics. For identifying the existence of Botnets in the network, passive network traffic monitoring is helpful. This technique can be classified into signature-based, anomaly-based, DNS-based, and mining-based. Signature-based detection techniques can just be used for detection of recognized Botnets. Therefore, this solution is not functional for unknown bots. Anomaly-based detection techniques attempt to detect Botnets based on several network traffic anomalies such as high network latency, high volumes of traffic, traffic on unusual ports, and unusual system behavior that could indicate presence of malicious bots in the network [20]. DNS-based detection techniques are based on DNS information generated by a Botnet. As mentioned before, bots normally begin connection with C&C server to get commands. In order to access the C&C server bots carry out DNS queries to locate the particular C&C server that is typically hosted by a DDNS(Dynamic DNS) provider. Therefore, it is feasible to detect Botnet DNS traffic by DNS monitoring and detect DNS traffic anomalies [21, 22].

Data mining techniques are also can be used to detect Botnets. Geobl and Holz [9] proposed Rishi in 2007. Rishi is based on traffic monitoring for IRC servers, suspicious IRC nicknames and uncommon server ports. They use n-gram analysis and a scoring system to detect bots that use unusual communication channels, which are commonly not detected by standard intrusion detection systems [9]. This technique cannot detect non-IRC Botnets as well as encrypted communication. Masud *et al.* [23] proposed efficient flow-based Botnet traffic detection by mining multiple log files. They used several log correlation for C&C traffic detection. This technique is applicable to non-IRC Botnets. Because this method does not require access to payload content, it is applicable even if C&C payload is not available or encrypted[23].

III. Proposed Botnet Detection Framework and Components

Our proposed framework is based on passively monitoring network traffics. Figure 1 shows the architecture of our proposed Botnet detection system, which consist of 4 main components: Filtering, Application Classifier, Traffic Monitoring, Malicious Activity Detector. Filtering is responsible to filter out irrelevant traffic flows. The main benefit of this stage is reducing the traffic workload and makes application classifier process more efficient. Application classifier is responsible for separating IRC and HTTP traffics from the rest of traffics. Malicious activity detector is responsible to analyze the traffics carefully and try to detect malicious activities that internal host may perform and separate those hosts and send to next stage. Traffic Monitoring is responsible to detect the group of hosts that have similar behavior and communication pattern by inspecting network traffics.

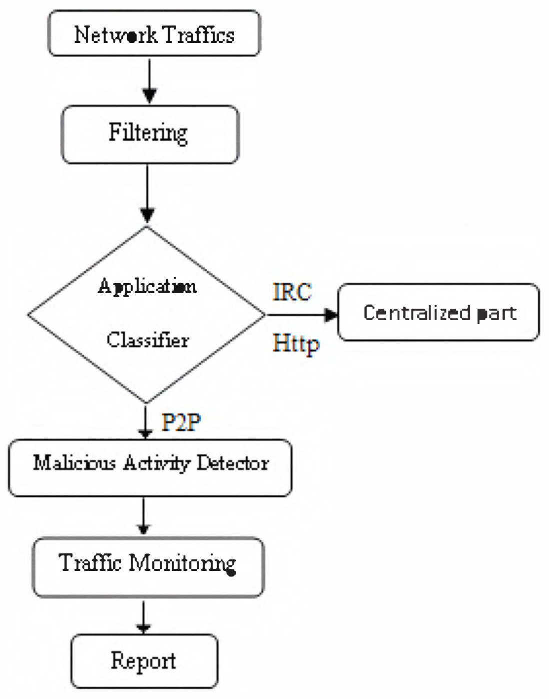

Figure 1: Architecture overview of our proposed detection framework

A. Filtering

Filtering is responsible to filter out irrelevant traffic flows. The main objective of this part is for reducing the traffic workload and makes the rest of the system perform more efficiently. Figure 2 shows the architecture of our filtering system In C1, we filter out those traffics which targets (destination IP address) are recognized servers and will unlikely host Botnet C&C servers. In C2, we filter out traffics that are established from external host towards internal hosts. In C3, we filter out handshaking processes (connection establishments) that are not completely established. Handshaking is an automated process of negotiation that dynamically sets parameters of a communications channel established between two entities before normal communication over the channel begins. It follows the physical establishment of the channel and precedes normal information transfer [24].

