Botnet Technologies: A Survey on Attack and Defense Mechanism

Surender Kumar Verma[1], Bhawna Chaudhri[2]

[1]Computer Emergency Response Team, India (CERT-In)
[2]Jawaharlal Nehru University, New Delhi

ABSTRACT

The threat model on internet has undergone a tremendous transformation. The attacks on the cyber infrastructure of an organization are no more generated by a single computing device, rather than by a group of computers attacking in tandem. These attack performing devices are mostly a group of compromised computing machines, called bot instance. These devices are controlled remotely by the attacker through a special high performance computing device called Command and Control (C&C) server. In this survey, we discuss a modus-operandi of existing botnets and their goals. Further we discuss the technology aspect of the botnet infrastructure and the defence mechanism employed by the corporations.

1. INTRODUCTION

Botnet is a carefully developed piece of malware which consists of remotely controlled computing devices by a special machine called Command and Control (C&C) server. These remotely controlled devices are often infected with some well know categories of malwares like viruses, worms, spyware and other malicious software [6]. All these comprised machines are used in conducting variety of cyber crimes like information stealing, Distributed Denial of Service (DDoS) attacks and spamming against the enterprise and the government internet resources. Despite the effort of many research and the commercial organizations, the botnet technology continues to be more sophisticated in conducting the attacks on the organizations. Security organizations and global CERTs (Computer Emergency Response Team) has identifies the botnet with more than 1 million infected hosts compromised [19]. This paper explores the various technologies used in creation of a typical botnet and the defence mechanism used to protect the organization network against a typical botnet attack [20].

2. PROPAGATION AND COMPROMISE

In most cases malicious software is installed by attacker in the victim machines (bot instances). The attack vector for compromising a machine has evolved over the time. There has been transition from manual installation of malware to the multiple automatic compromises of victim machines. The vulnerabilities in different components of the victim machine could be used for the propagation vector. Some of these are listed as below:

- **Operating systems:** Designing the attack vector to exploit the operating system vulnerability is difficult task. Since the frequent updates are available in the major operating systems, the chances of detection of the compromise are very high. Major advantage of this technique is that a large number of hosts could be infected using this technique in short span of time. Morris worm has been popular exploit in this class.

- **Applications:** There has been a constant shift toward targeting the higher level application like web browsers, media players, email applications etc. Malicious URLs has been used in exploiting the applications like Flash player and installing Trojan, worms and other malicious software [21]. Designing exploit for application is a complicated task. It is difficult to detect the compromise in this class of exploits.

3. THE COMMAND AND CONTROL MECHANISM

The major design issue in the botnet infrastructure is how to communicate with the compromised hosts (bot instance) [6]. An attacker is required to send instructions and commands to the bot instance as often as possible. The communication between the bot instance and the C&C server need to be covert for the effectiveness of the botnet attacks. The various command and controls topologies used in the botnets are given as:

- **Centralised**: In this topology, there is single C&C server controlling all other compromised bot instances. The design complexity of such a C&C mechanism is not complicated. A C&C server can deliver the command to the bot instance with no latency or negligible latency. The major disadvantage with this technique is that the C&C server is highly detectable.

- **Peer to Peer (P2P):** In this topology, there is no dedicated command and control server. All the bot instances are capable to deliver commands to each other. The design complexity of such a system is high. The major advantage of this topology is that the delectability of such system is very low since there is no single C&C server to deliver the commands. The communication mechanism in peer to peer topology is much harder to disrupt. The message latency in such a system is high. SDBot (known as **rBot**) is a typical example of this topology [21].

- **Hybrid topology:** This is the combination of above two topologies. In this topology there are multiple C&C servers organised in peer to peer architecture. So even if one C&C server is disrupted, others could be used to send commands to the bot instances [9].

