IACSIT International Journal of Engineering and Technology, Vol.2, No.2, April 2010
ISSN: 1793-8236

A Proposed Framework for P2P Botnet Detection

Hossein Rouhani Zeidanloo, Azizah Bt Abdul Manaf, Rabiah Bt Ahmad, Mazdak Zamani,

Saman Shojae Chaeikar

***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 Command-and-Control (C&C) infrastructure. They are used to distribute commands to 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 detection framework which focuses on P2P based botnets. This proposed framework is based on our definition of botnets. We define a botnet as a group of bots that will perform similar communication and malicious activity patterns within the same botnet. In our proposed detection framework, we monitor the group of hosts that show similar communication pattern in one stage and also performing malicious activities in another step, and finding common hosts on them.**

***Index Terms*— botnet; bot; centralized; decentralized; P2P; detection**

I. Introduction

Nowadays, the most serious manifestation of advanced malware is Botnet. 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 [1,2,3,4,5] .

There are many ways which attackers use to infect a computer in the Internet with Bot include sending email and using malicious websites, but the most common way is searching the Internet to look for vulnerable and unprotected computers [6.].

The main difference between Botnet and other kind of malwares is the existence of Command-and-Control (C&C) infrastructure. 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 [12]. Attackers have accordingly developed a range of strategies and techniques to protect their C&C infrastructure. The rest of the paper is organized as follows. In Section 2, we analyze different botnet topologies and completely consider the protocols that are currently being used in each model. In Section 3, we review the related work. In Section 4, we describe our proposed detection framework and all its components and finally conclude in section 5.

Manuscript received March 9, 2010.

Hossein Rouhani Zeidanloo, Rabiah Bt Ahmad and Saman Shojae Chaeikar, centre for Advanced Software Engineering, Universiti Teknologi Malaysia, Kuala Lumpur, Malaysia (email:h_rouhani@hotmail.com, rabiah@citycampus.utm.my, saman_shoja@yahoo.com).

Azizah Bt Abdul Manaf, college of Science and Technology, Universiti Teknologi Malaysia, Kuala Lumpur, Malaysia (email: azizah07@citycampus.utm.my).

Mazdak Zamani, faculty of Computer Science and Information System, Universiti Teknologi Malaysia, Kuala Lumpur, Malaysia (email: zmazdak2@siswa.utm.my).

II. Botnet Topologies

According to the Command-and-Control(C&C) channel, we categorized Botnet topologies into two different models, the Centralized model and the Decentralized model.

A. *Centralized model*

The oldest type of topology is the centralized model. In this model, one central point is responsible for exchanging commands and data between the BotMaster and Bots. Many well-known Bots, such as AgoBot, SDBot, Zotob and RBot used this model. In this model, BotMaster chooses a host (usually high bandwidth computer) to be the central point (Command-and-Control) server of all the Bots. The C&C server runs certain network services such as IRC or HTTP. The main advantage of this model is small message latency which cause BotMaster easily arranges Botnet and launch attacks. Since all connections happen through the C&C server, therefore, the C&C is a critical point in this model. In other words, C&C server is the weak point in this model. If somebody manages to discover and eliminates the C&C server, the entire Botnet will be worthless and ineffective. Thus, it becomes the main drawback of this model.

Since IRC and HTTP are two common protocols that C&C server uses for communication, we consider Botnets in this model based on IRC and HTTP. Figure 1 shows the basic communication architecture for a Centralized model.

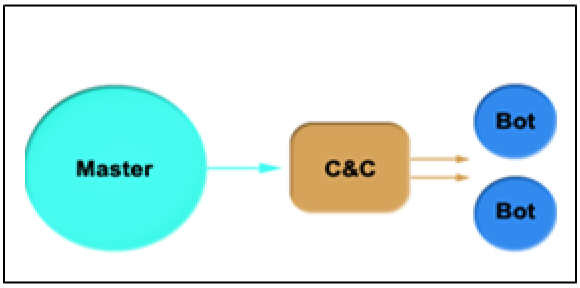

Figure 1: Command and control architecture of a Centralized model

1) Botnet based on IRC: The IRC is a form of real-time Internet text messaging or synchronous conferencing [13]. The protocol is based on the Client-Server model, which can be used on many computers in distributed networks. Some advantages which made IRC protocol widely being used in remote communication for Botnets are: (1) Low latency communication; (2) Anonymous real-time communication; (3) Ability of Group (many-to-many) and Private (one-to-one) communication; (4) simple to setup and (5) simple commands. The basic commands are connect to servers, join channels and post messages in the channels; (6) Very flexibility in communication. Therefore IRC protocol is still the most popular protocol being used in Botnet communication [5].