Figure 2: Traffics filtering stages

B. Application Classifier

Application Classifier is responsible to separate IRC and HTTP traffics from the rest of traffics and send them to

Centralized part. For detecting IRC traffics we can inspect the contents of each packet and try to match the data against a set of user defined strings. For this purpose we use payload inspection that only inspects the first few bytes of the payload and looking for specific strings. These IRC specific strings are NICK for the client's nickname, PASS for a password, USER for the username, JOIN for joining a channel, OPER that says a regular user wants to become a channel operator and PRIVMSG that says the message is a private message.[25] Using this strategy for detecting IRC traffic is almost simple for most network intrusion detection software like Snort.[19]. In some cases botmasters are using encryption for securing their communication that make using packet content analysis strategy useless. This issue actually is not our target here. In next step, we also have to separate Http traffics and send to Centralized part. For this purpose we also can inspect the first few bytes of Http request and if it has certain patterns or strings, separate it and send it to centralized part. For detecting Http traffics we focus on concept of Http protocol. Like most network protocols, HTTP uses the client-server model: An HTTP client opens a connection and sends a *request message* to an HTTP server (e.g. "Get me the file 'home.html'"); the server then returns a *response message*, usually containing the resource that was requested("Here's the file", followed by the file itself). After delivering the response, the server closes the connection (making HTTP a *stateless* protocol, i.e. not maintaining any connection information between transactions)[26]. In the format of Http request message, we are focusing on Http methods. Three common Http methods are "GET", "HEAD", or "POST" [26].

Therefore we inspect the traffics and if the first few bytes of an Http request contain "GET", "POST" or "HEAP", it's the indication of Http protocol and will separate those flows and send them to Centralized part. After filtering out Http and IRC traffics, the remaining traffics that have the probability of containing P2P traffics are send to Malicious Activity Detector. However in parallel we can use other approaches for identifying P2P traffics. We have to take into consideration that P2P traffic is one of the most challenging application types. Identifying P2P traffic is difficult both because of the large number of proprietary p2p protocols, and also due to the deliberate use of random port number for communication.

Payload-based classification approaches customized to p2p traffic have been presented in [29, 28], while identification of p2p traffic through transport layer characteristics is proposed in [27]. Our suggestion for using specific application or tools for identifying P2P traffics other than sending remaining traffics is use of BLINC [30] that can identify general P2P traffics. In contrast to previous methods, BLINC is based on observing and identifying patterns of host behavior at the transport layer. BLINC investigates these patterns at three levels of increasing detail (i) the social, (ii) the functional and (iii) the application level. This approach has two important features. First, it operates in the dark, having (a) no access to packet payload, (b) no knowledge of port numbers and (c) no additional information other than what current flow collectors provide.[30]

C. Malicious Activity Detector

In this part we have to analyze the outbound traffic from the network and try to detect the hosts that are performing at least one malicious activity. Each host may perform different kind of malicious activity but Scanning, Spamming, Binary downloading and exploit attempts are the most common and efficient malicious activities a botmaster may command their bots to perform [33, 17, 34]. In this paper we just focus on scanning and spam-related activities. The outputs of this part are the list of hosts which performed malicious activities.

1) Scanning: Scanning activities may be used for malware propagation and DOS attacks. There has been little work on the problem of detecting scan activities. Most scan detection has been based on detecting N events within a time interval of T seconds. This approach has the problem that once the window size is known, the attackers can easily evade detection by increasing their scanning interval. Snort[19] are also use this approaches. Snort version 2.0.2 uses two preprocessors. The first is packet-oriented, focusing on detecting malformed packets used for "stealth scanning" by tools such as nmap [35]. The second is connection oriented. It checks whether a given source IP address touched more than X number of ports or Y number of IP addresses within Z seconds. Snort's parameters are tunable, but it suffers from the same drawbacks as Network Security Monitor(NSM)[36] since both rely on the same metrics [37]. After assessing different approaches for detecting scanning activities, the best solution for using in this part is Statistical sCan Anomaly Detection Engine(SCADE)[10], a snort processor plug-in system which has two modules, one for inbound scan detection and another one for detecting outbound attack propagation.

