C&C Techniques in Botnet Development

Félix Brezo[1], José Gaviria de la Puerta[1], Igor Santos[1], David Barroso[2], and Pablo G. Bringas[1]

[1] DeustoTech Computing - University of Deusto, Bilbao BI 48007, Spain,
{felix.brezo, jgaviria, isantos, pablo.garcia.bringas}@deusto.es,
WWW home page: http://www.s3lab.deusto.es
[2] Telefonica I+D, Madrid MA, Spain
dbarroso@lostinsecurity.com

Abstract. Botnets are one of the most important threats towards nowadays users of the Internet. The joint of malware capabilities to be exploited in the network services and the increasing number of daily transactions performed in the cloud, makes them an attractive target for cybercriminals who have evolved their old IRC-based communication channels, into decentralized P2P networks, HTTP/S botnets and even Twitter-controlled networks. Against this background, this article analyses the threat that will affect computer networks in the upcoming years by going through these different Command & Control channels used by botmasters to keep the control of their hijacked networks.

Keywords: botnets, crimeware, cyberfraud, C&C, source analysis,

1 Introduction

More often than ever, computer security is gaining its place in human being's ordinary life. The Anonymous attacks on different e-platforms [1, 2] have brought closer this reality to ordinary people which will have heard about DDoS relegating to the memories of the elder the times in which computer security experts only appeared in the media to speak about punctual threats collapsing thousands of computers.

The scientific community is alerted about the problems that would involve an increase in the complexity of traditional communication channels in the next generation of botnets. This is a hot topic that has concerned security agencies in the world for a long time. One of the first important ones was the "Operation Bot Roast" [3] in 2007, by which the FBI detected a botnet compounded by more than a million compromised computers. In 2010, an action leaded by the Spanish *Guardia Civil* in cooperation with Panda Labs dismantled the "Botnet Butterfly", with 12.7 million computers committed [4]. Those capabilities have not been misjudged by governments and armies. In fact, the DDoS attacks received by Estonia in 2007 lead to the creation of Cooperative Cyber Defence Centre of Excellence [5], while their functionalities to achieve information about military [6] and economical targets had been already suggested by Colonel Charles W. Williamson III [7].

This document is structured as follows. Section 2 explains briefly how a botnet works and which are some of the most well-known functionalities. Section 3 analyses the characteristics of some of the most important botnet tipologies going through particular cases. Section 4 delves into the different approaches that exist for detecting and tracking these networks, together with the advance of some outlines to work on in the future. Finally, Section 5 brings down the conclusions after analysing the menaces commented all along this document.

2 The botnet threat

Microsoft's End to End Trust defines botnets as "networks of dedicated and remoted controlled computers by one or more cybercriminals" [8]. Botnets are attractive to these people for two reasons: they are easily managed and can produce direct or indirect economical benefit for the botnet controller. Nowadays bots, are a mixture of threats closely related to other previous' areas of malware, as they can spread themselves like worms, are hidden as many viruses and allow remote control of the infected machine by third parties. These circumstances, together with other evidence related to the writing of code by means of cooperative efforts (as what happens with SDBot, whose code is commented even by different authors), allow the proliferation of a wide range of variations and mutations of bots based on the specific purpose for which they are sought.

These functionalities can vary depending on the complexity of the final target, but we have to assume that may include a combination of the following:

- **DDoS support**. A denial of service attack, also called DoS, is an attack on a computer system or network that causes a service or resource being inaccessible for legitimate users. A Distributed DoS (DDoS) will cause the loss of network connectivity by consuming the bandwidth of the victim network or overloading its computational resources. At the same time, Roman Studer states that DDoS attacks can cause various types of financial costs to a company [9]. For those that depend on online functionalities like e-banking and ecommerce businesses, the revenue out of online functions will be lost during system unavailability. However, Studer also points out that "they may also sufer monetary penalties as a result of losing the ability to meet service level agreements and by failing to provide an agreed service the company could cause damage to a third party and be faced litigation and charged accordingly". Furthermore, investment on recovering tools and backup policies should have been performed, as well as an in-depth analysis of what had happened to prevent it from happening again. It may also affect the companies image which lately may also imply a fall of the stock price in the extremely sensible markets of new technologies.
- **Remote execution of protected programs**. With very different targets, botnets may only be the first step into a deeper infection in the system. Thus, attackers may get full control of the infected machines to perform different covert operations without the victim noticing it.