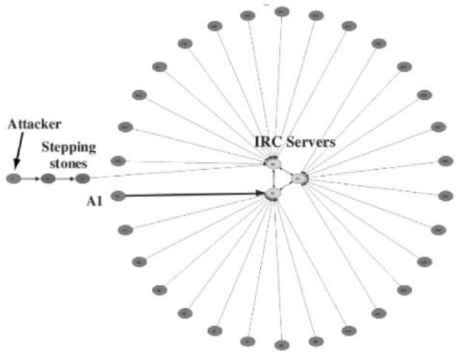

Figure 2: IRC based Botnet (source: Dave Dittrich *et al*. [14])

In this model, BotMasters can command their Bots as a whole or command a few of the Bots selectively using one-to-one communication. The C&C server runs IRC service that is the same with other standard IRC service. BotMaster usually creates a designated channel on the C&C servers where all the Bots will connect, awaiting commands in the channel which will instruct each connected Bot to do the BotMaster's command. Figure 2 showed that there is one central IRC server that forwards commands and data between the BotMaster and his Bots.

2) Botnet based on HTTP: The HTTP protocol is another popular protocol used by Botnets. Since IRC protocol within Botnets became well-known, more internet security researchers gave attention to monitoring IRC traffic to detect Botnet. Consequently, attackers started to use HTTP protocol as a Command-and-Control communication channel to make Botnets become more difficult to detect. The main advantage of using the HTTP protocol is hiding Botnets traffics in normal web traffics, so it can easily bypasses firewalls with port-based filtering mechanisms and avoid IDS detection. There are some known Bots using the HTTP protocol, such as Bobax [16], ClickBot [13] and Rustock [17]. Gu et al in the reference [10] pointed out that the HTTP protocol is in a "pull" style and the IRC is in a "push" style.

B. *Decentralized Model*

Due to major disadvantage of Centralized model – Central Command-and-Control(C&C) – attackers started to build alternative Botnet communication system that is much harder to discover and to destroy. Hence, they decided to find a model in which the communication system does not heavily depending on few selected servers and even discovering and destroying a number of Bots. As a result, attackers exploit the idea of Peer-to-Peer (P2P) communication as a Command-and-Control(C&C) pattern which is more resilient to failure in the network. The P2P based C&C model will be used dramatically in Botnets in the near future, and definitely Botnets that use P2P based C&C model impose much bigger challenge for defense of networks. Since P2P based communication is more robust than Centralized C&C communication, more Botnets will move to use P2P protocol for their communication.

In P2P model, as shown in Figure 3, there is no Centralized point for communication. Each Bot keeps some connections to the other Bots of the Botnet. Bots act as both Clients and servers. A new Bot must know some addresses of the Botnet to connect there. If Bots in the Botnet are taken offline, the Botnet can still continue to operate under the control of BotMaster. P2P Botnets aim at removing or hiding the central point of failure which is the main weakness and vulnerability of Centralized model.

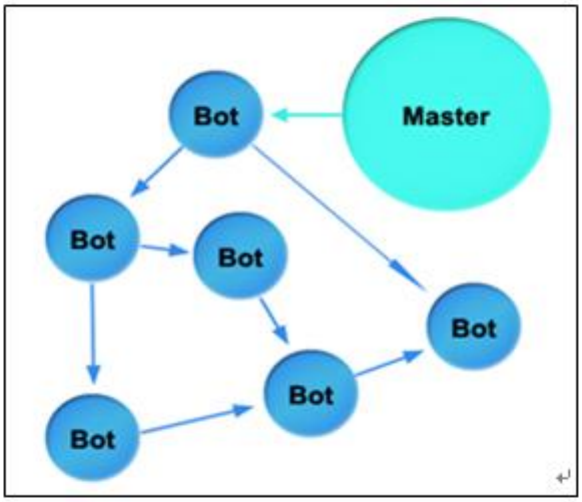

Figure 3: Example of Peer-to-peer Botnet Architecture

III. Related Work

The detection of botnet has been a major research topic in recent years. Different solutions and approaches have been proposed for detection and tracking of botnet. There are mainly two approaches for botnt detection. One approach is based on setting up honeynets. And another approach is monitoring and analysis of passive network traffic [20].