2) spam-related activities: E-mail spam, known as Unsolicited Bulk Email (UBE), junk mail, is the practice of sending unwanted email messages, in large quantities to an indiscriminate set of recipients. More than 95% of email on the internet is spam, which most of these spams are sent from Botnets. A number of famous Botnets which have been used specially for sending spam is Storm Worm [14] which is P2P Botnet. Our target here is not recognizing which email message is spam, though for detecting group of bots that sending spam with detecting similarities among their actions and behaviors. Therefore the content of emails from internal network to external network is not important in our solution. All we want to do is determining which clients have been infected by bot and are sending spam. For reaching to this target, we are focusing on the number of emails sending by clients to different mail servers. Based on our experience in our lab, using different external mail servers for many times by same client is an indication of possible malicious activities. It means that it is unusual that a client in our network send many emails to the same mail server (SMTP server) in the period of time like one day. Therefore, we are inspecting outgoing traffic from our network(gateway), and recording SIP and DIP of those traffics that destination ports are 25(SMTP) or 587(Submission) in the database. Based on network flows between internal hosts and external computers(SIP belong to mail servers) and the number of times that it can happen we can conclude which internal host is behaving unusual and are sending many emails to mail servers.

D. Traffic Monitoring

Traffic Monitoring is responsible to detect the group of hosts that have similar behavior and communication pattern by inspecting network traffics. Therefore we are capturing network flows and record some special information on each flow. We are using Audit Record Generation and Utilization System (ARGUS) which is an open source tool [31] for monitoring flows and record information that we need in this part. Each flow record has following information: Source IP(SIP) address, Destination IP(DIP) address, Source Port(SPORT), Destination Port(DPORT), Duration, Protocol, Number of packets(*np*) and Number of bytes(*nb*) transferred in both directions. Then we insert this information on a data base like Figure 3, which $\{fi\}i = 1$ …n are network flows. After this stage we specify the period of time which is 6 hours and during each 6 hours, all n flows that have same Source IP, Destination IP, Destination port and same protocol (TCP or UDP) are marked and then for each network flow we calculate Average number of bytes per second and Average number of bytes per packet:

- Average number of bytes per second(*nbps*) = Number of bytes/ Duration
- Average number of bytes per packet(*nbpp*) = Number of Bytes/ Number of Packets

f1	Sip	Dip	S. port	D. port	Protocol	*np*	*nb*	duration
f2								
f3								
.								
fn								

Figure 3: Recorded information of network flows using ARGUS

Then, we insert this two new values (*nbps* and *nbpp*) including SIP and DIP of the flows that have been marked into another database, similar to figure 4 . Therefore, during the specified period of time (6 hours), we might have a set of database, $\{di\}i=1\ldots m$ which each of these databases have same SIP, DIP, DPORT and protocol (TCP/UDP). We are focusing just at TCP and UDP protocols in this part.

Source Port	Destination Port	*nbps*	*nbpp*

Figure 4: Database for analogous flows

As we mentioned earlier, the bots belonging to the same Botnet have same characteristics. They have similar behavior and communication pattern, especially when they want to update their commands from botmasters or aim to attack a target; their similar behaviors are more obvious. Therefore, next step is looking for group of databases that are similar to each other. For finding similar communication flows among databases $\{di\}i = 1$.., one soloution is using clustering algorithm like X-means clustering algorithm [32]. X-means is one of the most famous clustering algorithms.