4. COMMUNICATION OF C&C SERVERS WITH BOTS

A botnet without C&C communication is just a random collection of infected machines. In other words the C&C communication is the backbone of any botnet. This communication must be covert in nature and should involve the minimum latency to deliver the control messages to the bot instances. In order to communicate with a bot instance, the C&C server could use either existing communication protocols or could use the custom protocols. Some of the techniques used for the purpose are discussed below:

IRC based C&C

Historically, IRC has proven to be very successful because of its communication protocol. Basically IRC is useful in audio/video conferencing, online chat, and text-based communication functions. IRC has some advantages as IRC permits C&C to issue commands easily [18]. C&C server creates an IRC channel for victims to join and broadcast messages to all. Secondly, C&C traffic is hard to differentiate from basic IRC traffic. Third, the topology used by IRC-based botnet is centralized, so it can communicate directly with all bots. IRC-based C&C infrastructure is easy to design and serves as an effective mode for distributing C&C instructions.

HTTP based C&C

HTTP based protocols used by botnets are hard to detect. Using the HTTP protocol, botnet usually hides traffics in normal web traffics, so it can easily avoids security devices [10]. The HTTP traffic uses port 80, which has a high proportion of normal traffic and is used in majority of Web communications. C&C control commands to the infected hosts could easily be intermixed with the normal web traffic.

Peer to peer C&C [17]

In P2P architecture, any node can act as client or server, or both simultaneously. Further, in a P2P C&C infrastructure, centralized server is not needed for updates and commands. Rather, for required updates the bots communicate with one another and the machine with latest malware version transfers it to the other machines. Since there is no single point of failure so it is highly difficult for security mechanism to dismantle a P2P botnet even though herder is taken offline, the botnet may continue to function.

Social Network C&C

Social networks based C&C servers are have been there is existence since a while. Some botnet used Twitter and Facebook profile status to deliver command to the bot instance. For example any change in status of a profile is interpreted as a command by the connected bot instance. Dummy profiles are created on social networking sites and then set of commands are updated on to these profiles [3]. Once the bot instance is infected with a malware such as a Trojan, the malware receives the commands from the profile. In this scenario, even if a malicious profile is found and blocked, the malware writer can easily create other dummy profiles.

C&C in the Cloud

The cloud service such as Amazon Web Services (AWS), are extensively used by the botnet developers to maintain their C&C control communication [2]. Many of such cloud services are freely available and serve as major motivation to the malware writers. AppEngines have been extensively lucrative to botnet developers because of their Omni-presence and high availability. This allows performing the attack with high magnitude of traffic before they are detected. Futher, AppEngine allows the malware writer to enjoy absolutely anonymity with updating commands on AppEngine from anywhere. Zeus [19] botnet on Amazon’s EC2 cloud computing infrastructure is example of such a botnet. Nowadays it has been extremely difficult for criminals to use the legitimate cloud services to host their C&C server.

5. RALLYING MECHANISM

After compromising the victim host, botmaster (C&C server) wants to register the newly discovered bot with the botnet in hidden and portable manner. This is the actual connection of a newly compromised bot instance with the C&C server. This mechanism is known as rallying. Rallying mechanisms are very crucial part of botnet design [4]. Most commonly used rallying mechanisms as follows:

Hardcoded IP Address

Upon successful compromise, the bot instance is required to contact C&C server to receive the commands. For this purpose, the Internet Protocol (lP) address of C&C server is needed which is provided in binary executable file of the bot instance itself Nugache [23]. Earlier botnets were tempted to use this technique as it eliminated the DNS communication from the picture. This rallying mechanism is not in demand recently as lawful supremacy might get the IP address of the C&C server from the binary executable file of bot instance and might block the traffic specific to that server. Once the C&C server is disabled, bots owned by this server becomes futile because they cannot receive any instructions or performing attack.

Dynamic DNS domain name

The bots usually contains hard-coded domain names, assigned by Domain Name Systems (DNS) providers which enable the C&C server to relocate its bots easily. Sometime a C&C server uses several other subordinate C&C servers by placing a list of domain names in the binary executable of bot instance. In order not to be recognised and expelled, it uses multiple domain names and to keep the botnet mobile. If somehow connection to C&C server is unsuccessful, then through DNS, the bot instance will be switched to a different C&C server. This redirection behaviour is called herding. Herding introduces mobility and undetectability to the botnets.