There are many papers [5,3, 21, 22, 23, 24, 1, 25, 26] discussed how to use honeynets for botnet detection and tracking. Honeynets are useful to understand botnet characteristics and technology, but cannot detect bot infection all the times.

The passive network traffic monitoring and analysis approach is useful to identify the existence of botnet in the networks. This technique can be classified as being signature-based, anomaly-based, DNS-based, and mining-based that will be described in this section respectively.

Signature-based botnet detection technique uses the signatures and behaviors of existing botnets for its detection. For example, Snort [27] is an open source intrusion detection system (IDS) that monitors network traffic to find signs of intrusion. Signature-based detection techniques can just be used for detection of known botnets. Therefore, this solution is not useful 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 [28]. However this technique meets the problem of detecting unknown botnets, but is unable to discover a IRC network botnet which has not been used yet for attacks. To solve this, Binkley and Singh [14] proposed an effective algorithm that combines TCP-based anomaly detection with IRC tokenization and IRC message statistics to create a system that can clearly detect client botnets. This algorithm can also reveal bot servers [14]. However, Binkley's approach could be easily crushed by simply using a minor cipher to encode the IRC commands.

Recently, Gu *et al.* have proposed Botsniffer [13] that uses network-based anomaly detection to identify botnet C&C channels in a local area network. Botsniffer is based on observation that bots within the same botnet will likely reveal very strong similarities in their responses and activities. Therefore, it employs several correlation analysis algorithms to detect spatial-temporal correlation in network traffic with a very low false positive rate [13].

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 [29, 30].

In 2005, Dagon [31] proposed a mechanism to identify botnet C&C servers by detecting domain names with abnormally high or temporally concentrated DDNS query rates. This technique is similar to the approach proposed by Kristoff [32] in 2004. However, both techniques have the weakness point by using faked DNS queries.

In 2007, Choi *et al.* [29] proposed an anomaly-based botnet detection mechanism by monitoring group activities in DNS traffic, which form a group activity in DNS queries that are sent by distributed bots. They have determined unique features of DNS traffic as group activity to differentiate botnet DNS queries from valid DNS queries. Since DNS traffic appears in several stages of botnet life-cycle, it is possible to detect botnet by using the group activity property of botnet DNS traffic. This anomaly-based botnet detection mechanism is more efficient than the previous approaches and can detect botnet despite the type of bot by looking at their group activities in DNS traffic.

Several data mining techniques including machine learning, classification, and clustering can be used efficiently to detect botnet C&C traffic. Geobl and Holz [15] proposed Rishi in 2007. Rishi is mainly based on passive traffic monitoring for unusual or suspicious IRC nicknames, IRC servers, and uncommon server ports. They use n-gram analysis and a scoring system to detect bots that use uncommon communication channels, which are commonly not detected by classical intrusion detection systems [15]. The disadvantages of this method are that it cannot detect encrypted communication as well as non-IRC botnets.

In 2008, Strayer *et al.* [33] proposed a network-based solution using machine learning techniques for detecting botnet traffic. They showed that evidence of botnet command and control activity can be extracted from flow characteristic using passive traffic analysis. They adopt a two stage process which first distinguish IRC flows, and then identify botnet C&C traffic from normal IRC flows [33]. Although these techniques are effective to detect some botnets, they are specific to IRC-based botnets. Moreover, for precise analysis and detection these techniques need access to payload content. Consequently, it cannot detect encrypted C&C traffic.

Masud *et al.* [34] proposed effective flow-based botnet traffic detection by mining multiple log files. They introduce multiple log correlation for C&C traffic detection. They classify an entire flow to identify botnet C&C traffic. This method does not impose any restriction on the botnet communication protocol and is therefore applicable to non-IRC botnets. Furthermore, this method does not require access to payload content. Hence, it is effective even if the C&C payload is encrypted or is not available [34].

