2009 Second International Conference on Computer and Electrical Engineering

Botnet Command and Control Mechanisms

Hossein Rouhani Zeidanloo
Faculty of Computer Science and Information System
UTM International Campus, UTM
Kuala Lumpur, Malaysia
E-mail: H_Rouhani@hotmail.com

Azizah Abdul Manaf
Faculty of Computer Science and Information System
UTM International Campus, UTM
Kuala Lumpur, Malaysia
E-mail: Azizah07@ ic.utm.my

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, sending large amount of SPAM and other nefarious purposes. Understanding the Botnet C&C channels is a critical component to precisely identify, detect, and mitigate the Botnets threats. Therefore, in this paper we provide a classification of Botnets C&C channels and evaluate well-known protocols (e.g. IRC, HTTP, and P2P) which are being used in each of them.**

Keywords- botnet; bot; centralized; decentralized; P2P

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]. Bots infect a person's computer in many ways.

Bots usually disseminate themselves across the Internet by looking for vulnerable and unprotected computers to infect. When they find an unprotected computer, they infect it and then send a report to the BotMaster. The Bot stay hidden until they are announced by their BotMaster to perform an attack or task. Other ways in which attackers use to infect a computer in the Internet with Bot include sending email and using malicious websites, but common way is searching the Internet to look for vulnerable and unprotected computers [6]. Based on our understanding, we could say that the activities associated with Botnet can be classified into three parts: (1) *Searching* – searching for vulnerable and unprotected computers. (2) *Dissemination* – the Bot code is distributed to the computers (targets), so the targets become Bots. (3) *sign-on* – the Bots connect to BotMaster and become ready to receive command and control traffic.

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. BotMaster must ensure that their C&C infrastructure is sufficiently robust to manage thousands of distributed Bots across the globe, as well as resisting any attempts to shutdown the Botnets. However, detection and mitigation techniques against Botnets have been increased [7, 8, 9, 10, 11]. Recently, attackers are also continually improving their approaches to protect their Botnets. 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.

Therefore, considering the C&C function gives better understanding of Botnet and help defenders to design proper detection or mitigation techniques. According to the C&C channel we categorize Botnets into three different topologies: a) Centralized; b) Decentralized and c) Hybrid. In Section 2, we analyzed these topologies and completely considered the protocols that are currently being used in each model.

II. Botnet Topologies

According to the Command-and-Control(C&C) channel, we categorized Botnet topologies into three different models, the Centralized model, the Decentralized model and Hybrid 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. A lot of modern centralized Botnets employed a list of IP addresses of alternative C&C servers, which will be used in case a C&C server discovered and has been taken offline.

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. There are two central points that forward commands and data between the BotMaster and his Bots.

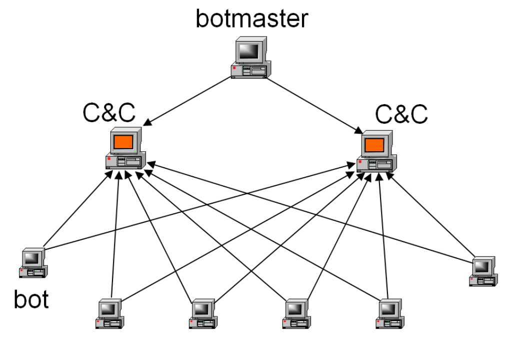

Figure 1. Command and control architecture of a Centralized model.

1) *Botnets 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: (i) low latency communication; (ii) anonymous real-time communication; (iii) ability of Group (many-to-many) and Private (one-to-one) communication; (iv) simple to setup and (v) simple commands. The basic commands are connect to servers, join channels and post messages in the channels; (vi) very flexibility in communication. Therefore IRC protocol is still the most popular protocol being used in Botnet communication [5].

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 bidding. Figure 2 showed that there is one central IRC server that forwards commands and data between the BotMaster and his Bots.