We proposed a simple solution for finding similarities among group of databases. For each database we can draw a graph in x-y axis, which x-axis is the Average Number of Bytes per Packet (*nbpp*) and y-axis is Average Number of Byte Per Second (*nbps*). (X, Y)= (bpp, bps)

For example, in database (d_i), for each row we have *nbpp* that specify x-coordinate and have *nbps* that determine y-coordinate. Both x-coordinate and y-coordinate determine a point (x,y) on the x-y axis graph. We do this procedure for all rows (network flows) of each database. At the end for each database we have number of points in the graph that by connecting those points to each other we have a curvy graph.

Next step is comparing different x-y axis graphs, and during that period of time (each 6 hours) those graphs that are similar to each other are clustered in same category. The results will be some x-y axis graphs that are similar to each other. Each of these graphs is referring to their corresponding databases in previous step. We have to take record of SIP addresses of those hosts and report them as possible bots in the network.

IV. Conclusion

In comparison to other kind of malware Botnets are harder to monitor and shutdown and detection of them become a challenging problem. In this paper we proposed a new general detection framework. In our proposed detection framework, we monitor the group of hosts that perform at least one malicious activity in one step and then try to find the hosts that show similar communication patterns among them. The point that distinguishes our proposed detection framework from many other similar works is that there is no need for prior knowledge of Botnets such as Botnet

signature. In addition, we plan to further improve the efficiency of our proposed detection framework with adding unique detection method in centralized part and make it as one general system for detection of Botnet and try to implement it in near future.

REFERENCES

[1] P. Barford and V.Yagneswaran, "An Inside Look at Botnets". In: Special Workshop on Malware Detection, Advances in Information Security, Springer, Heidelberg (2006).

[2] N. Ianelli, A. Hackworth, Botnets as a Vehicle for Online Crime, CERT, December 2005.

[3] E. Cooke, F. Jahanian, and D. McPherson, "The zombie roundup: Understandinng, detecting, and disrupting Botnets," Proc. of Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI'05), June 2005.

[4] Honeynet project, Know your Enemy:tracking Botnets, march 2005. http://www.honeynet.org/papers/bots

[5] M.A Rajab, J. Zarfoss, F. Monrose, and A. Terzis, "A multifaceted approach to understanding the Botnet phenomenon," *6th ACM* SIGCOMM on Internet Measurement Conference, IMC 2006, 2006, pp. 41-52.

[6] Zeidanloo, H.R; Manaf, A.A. "Botnet Command and Control Mechanisms". Second International Conference on Computer and Electrical Engineering,2009. ICCEE. Page(s):564-568.

[7] Duc T. Ha, Guanhua Yan, Stephan Eidenbenz, Hung Q. Ngo. "On the effectiveness of structural detection and defense against P2P-based Botnet," Proc. of the 39th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'09), june2009

[8] J. R. Binkley and S. Singh. An algorithm for anomaly-based Botnet detection. In *Proceedings of USENIX SRUTI'06*, pages 43–48, July 2006

[9] J. Goebel and T. Holz. Rishi: identify bot contaminated hosts by irc nickname evaluation in proceeding of USENIX security symposium (security 2007)

[10] G. Gu, P. Porras, V. Yegneswaran, M. Fong, and W. Lee. BotHunter:Detecting malware infection through ids-driven dialog correlation. In proceedings of the 16th USEBIX security symposium(security 2007).

[11] G. Gu, J. Zhang, and W. Lee. BotSniffer: Detecting Botnet command and control channels in network traffics. In proceeding of 15th Annual network and Distributed system security symposioum,2008.

[12] A. Karasaridis, B. Rexroad, and D. Hoeflin. Widescale Botnet detection and haracterization. In *Proceedings of USENIX* HotBots'07, 2007.

[13] W. T. Strayer, R.Walsh, C. Livadas, andD. Lapsley. Detecting Botnet with tight command and control. In Proceeding of the 31th IEEE conference on local computernetwork,2006.

[14] J. B. Grizzard, V. Sharma, C. Nunnery, B. B. Kang, and D. Dagon. Peer-to-peer Botnets: Overview and case study. In *Proceedings of USENIX HotBots'07*, 2007.

[15] R. Lemos. Bot software looks to improve peerage. Http://www.securityfocus.com/news/11390, 2006.