Botminer [35] is the most recent approach which applies data mining techniques for botnet C&C traffic detection. Botminer is an improvement of Botsniffer [13]. It clusters similar communication traffic and similar malicious traffic. Then, it performs cross cluster correlation to identify the hosts that share both similar communication patterns and similar malicious activity patterns. Botminer is an advanced botnet detection tool which is independent of botnet protocol and structure. Botminer can detect real-world botnets including IRC-based, HTTP-based, and P2P botnets with a very low false positive rate [35].

As we mentioned above researches have proposed some approaches and techniques [14,15,16,13,17,18] for detecting botnets. Majority of these approaches are developed for detecting IRC or HTTP based botnets[14,15,18]. For instance, BotSniffer[13] is designed especially for detecting IRC and HTTP based botnets. Rishi[15] is also desingned 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 [9,19]). 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. In addition, we have to take into consideration that this detection system should require no prior knowledge of particular botnets (such as botnet signature, or C&C server names/addresses).

In order to come up with a new detection system that specifically meet the requirements for detection of P2P based botnets, we studied the communication and activity characteristics of few P2P based botnet(e.g. Storm Worm) and eventually come up with effective definition of botnets; specially for P2P based botnet:

"A group of bots (at least three) within the same botnet will perform similar communication and malicious activities".It means that if each bot within the same botnet show different behavior (e.g. in terms of receiving commands), the bots are nothing but just isolated and not related infected systems that we cannot consider them as a botnet according to our definition. According to definition above we proposed a new framework for detection of P2P botnets .

IV. Proposed Botnet Detection Framework And Components

Our proposed framework is based on passively monitoring network traffics. Consequently this model is not provided for detecting botnet at the very moment when hosts are infected with bots. This model is based on the definition of P2P botnets that multiple bots within the same botnet will perform similar communication patterns and malicious activities. Figure 4 shows the architecture of our proposed botnet detection system, which consist of 4 main components: Filtering, Traffic Monitoring, Malicious Activity Detector and Analyzer. 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. 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 patterns by inspecting network traffics. Analyzer is responsible for comparing the results of previous parts (Traffic Monitoring and Malicious Activity Detector) and finding hosts that are common on the results of both parts.

Figure 5: Traffics filtering stages

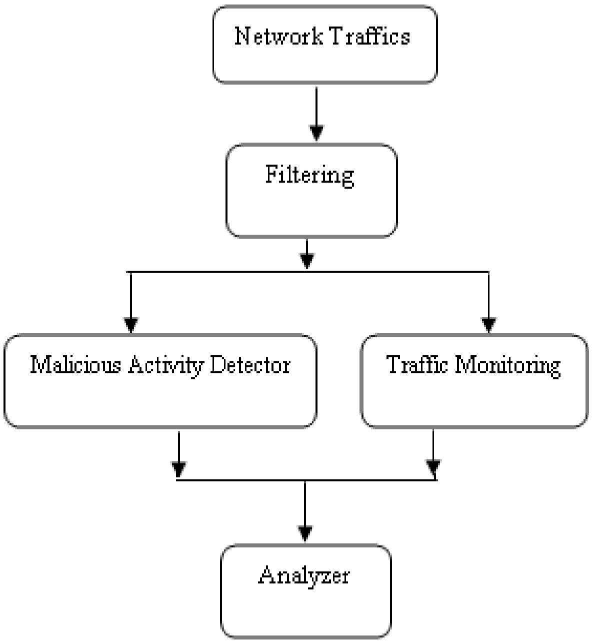

Figure 4: Architecture overview of our proposed detection framework

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 5 shows the architecture of the filtering.

In C1, we filter out those traffics which targets (destination IP address) are recognized servers and will unlikely host botnet C&C servers. For this purpose we used the top 500 websites on the web (http://www.alexa.com/topsites) , which the top 3 are google.com, facebook.com and yahoo.com. 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 [36]. A good example that usually we face with that in network is TCP protocol operations. To establish a connection, TCP uses a three-way handshake; in this case we filter out the traffics that TCP handshaking have not completed. Like a host sends SYN packets without completing the TCP handshake. Based on our experience these flows are mostly caused by scanning activities.