- **Keylogging**. A keylogger is a software (or even hardware) device that handles specific record keystrokes on the keyboard to store them in a file or to send them via Internet. Often used as a daemon malware type, the objective of this feature is to open a door to capture login and password details, credit card information and other sensible information.
- **Click fraud**. Click fraud refers to a type of Internet fraud, in which the target of the attack are the pay-per-click services (such as Google AdSense). Though click fraud can also be performed manually, with the help of certain kinds of software its automatication makes the fraud more profitable. By means of these tools, fraudsters can manipulate billing systems lying behind target by increasing the traffic and the amount of clicks.
- **Spam**. Practically all the spam sent worldwide, is mailed through machines under direct control of spam operators using botnets to accomplish such objectives. According to SpamHaus[3], amongst 300 and 400 spammers were responsible for themselves of the 80% of global traffic of such malicious content. Microsoft [8] estimates that botnets are responsible for 87 percent of all the e-mail unsolicited, equivalent to about 151,000 million e-mail messages a day. At the same time, on February 25, 2010, Microsoft, industry partners, academic and legal communities announced a successful collaborative effort to disable a botnet win32/Waledac important call [10].

3 Botnet samples

In this section we are going to analyse some aspects in the procedure of coding a botnet. With recent attacks on several credit card servers as a retaliatory measure by the constraints brought by these companies to Wikileaks [11, 12], it has been shown that even in 2011 IRC remains being a channel used to transmit commands in Anonymous botnets [1, 2]. However, more modern techniques should be developed in the design of a reliable botnet detection system to fight the proliferation of new C&C techniques such as chat (or private messages) in the P2P servers, static HTML pages connections, more complex HTTP-based botnets and more modern Twitter or Pastebin based ones. To complete this objective, the following botnets have been studied.

3.1 IRC

IRC-based botnets compounded the initial stage of the botnet threat with an impact between 2005 and 2008 [13]. Those networks were connected periodically to a specified IRC channel awaiting further instructions, being capable of retransmitting the commands in real time, either through private messages (PRIVMSG IRC) or through the publication of thematic messages (TOPIC).

[3] ROKSO: Register Of Known Spam Operations http://www.spamhaus.org/rokso

Sdbot. Sdbot was a backdoor that allows hackers to gain remote access to the affected computer to carry out malicious actions that compromise user confidentiality and impede the tasks. Sdbot also uses an own IRC client as C&C channel. Amongst his malicious functionalities it can launch denial of service attacks against websites and download and execute files on the infected computer. Nevertheless, Sdbot requires user intervention for propagation leading to a great variety of means which include, among others, floppy disks, CD-ROMs, email messages with attachments, Internet downloads, file transfers via FTP, IRC channels, file sharing, P2P networks, etc.

Agobot. Agobot, also frequently known as Gaobot, is a family of computer worms used for botnet creation and released under version 2 of the GNU General Public License. The Agobot source code describes it as: "a modular IRC bot for Win32/Linux" and its first implementation is attributed to Axel "Ago" Gembe, a German programmer. Agobot is a multi-threaded and mostly object oriented program written in C++ as well as a small amount of assembly. Browsing the project, anyone will be able to surf easily inside the forest of modular .cpp files: config.cpp, ddos.cpp (whose starting command is *ddos.httpflood*), harvest_emails.cpp amongst others, which are good examples of the multiple capabilities of this software. Agobot is another example of a botnet that requires little or no programming knowledge to be used as its modularity permits the writer to focus on just a concrete part of the programming.

Rbot. Rbot is a worm that spreads through network shares. It searches and makes a list of the shared folders used for application downloads (eg. P2P), where it will release a copy of itself. It has some features that allow it to act as a backdoor developing botnet functionalities using IRC as channel. The worm uses its engine to connect to a preset IRC channel to wait for commands from a remote attacker. Some of the commands that an attacker may send to any victim are: launching DDoS attacks; getting registration keys (Product ID and CD Keys); adding and deleting shares, groups and users; performing caches attack (*flushing*); screenshots; image and video captured via webcam; executing remote files; running a keylogger and password stealing procedures, etc.

3.2 P2P

Traditionally propagated through shared binary in other P2P networks by making use of the P2P protocol, they have the advantage of being more difficult to destabilize as they do not have a unique core from which issuing orders and/or sharing resources and information. They make use of the facilities of traditional P2P networks which allow high connection and disconnection ratios. Some examples of P2P network architectures are the aforementioned Trojan.Peacomm and Stormnet [14].