[16] Z. Zhu, G. Lu, Y. Chen, Z. J. Fu, P.Roberts, K. Han, "Botnet Research survey". in *Proc. 32nd Annual IEEE International conferences on computer software and applications(COMPSAC)2008.page 967-972*

[17] A. Ramachandran and N. Feamster, "Understanding the network-level behavior of spammers," in *Proc. ACM SIGCOMM*, 2006.

[18] K. K. R. Choo, "Zombies and Botnets," Trends and issues in crime and criminal justice, no. 333, Australian Institute of Criminology, Canberra, March 2007.

[19] Snort IDS web page. http://www.snort.org, March 2006.

[20] B. Saha and A, Gairola, "Botnet: An overview," *CERT-In White* PaperCIWP-2005-05, 2005.

[21] H. Choi, H. Lee, H. Lee, and H. Kim, "Botnet Detection by Monitoring group avtivities in DNS Traffic," in *Proc. 7th IEEE international conference on computer and information technology(CIT 2007).page 715-720.*

[22] R.Villamarin-Salomon and J.C. Brustoloni, "Identifying Botnets using Anomaly Detection Techniques Applied to DNS Traffic," in proceeding *5th IEEE Consumer Communications and Networking confernce. (CCNC 2008)*, 2008, pp. 476-481.

[23] M. M. Masud, T. Al-khateeb, L. Khan, B. Thuraisingham, K. W. Hamlen, " Flow-based identification of Botnet traffic by mining multiplelog file," in Proc. International Conference on Distributed Frameworks & Applications (DFMA), Penang, Malaysia, 2008

[24] "handshaking". http://en.wikipedia.org/wiki/Handshaking

[25] J. Rayome. "IRC on Your Dime? What You Really Need to Know about Internet Relay Chat". CIAC/LLNL. May 22, 1998

[26] HTTP Made Really Easy, http://www.jmarshall.com/easy/http/.

[27] T. Karagiannis, A.Broido, M. Faloutsos, and kc claffy. Transport layer identification of P2P traffic. In ACM/SIGCOMM IMC, 2004.

[28] T. Karagiannis, A.Broido, N.Brownlee, kc claffy, and M.Faloutsos Is P2P dying or just hiding? In IEEE Globecom 2004, GI.

[29] S. Sen, O. Spatscheck, and D. Wang. Accurate, Scalable In-Network Identification of P2P Traffic Using Application Signatures. In WWW, 2004

[30] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "BLINC:multilevel traffic classification in the dark," In Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer *Communications*, pp. 229-240, Philadelphia, Pennsylvania, 2005

[31] Argus (Audit Record Generation and Utilization System, http://www.qosient.com/argus)

[32] PELLEG, D. and MOORE, A. W., "X-means: Extending k-means with efficient estimation of the number of clusters," in *Proceedings* of the Seventeenth International Conference on Machine Learning *(ICML'00)*, (San Francisco, CA, USA), pp. 727– 734, Morgan Kaufmann Publishers Inc., 2000.

[33] COLLINS, M., SHIMEALL, T., FABER, S., JANIES, J., WEAVER, R., SHON, M. D., and KADANE, J., "Using uncleanliness to predict future Botnet addresses,," in Proceedingsof ACM/USENIX Internet Measurement Conference (IMC'07), 2007

[34] ZHUGE, J., HOLZ, T., HAN, X., GUO, J., and ZOU, W., "Characterizing the ircbased Botnet phenomenon." Peking University& University ofMannheim Technical Report, 2007.

[35] Nmap — free security scanner for network exploration & security audits. http://www.insecure.org/nmap/.

[36] L. T. Heberlein, G. V. Dias, K. N. Levitt, B. Mukherjee, J.Wood, and D.Wolber. A network security monitor. In *Proc. IEEE* Symposium on Research in Security and Privacy, pages 296–304, 1990.

[37] JUNG, J., PAXSON, V., BERGER, A. W., and BALAKRISHNAN, H., "Fast Portscan Detection Using Sequential Hypothesis Testing," in *IEEE Symposium on Security and Privacy 2004*, (Oakland, CA), May 2004.