B. *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[43] 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.

fi	SIP	DIP	SPORT	DPORT	Protocol	*np*	*nb*	duration
$f1$								
$f2$								
.								
.								
.								
fn								

Figure 6: Recorded information of network flows using

Then we insert this information on a data base like Figure 2, which $\{fi\}i = 1 \dots n$ 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 for each network flow $\{fi\}$ (row) 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

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 3 . Therefore, during the specified period of time (6 hours), we might have a set of database, $\{di\}i = 1 \dots m$ 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 7: 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 to looking for groups of Databases that are similar to each other. For finding similar communication flows among databases $\{di\}i = 1 \dots m$, , one solution is using clustering algorithm like X-means clustering algorithm [44]. 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 (di), 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. We have an example, figure 8, for two different databases based on data in our lab that their graphs are almost similar to each other

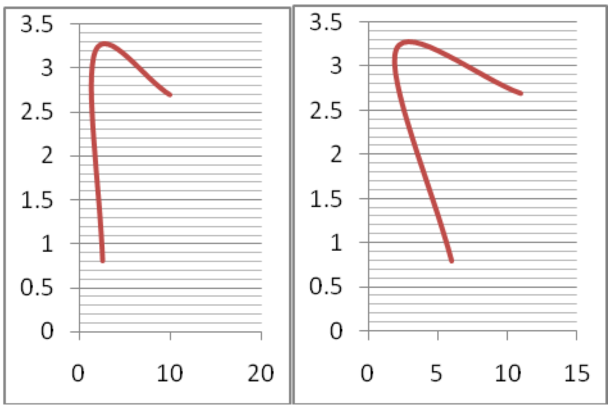

Figure 8: Example of two similar graphs based on data in our lab

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 send the list to next step for analyzing.

C. *Malicious Activity Detector*

In this part we have to analyze the outbound traffic from the network and try to detect the possible malicious activities that the internal machines are performing. 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 [45,26,46]. 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 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 [47]. 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)[48] since both rely on the same metrics [49]. Other works that are focusing on scan detection is by Staniford et al. on Stealthy Probing and Intrusion Correlation Engine (SPICE) [50]. SPICE is focusing on detecting stealthy scans, especially scans that spread across multiple source addresses and execute at very low rates. In SPICE there are anomaly scores for packets based on conditional probabilities derived from the SIP and DIP and ports. It uses simulated annealing to cluster packets together into port scan using heuristics that have developed from real scans [50]. An important need in our system is prompt response, however reaching to our goals which are promptness and accuracy in detecting malicious scanners is a difficult task. Another solution is also using Threshold Random Walk (TRW)[49], an online detection algorithm. TRW is based on sequential hypothesis testing.

After assessing different approaches for detecting scanning activities, the best solution for using in this part is Statistical sCan Anomaly Detection Engine(SCADE)[16], a snort processor plug-in system which has two modules, one for inbound scan detection and another one for detecting outbound attack propagation.

a) Inbound Scan Detection (ISD): In this part SCADE has focused on detection of scan activities based on ports that are usually used by malware. One of the good advantages of this procedure is that it is less vulnerable to DOS attacks, mainly because its memory trackers do not maintain per-external-source-IP. SCADE here just tracks scans that are targeted to internal hosts. The bases of Inbound Scan Detection are on failed connection attempts. SCADE in this part has defined two types of ports: High-Severity (hs) ports which representing highly vulnerable and commonly exploited services and low-severity (ls) ports. For make it more applicable in current situation SCADE focused on TCP and UDP ports as high-secure and all other as low-secure ports. There are different weights to a failed scan attempt for different types of ports.

The warning for ISD for a local host is produced based on an anomaly score that is calculated as based on this formula:

$$s = (w1Fhs + w2Fls)$$

Fhs: indicate numbers of failed attempts at high-severity ports.

Fls : shows numbers of failed attempts at low-severity ports.

b) Outbound Scan Detection (OSD): OSD is based on a voting scheme (AND, OR or MAJORITY). SCADE in this part has three parallel anomaly detection models that track all outbound connection per internal host:

• Outbound scan rate (s1): Detects local hosts that perform high-rate scans for many external addresses.

• Outbound connection failure rate (s2): Detects unusually high connection fail rates, with sensitivity to HS port usage. The anomaly score s2 is calculated based on this formula:

$$s2 = \frac{(w1Fhs + w2Fls)}{C}$$

Fhs : indicate numbers of failed attempts at high-severity ports.

Fls : shows numbers of failed attempts at low-severity ports.

C : is the total number of scans from the host within a time window.