Special mention must be the *parasitic botnets* [15, 16], whose communication channel is also quite simple: first, a bot is selected to perform a specific search with a specific title (which can be fixed or calculated by some sort of algorithm) that would allow each bot to identify a) what other file must be found to reveal the commands to execute or b), in which searching response the command to be sent will be coded not needing to download any file. Thus, the botmaster will only receive a positive response if a member of the same botnet contains it, initiating then the communication protocol. This philosophy uses *in-band-messages* (common P2P traffic) which permits the sending and reception of commands [17]. The pro is that this traffic might be easily confused with legitimate traffic within the network, difficulting its detection and complicating the analysis of traffic as this approach will sue a pre-filtering process.

However, Shishir Nagaraja et al. [18] have worked on P2P network detection. Concretely, they have developed some research to find whether ISPs can detect those efficient communication structures of P2P botnets so as to use this basis for botnet defense. The point is that ISPs, enterprise networks, and IDSs have significant visibility into these communication patterns due to the potentially large number of paths between bots that traverse their routers, making them a good starting point to detect this traffic.

IMMONIA. IMMONIA is a P2P bot designed to be spread by its own means on IRC platforms. It is written in AutoIt v3[4]. Being coded using a scripting programming language, whose code[5] was attributed to someone named Hypoz in 2011, is fully documented and pretty detailed commented. The most interesting point here is its decentralized P2P structure running through IRC platforms developing greater structural complexity because all of them can act as both, client and server, being more difficult to intercept and study.

3.3 HTTP/HTTPS

In the HTTP-based, the botmaster informs the infected computers from the new list of commands to execute, updating the contents of a web page that bots are to periodically visit. Some modern examples are Flu and Zeus, described below.

Zeus. Zeus botnet is an evolution of a Trojan horse that stole banking information by Man-in-the-browser keystroke logging and Form Grabbing, being mai spread through drive-by downloads and phishing schemes. First identified in July 2007 when it was used to steal information from the United States Department of Transportation [3], it became more widespread in March 2009. By that time, Zeus had infected over 3.6 million computers in the United States. Binsaleeh et al. presented in 2010 [19] a reverse engineering analysis on the Zeus crimeware toolkit, which they defined as "one of the recent and powerful crimeware tools that emerged in the Internet underground community to control botnets".

[4] AutoIt v3 is a freeware BASIC-like scripting language designed for automating the Windows GUI and other general scripting. Its default extension is .au3.

[5] By this publication available on http://www.mediafire.com/?1wdq1o076amww03.

Its source code was released in May 2011 after being sold in different underground forums [20]. Pretty well balanced, the attackers C&C channel is written in PHP, while the client side of the Malware is coded C++. According to its documentation, Zeus developers were working on more than 15 updates and modifications only in 2010 [21].

– **DDBB connection**. The Zeus control panel was developed on PHP 5.2.6, so there are PHP.ini settings included that will need to be configured, as well as MySQL setting recommendations.
– **Server-side capabilities**. Zeus has Socks 4/4a/5 support with IPv4 and IPv6 via UDP. This will allow the botmaster to connect to an infected host even if there is a NAT in place or a restrictive firewall.
– **C&C channels**. The botmaster can do a number of things with the stolen data and the infected systems. One interesting aspect of the Zeus control panel is the ability to search for targeted information: IP, NAT, number of active bots online, country, operating system, and other key demographics. The botmaster can have alerts delivered via IM on Jabber if they wish to let the botmaster know when a victim is using a given bank, allowing them to trigger a session capture on demand.

The following are some of the evil functionalities implemented on Zeus:

– **HTTP/S**. Zeus will intercept traffic from Internet Explorer, Firefox, and other browsers. It can modify pages on the fly, inject forms, or redirect the system to a fake page. It can also block access and log to URLs on demand.
– **Data Harvesting**. Zeus will collect all of the stored information on a system that is not immediately protected (eg. browser cookies). Other harvesting behaviours include: import certificates, intercept FTP-Login and POP3 data and a special functionality as keylogger and capturing desktop screenshots.

Flu. Flu[6] is a reverse Trojan designed to permit the construction of a botnet. It is designed with a client-server architecture, where, the server, is a small executable programmed in C# which will infect any Windows operating system to get control of the machine that is hosting. The client is developed in PHP and runs on an Apache web server. It aims to provide commands to all those individual servers that Flu has spread across the Internet to obtain information from the machines where they are installed and to store it in the web server. The user information may be accessed at any time from a graphical interface developed in HTML and PHP using the browser. The purpose of the project is to allow all those users who want to learn in the development of remote management tools, to take advantage of the expertise of others. Also, the project also aims to raise awareness among users about the dangers of malware, showing the operation of a simple way through and showing Flu protection and disinfection.