Distributed DNS service

With this method botnets drive their own distributed Domain Name System (DNS) services at locations that are not reachable by law enforcement [8]. To receive the information regarding IP of the C&C server, the bot needs to communicate with its own DNS server. Because the service hosts are not cooperative at times and there are many systems, possibly in different territories, this rallying mechanism is resistant to detection as these servers use a high
port for communication.

6. BOTNET DETECTION TECHNIQUES

Detection by Signatures: There are specific strings and regular expressions are used to represent the set of suspicious pattern originating from botnet traffic. The n-gram analysis is performed on this set to evaluate the patterns to determine whether a particular communication belong to a bot infected host [11].

Detection by Attack Behaviour: The amount of traffic sent and received in a given span of time could be the indicator of the behaviour of the botnet. The traffic properties of spam server and spam payload have been used to construct a framework to generate spam signatures [16].

Detection by Behavior of Cooperation: This technique of botnet detection has been in trend and used by many botnet researchers. Bothunter [14] designed the bot compromise phase an ordered communication between an inside network host and one or more outside network bot instances. A statistical technique is used in Botsniffer [15] to locate the botnets based on their clustering behaviours (e.g. propagation of spam, hash based scanning of binaries) in a central topology. A Clustering based framework for botnet detection is used in Botminner [13] that uses clustering on C&C communication traffic and activities performed.

The cross correlation on the clusters is then used to generate the final result for botnet detection. Karasaridis [18] created a detection scheme based on calculation of the distances between a pre-defined IRC traffic flow and monitored flow data. Three metrics (response, relationship and synchronization) are defined by Akiyama [1] to infer the cooperative behaviour of various botnets. A time stamp based correlation method is proposed by Strayer [22] in a multi dimensional space about inter packet arrival time and size of the packet. A group based DNS traffic activities used by Chois [5] to perform botnet detection.

CONCLUSIONS

The proposed survey provides an exhaustive approach that could be used by botnet infrastructure and its solution space. The inherent logical connection between various botnet techniques and the detection approach has been highlighted in this survey. This information is very important for the security researches to conduct the further analysis of the unknown botnets. Different botnet detection mechanism has been discussed in this survey that advocates the passive detection and restricts active methods. The classification by other dimension is left as future Avenue of research.

REFERENCES

[1]. M. Akiyama, T. Kawamoto, M. Shimamura, T. Yokoyama, Y. Kadobayashi, and S. Yamaguchi. A proposal of metrics for botnet detection based on its cooperative behavior. In Proceedings of the 2007 International Symposium on Applications and the Internet Workshops(SAINT-W'07), Washing-ton, DC, May 2007.

[2]. D. S. Anderson, C. Fleizach, S. Savage, and G. M. Voelker. Spamscatter: Characterizing internet scam hosting infrastructure. In Proceedings of the 16th USENIX Security Symposium (Security'07), Boston, MA, August 2007.

[3]. J. R. Binkley and S. Singh. An algorithm for anomaly-based botnet detection. In Proceedings of the 2nd conference on Steps to Reducing Unwanted Traffic on the Internet (SRUTI'06), San Jose, CA, July 2006.

[4]. A. Broadsky and D. Broksky. A distributed content independent method for spam detection. In First Workshop on Hot Topics in Understanding Botnets (HotBots'07), Cambridge, MA, April 2007.

[5]. H. Choi, H. Lee, H. Lee, and H. Kim. Botnet detection by monitoring group activities in dns traffic. In Proceedings of the 7th IEEE International Conference on Computer and In-formation Technology (CIT'07), Washington, DC, October 2007.

[6]. E. Cooke, F. Jahanian, and D. McPherson. The zombie roundup: Understanding, detecting, and disrupting botnets. In Proceedings of the Steps to Reducing Unwanted Traffic on the Internet (SRUTI 2005 Workshop), Cambridge, MA, July 2005.