• Normalized entropy of scan target distribution (s3): Calculates a Zipf (power-law) distribution of outbound address connection patterns. A consistently distributed scan target model provides an indication of a possible outbound scan. It is used an anomaly scoring technique based on normalized entropy to identify such candidates:

$$s3 = \frac{H}{\ln(m)}$$

H: is the entropy of scan target distribution which

$$H = -\sum_{i=1}^{m} pi \ln(pi)$$

m : is the total number of scan targets

pi : is the percentage of the scans at target i

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 [51], which most of these spams are sent from botnets. A number of famous botnets which have been used specially for sending spam are Storm Worm which is P2P botnet and Bobax that used Http as its C&C.

A common approach for detecting spam is the use of DNS Black/Black Hole List (DNSBL) such as (http://www.dnsbl.info/dnsbl-list.php). DNSBLs specify a list of spam senders’ IP addresses and SMTP servers are blocking the mail according to this list. This method is not efficient for bot-infected hosts, because legitimate IP addresses may be used for sending spam in our network. Creation or misuse of SMTP mail relays for spam is one of the most well-known exploitation of botnets. As we know user-level client mail application use SMTP for sending messages to mail server for relaying. However for receiving messages, client application usually use Post Office Protocol (POP) or the Internet Message Access Protocol (IMAP) to access the mail box on a mail server. Our idea in this part is very simple and efficient. 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 different or same mail servers.

D. Analyzer

Analyzer which is the last part of our proposed framework for detection of botnets, is responsible for finding common hosts that appeared in the results of previous parts(Traffic Monitoring and Malicious Activity Detector).

V. Conclusion

The first workshop on botnets was hold in 2007 and since then many detection approaches have been proposed and also some real bot detection systems have been implemented (e.g. BotHunterTM by Gu et al. [16]). Botnet detection is a challenging problem. In this paper we proposed a new P2P botnet detection framework. This proposed framework is based on our definition of botnets. We define a botnet as a group of bots that will perform similar communication and malicious activities pattern within the same botnet. In our proposed detection framework, we monitor the group of hosts that show similar communication pattern in one stage and also performing malicious activities in another step, and finding common hosts on 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 '09. Page(s): 564 – 568 .2009

[7] Duc T. Ha, Guanhua Yan, Stephan Eidenbenz, Hung Q. Ngo. On the Effectiveness of Structural Detection and Defense Against P2P-based Botnet, Proceedings of the 39th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'09), June 2009

[8] J. Oikerinen , D. Reed. Internet Relay Chat protocol .May 1993. Web publication. Available at URL : http://tools.ietf.org/html/rfc1459#section-1

[9] 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.

[10] J.Stewart.Bobax Trojan analysis. http://www.secureworks.com/research/threats/bobax/

[11] Daswani N, Stoppelman M, the Google Click Quality and Security Teams. The anatomy of ClickBot.A. In: Proc. of the 1st Workshop on Hot Topics in Understanding Botnets(HotBots 2007). 2007.

[12] Chiang K, Lloyd L. A case study of the rustock rootkit and spam Bot. In: Proc. of the 1st Workshop on Hot Topics in Understanding Botnets (HotBots 2007). 2007.

[13] Guofei Gu, Junjie Zhang, and Wenke Lee. "BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic." In Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS'08), San Diego, CA, February2008.

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

[15] J. Goebel and T. Holz. Rishi: Identify bot contaminated hosts by irc nickname evaluation. In Proceedings of USENIX HotBots'07, 2007.

[16] 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 USENIX Security Symposium (Security'07), 2007.

[17] A. Karasaridis, B. Rexroad, and D. Hoeflin. Widescale botnet detection and characterization. In Proceedings of USENIX HotBots'07, 2007.

[18] W. T. Strayer, R.Walsh, C. Livadas, andD. Lapsley. Detecting botnets with tight command and control. In Proceedings of the 31st IEEE Conference onLocal Computer Networks (LCN'06), 2006.

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

[20] Z. Zhu, G. Lu, Y. Chen, Z. J. Fu, P.Roberts, K. Han, "Botnet Research Survey," in Proc. 32nd Annual IEEE International Conference on Computer Software and Applications (COMPSAC '08), 2008, pp.967-972.