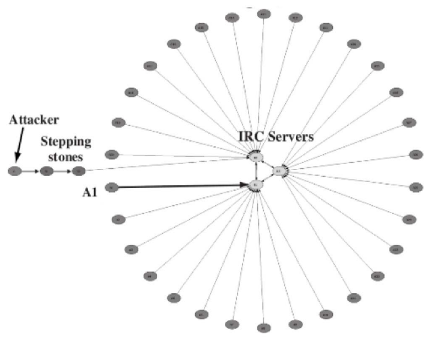

Figure 2. IRC based Botnet (source: Dave Dittrich *et al*. [14])

Puri [15] presented the procedures and mechanism of Botnet based on IRC, as shown in Fig. 3.

Bots infection and control process [15]:

(1) The attacker tries to infect the targets with Bots.
(2) After the Bot is installed on target machine, it will try to connect to IRC server; randomly generated nick name representing that Bot in attacker's private channel.
(3) Request to the DNS server, dynamic mapping IRC server's IP address.
(4) The Bot will join the private IRC channel set up by the attacker and wait for instructions from the attacker. Most of these private IRC channel is set as the encrypted mode.
(5) Attacker sends attack instruction in private IRC channel.
(6) The attacker joins in their private IRC channel, and sends out the authentication password. After the

access is accepted, the attacker will sent instructions which are scheduled previously, such as theft information and denial of service attacks.

(7) Bots receive instructions and launch attacks such as DDoS attacks.

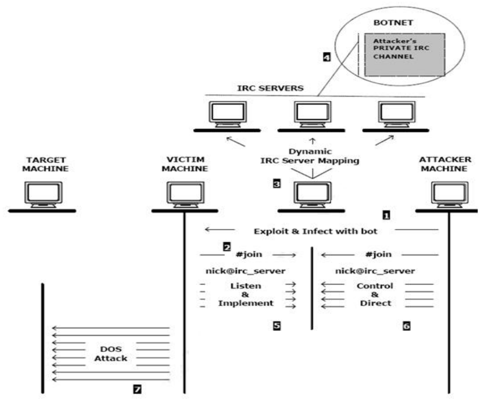

Figure 3. IRC-based Bots Infection & Control Process [15]

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. Usually firewalls block incoming/outgoing traffic to unwanted ports, which often include the IRC port. There are some known Bots using the HTTP protocol, such as Bobax [16], ClickBot [17] and Rustock [18]. Gu *et. al.* [10] pointed out that the HTTP protocol is in a "pull" style, as shown in Fig. 4, and the IRC is in a "push" style. However the architecture of both is same.

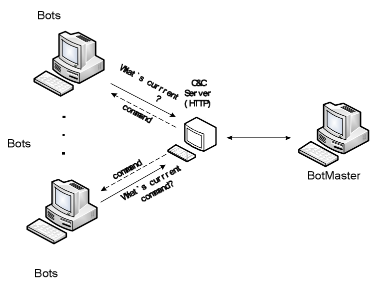

Figure 4. HTTP-based Botnet architecture

B. Decentralized

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 the P2P model, as shown in Fig. 5, 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.

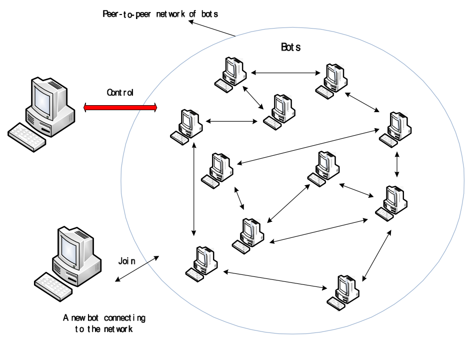
Peer-to-peer network of bots
Bots
Control
Join
A new bot connecting to the network

Figure 5. Example of Peer-to-peer Botnet Architecture

The first known Botnet which utilized P2P model was *Slapper* worm that appeared in 2003[19]. Other famous P2P Botnets include Sinit[20], Phatbot[21], Nugache in 2005[22], Spamthru in 2006 [23] and Storm worm in 2007. P2P Botnets aim at removing or hiding the central point of failure which is the main weakness and vulnerability of Centralized model.