[7]. D. Dagon, G. Gu, C. P. Lee, and W. Lee. A taxonomy of botnet structures. In Twenty-Third Annual Computer Security Applications Conference (ACSAC'07), Florida, USA, November 2007.

[8]. D. Dagon, C. Zou, and W. Lee. Modeling botnet propagation using time zones. In Proceedings of the 13rd Network and Distributed System Security Symposium (NDSS'06), San Diego, CA, February 2006.

[9]. D. Dittrich and S. Dietrich. P2p as botnet command and control: a deeper insight. In Proceedings of the 2008 3rd International Conference on Malicious and Unwanted Soft-ware (Malware 2008), Alexandria, VA, Oct 2008.

[10]. S. Gianvecchio, M. Xie, Z. Wu, and H. Wang. Measurement and classification of humans and bots in internet chat. In Proceedings of the 17th USENIX Security Symposium (Se-curity'08), San Jose, CA, July 2008.

[11]. J. Goebel and T. Holz. Rishi: Identify bot contaminated hosts by irc nickname evaluation. In First Workshop on Hot Topics in Understanding Botnets (HotBots'07), Cambridge, MA, April 2007.

[12]. J. B. Grizzard, V. Sharma, C. Nunnery, B. B. Kang, and D. Dagon. Peer-to-Peer Botnets: Overview and case study. In First Workshop on Hot Topics in Understanding Botnets (HotBots'07), Cambridge, MA, April 2007.

[13]. G. Gu, R. Perdisci, junjie Zhang, and W. Lee. BotMiner: Clustering analysis of network traffic for protocol- and structure-independent botnet detection. In Proceedings of the 17th USENIX Security Symposium (Security'08), San Jose, CA, July 2008.

[14]. G. Gu, P. Porras, V. Yegneswaran, M. Frog, and W. Lee. BotHunter: Detecting malware infection through ids-driven dialog correlation. In Proceedings of the 16th USENIX Security Symposium (Security'07), Boston, MA, August 2007.

[15]. G. Gu, J. Zhang, and W. Lee. BotSniffer: Detecting botnet command and control channels in network traffic. In Pro-ceedings of the 15th Annual Network & Distributed System Security Symposium (NDSS'08), San Diego, CA, February 2008.

[16]. T. Holz, M. Steiner, F. Dahl, E. Biersack, and F. Freiling. Measurements and Mitigation of Peer-to-Peer-based Bot-nets: A case study on storm worm. In First Usenix Workshop on Large-scale Exploits and Emergent Threats (LEET'08), San Francisco, CA, April 2008.

[17]. C. Kanich, K. Lechenko, B. Enright, G. M. Voelker, and Savage. The Heisenbot Uncertainty Problem: Challenges in separating bots from chaff. In First Usenix Workshop on Large-scale Exploits and Emergent Threats (LEET'08), San Francisco, CA, April 2008.

[18]. A. Karasaridis, B. Rexroad, and D. Hoeflin. Wide-scale botnet detection and characterization. In First Workshop on Hot Topics in Understanding Botnets (HotBots'07), Cambridge, MA, April 2007.

[19]. L. McLaughlin. Bot software spreads, causes new worries. IEEE Distributed Systems Online, 5(6), June 2004.

[20]. Microsoft. Microsoft security intelligence report: July-december 2006. http://www.microsoft.com/technet/security/default.mspx, May 2007.

[21]. N. Provos, D. McNamee, P. Mavrommatis, K. Wang, and Modadugu. The ghost in the browser: Analysis of web-based malware. In First Workshop on Hot Topics in Under-standing Botnets, HotBots'07. USENIX, 2007.

[22]. M. A. Rajab, J. Zarfoss, F. Monrose, and A. Terzis. A multi-faceted approach to understanding the botnet phenomenon. In Proceedings of Internet Measurement Conference 2006 (IMC'06), Rio de Janeiro, Brazil, October 2006.

[23]. M. A. Rajab, J. Zarfoss, F. Monrose, and A. Terzis. My Botnet is Bigger than Yours (Maybe, Better than Yours): why size estimates remain challenging. In First Workshop on Hot Topics in Understanding