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

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

[23] M. Vrable, J. Ma, J. Chen, D. Moore, E.Vandekieft, A. C. Snoeren, G.M. Voelker, and S.Savage," Scalability, Fidelity and Containment in the Potemkin Virtual Honeyfarm," in Proc. ACM SIGOPS OperatingSystem Review, vol. 39(5), pp. 148–162, 2005.

[24] F. Freiling, T. Holz, and G. Wicherski, "Botnet tracking: Exploring a root-cause methodology to prevent distributed denial-of-service attacks," in Proc. 10th European Symposium on Research in Computer Security (ESORICS), vol. Lecture Notes in Computer Science 3676, September 2005, pp. 319–335

[25] D. Dagon, C. Zou, and W. Lee, "Modeling botnet propagation using time zones," in Proc. 13th Network and Distributed System Security Symposium (NDSS'06), 2006

[26] J. Oberheide, M. Karir, and Z.M. Mao, "Characterizing Dark DNS Behavior," in Proc. 4th nternational Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, 2007.

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

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

[29] H. Choi, H. Lee, H. Lee, and H. Kim, "Botnet Detection by Monitoring Group Activities in DNS Traffic," in Proc. 7th IEEE International Conference on Computer and Information Technology (CIT 2007), 2007,pp.715-720.

[30] R.Villamarin-Salomon and J.C. Brustoloni, "Identifying Botnets Using Anomaly Detection Techniques Applied to DNS Traffic," in Proc. 5th IEEE Consumer Communications and Networking Conference (CCNC2008), 2008, pp. 476-481.

[31] D. Dagon, "Botnet Detection and Response, The Network is the Infection," in OARC Workshop, 2005.

[32] J. Kristoff, "Botnets," in 32nd Meeting of the North American Network Operators Group, 2004.

IACSIT
International Association of Computer Science and Information Technology
WWW.IACSIT.ORG

[33] W. Strayer, D. Lapsley, B. Walsh, and C. Livadas, Botnet Detection Based on Network Behavior, ser. Advances in Information Security.Springer, 2008, PP. 1-24.
[34] M. M. Masud, T. Al-khateeb, L. Khan, B. Thuraisingham, K. W.Hamlen, " Flow-based identification of botnet traffic by mining multiple in Proc. International Conference on Distributed Framework& Application,Penang,Malaysia.2008
[35] G. Gu, R. Perdisci, J. Zhang, and W. Lee, "Botminer: Clustering analysis of network traffic for protocol- and structure independent botnet detection," in Proc. 17th USENIX Security Symposium, 2008
[36] http://en.wikipedia.org/wiki/Handshaking
[37] J. Rayome. " IRC on your dime? What you really need about Internet Relay Chat, CIAC/LLNL. May 22, 1998
[38] HTTP Made Really Easy, http://www.jmarshall.com/easy/http
[39] T. Karagiannis, A.Broido, M. Faloutsos, and kc claffy. Transport layer identification of P2P traffic. In ACM/SIGCOMM IMC, 2004.
[40] T. Karagiannis, A.Broido, N.Brownlee, kc claffy, and M.Faloutsos. Is P2P dying or just hiding? In IEEE Globecom 2004, GI.
[41] S. Sen, O. Spatscheck, and D. Wang. Accurate, Scalable In-Network Identification of P2P Traffic Using Application Signatures. In WWW, 2004
[42] 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
[43] Argus (Audit Record Generation and Utilization System, http://www.qosient.com/argus)
[44] 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.
[45] Collins, m., Shimeall, t., Faber, s., Janies, j., Weaver, R.Shon, M.D.,and Kadane,j. "using uncleanliness to predict future botnet addresses". In Proceedings ACM/USENIX Internet Measurement Conference.2007
[46] ZHUGE, J., HOLZ, T., HAN, X., GUO, J., and ZOU, W., "Characterizing the ircbased botnet phenomenon." Peking University& University ofMannheim Technical Report, 2007.
[47] Nmap — free security scanner for network exploration & security audits. http://www.insecure.org/nmap/.
[48] 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.
[49] 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
[50] S. Staniford, J. A. Hoagland, and J. M. McAlerney. Practical automated detection of stealthy portscans. In Proceedings of the 7th ACM Conference on Computer and Communications Security, Athens, Greece, 2000.
[51] WARD, M., "More than 95% of e-mail is 'junk'." http://news.bbc.co.uk/1/hi/technology/5219554.stm, 2006.