Some P2P Botnets operate to a certain extent decentralized and some completely decentralized. Those Botnets that are completely decentralized allow a BotMaster to inject a command into any Bots, and have it either be broadcasted to a specified node. Since P2P Botnets usually allow commands to be injected at any node in the network, the authentication of commands become essential to prevent other nodes from injecting incorrect commands.

For a better understanding in this model, we have included some characteristics and important features of famous P2P Botnets:

2) *Slapper*: Allows the routing of commands to distinct nodes. Uses Public key/ private key cryptography to authenticate commands. BotMasters sign commands with private key and only those nodes which has corresponding public key can verify the commands [19]. Two important weak points are: (a) its list of known Bots contains all (or almost all) of the Botnet. Thus, one single captured Bot would expose the entire Botnet to defenders [19] (b) its sophisticated communication mechanism produces lot traffic, making it vulnerable to monitoring via network flow analysis.

3) *Sinit:* A sinit Bot uses random probing to find other Bots to communicate with. It can results in an easy detection due to the extensive probing traffic [11].

4) *Nugache:* Its weakness is based on its reliance on a seed list of 22 IP addresses during its bootstrap process [24].

5) *Phatbot:* Uses Gnutella cache server for its bootstrap process which can be easily shutdown. Also its WASTE P2P protocol has a scalability problem across a long network [25].

6) *Strom worm (Trojan.Peacomm):* Uses the Overnet P2P protocol for controlling the Bots. The Overnet protocol implements a distributed hash table based on the Kademlia algorithm as described in [26]. According to Grizzard *et al* [14] analysis of network trace data , the communication protocol for Trojan. Peacomm Bot can be divided into five steps , as describes below :

- *Connect to Overnet*-Bots try to join Overnet network. Each Bot initially has hard-coded binary files which is included the IP addresses of P2P-based Botnet nodes.

- *Search and Download Secondary Injection URL*-Bot uses hard-coded keys to search for and download the URL on the Overnet network.

- *Decrypt Secondary Injection URL*- Bot uses a hard-coded key to decrypt the encrypted URL. Now the Bot has the URL address of a secondary injection executable.

- *Download Secondary Injection* – The Bot downloads the secondary injection from a web server using the decrypted URL. It could be infected files or updated files or list of the P2P nodes.

- *Execute Secondary Injection*-The Bot executes the secondary injection, possibly scheduling future upgrades on the peer-to-peer network or scheduling bot stat tracking at some other resource.

C. *Hybrid*

The reference [25] proposed more advanced hybrid P2P-based Botnet architecture.

The Bots in the proposed P2P Botnet are classified into two groups:

1) *Servant Bots*- Bots in the first group are called as servant Bots, because they behave as both clients and servers, which have static, routable IP addresses and are accessible from the entire Internet.

2) *Client Bots*- Bots in the second group is called as client Bots since they do not accept incoming connections. This group contains the remaining Bots, including:- (a) Bots with dynamically designated IP addresses; (b) Bots with Non-routable IP addresses; and (c) Bots behind firewalls which they cannot be connected from the global Internet.

Proposed Hybrid P2P Botnet as shown in Fig. 6 has the following features [25]:

- Just IP addresses of Servant Bots are candidate for being in peer lists. This ensures that the peer list in each Bot has a long lifetime.

- Each Servant Bot listens on a self-determined port for incoming connections and uses a self-generated symmetric encryption key for incoming traffic. This makes the detection of the Botnet very hard through network flow analysis of the Botnet communication traffic.

- A BotMaster injects commands through any Bots in the Botnet. All Bots have to periodically connect to the Servant Bots in their peer list in order to retrieve commands issued by their BotMaster. When a Bot receive new commands that it has never seen before; it quickly forwards the command to all Servant Bots in its peer list.

- There is a Sensor host (also compromised host) that is controlled by the BotMaster for monitoring the entire Botnet. Sensor host does not have a fixed IP address.

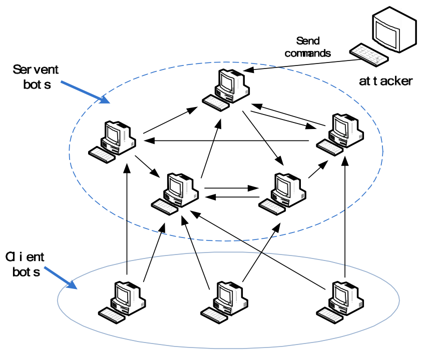

Figure 6. Command and control architecture of the proposed hybrid P2P Botnet[25]

III. Conclusion

Understanding the Botnet Command-and-Control(C&C) channels that are used by attackers is a critical part in recognizing how to best protect against the overall botnet threat. The C&C channels utilized by the Botnets will often show the type and degree of actions an enterprise can follow in either blocking or shutting down a botnet, and the probability of success.

It is also obvious that attackers have been trying for years to move away from Centralized C&C channels, and are achieving some success using Decentralized (P2P) C&C channels over the last 5 or so years. Therefore in this paper we have defined a classification for better understanding of Botnets C&C channels, which is included Centralized, Decentralized, and Hybrid model and tried to evaluate well-known protocols in each of them. Understanding the communication topologies in Botnets is essential to precisely identify, detect, and mitigate the ever-increasing Botnets threats

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] Crimeware:Bots. Web publication, Available at URL: http://www.symantec.com/norton/cybercrime/bots.jsp

[7] J. R. Binkley , S. Singh, "An algorithm for anomaly-based botnet detection," Proc. of 2nd Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI'06), July 2006, pp 43-48.

[8] F. Freiling, T. Holz, and G. Wicherski, "Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks," Proc. of 10th European Symposium On Research In Computer Security (ESORICS'05), 2005

[9] G. Gu, P. Porras, V. Yegneswaran, M. Fong, and W. Lee, "Bothunter: Detecting malware infection through ids-driven dialog correlation," Proc. Of the 16th USENIX Security Symposium, 2006.

[10] G. Gu, J. Zhang, and W. Lee, "BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic," Proc. of the 15th Annual Network and Distributed System Security Symposium (NDSS'08), San Diego, CA, February 2008.

[11] A. Karasaridis, B. Rexroad, and D. Hoein, "Widescale botnet detection and characterization," Proc. of Hot Topics in Understanding Botnets (HotBots'07), April 2007.

[12] 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 *39*th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'09), June 2009.

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

[14] J.B. Grizzard, V. Sharma and C. Nunnery. "Peer-to-Peer Botnets: Overview and case study," Proc. of the 1st Workshop on Hot Topics in Understanding Botnets (HotBots 2007). 2007.

[15] R. Puri. Bots & Botnet : An Overview. Research on Topics in Information Security. 2003, 08.

[16] J.Stewart.Bobaxtrojananalysis. http://www.secureworks.com/research/threats/bobax/

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

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

[19] Iv´an Arce and Elias Levy. An analysis of the slapper worm. IEEE Security and Privacy Magazine, 1(1):82–87, 2003.

[20] J. Stewart. Sinit P2P trojan analysis. Web publication. Available at URL: http://www.secureworks.com/research/threats/sinit,2003

[21] J. Stewart. Phatbot trojan analysis. Web publication. Available at URL: http://www.secureworks.com/research/threats/phatbot/,2004

[22] S. Stover, D. Dittrich, J. Hernandez, and S. Dietrich. Analysis of the storm and nugache trojans: P2p is here. *;login:*, 32(6), 2007.

[23] SecureWorks. SpamThru trojan analysis, October 2006. http://www.secureworks.com/analysis/spamthru/

[24] R. Lemos. (2006, May) Bot software looks to improve peerage. http://www.securityfocus.com/news/11390

[25] P. Wang , S. Sparks, C. Zou. An advanced hybrid peer-to-peer botnet. Hotbots'07 workshop program, School of Electrical Engineering and Computer Science, University of Central Florida, 2007.

[26] P. Maymounkov, D. Mazières, "Kademlia: A peer-to-peer information system based on the XOR metric," 1st International Workshop on Peer-to-Peer Systems, March 2002, pp. 53-62.