The source is quite well commented to match the educational purposes. At the same time, there exists a fully accessible document ready to be checked by

[6] www.-flu-project.com

occasional writers at the official webpage. The basic deployment of the botnet is fully explained by Juan Antonio Calles and Pablo González [22] in the *Flu Trojan's User Guide* with very brief explanations.

- **DDBB connection**. The DB structure in the free version of Flu is not professional. It includes a pair of tables: one for botnet users (with users and passwords stored in plain text) and another one to store the details of the machines belonging to it. Another aspect to pinpoint is the way in which the users connect to the database: the DB username and password are directly hardcoded in the PHP web server.
- **C&C channels**. The bidirectional communications in Flu are performed by means of an Internet browser. On the one hand, the infected victim requests the .xml file that contains the list of commands to be executed to the web server. Once the victim has executed the requested commands, it returns its answers to the web server. All the information is encrypted by default using the AES algorithm and a key of 128 bits. In the web interface, the botmaster will be able to track all the infected nodes, together with an option to load, in each and every node, a remote shell to give personalised orders to that node. In another tab, group orders can also be dispatched, by selecting predefined-attacks or launching commands in every victim's console.
- **Bot program creation**. The bot generation takes place by means of an automated generator which receives as parameter the URL where the botnet server will be allocated, together with the absolute path to the .xml commands file.

Though Flu b0.4 is conceived as an educational project, it includes some interesting features even for malware writers: individual and group remote shell execution; the possibility of executing powershell commands; information gathering, including, Operative System, installed hardware, drivers, etc.; screen printing, user account creation with full control capabilities; file stealing (though for design limitations, up to 3.5 MB only); Windows firewall deactivation; process registering in the Windows Registry to guarantee that each time that the user logs in, Flu will start automatically; and keylogging functionalities.

Though Flu Project has an educational goal, there also exists a more complex version of Flu oriented to fight pederasts called Flu-AD (standing for *Flu-Anti Depredadores* or Flu-CounterPredator in English) presented in No cON Name 2011 with some fully implemented characteristics as detailed below.

- **Rootkit**. To hide operations in the Task Manager.
- **Firm**. To obtain the corresponding hash whenever a new evidence is recovered in the *predator* computer, so as to guarantee that the evidence has not being modified afterwards.
- **Webcam and Sound capture**.
- **Crypter**. So as not to be detected by antivirus software.
- **Stealer**. So as to recover usernames and passwords of *predators* accounts.

Despite all the aforementioned capabilities, a proper isolation of the connections performed by the infected nodes may lead to the site where the botnet server is installed. Given the aforementioned DB restrictions, it should be easy to gain control of the botnet by accessing the users table in the DB to manually put it out of service from inside, by deleting bot instances and modifying permissions. At the same time, Flu Project developers provide a very simple tool that removes completely any trace created after the infection. However, it will only be useful if the executable file is named as flu.exe, making it useless in a real environment.

Prablinha. Prablinha[7] is another reverse Trojan designed to permit the construction of a botnet, made open-source in December 2010. It is also designed with a client-server architecture, where, the zombies in designed in C# are generated by a bot generator also designed in C#. The management utilities are developed in PHP and run on an Apache web server. Prablinha was also thought to teach users about how easy DDoS could be made. The system is conceived to perform HTTP, UDP and TCP Denial of service attacks, as well as to provide the attacker a simple remote shell in the victim's system.

3.4 New Trends: social malware

The widespread of social networks is also attractive for malware writers. In fact, very recently, botmasters have begun to exploit social network websites such as Twitter [23] with pretty good results. As social networks existence depends on being online 24/7, some programmers have found in them a good starting point to successfully host their Command & Control platforms [24]. Additionally, if we add that users trust in their contacts suggestions to click and visit the links they receive, much more than on anonymous webpages, social networks will undoubtedly gain more and more attention in short-term malware evolution.

4 Current work on generic countermeasures

At the time of detection and tracking botnets, there are two different approaches: the active one, usually based on *honeypots*[8] as the one developed in *The Honeynet Project* [25] and the passive one, monitoring the network traffic. Nevertheless, one of the main obstacles when dealing with botnets arises when trying to simulate a real environment under controlled circumstances, as some authors have stated when developing tools to generate simulated botnet traffic for researching purposes in large-scale networks (Massi et al. [26]). We can divide the detection efforts in 4 main lines:

- **By means of signatures**. Detection using signatures is currently facing mayor challenges posed by modern techniques capable of developing polymorphic families [27].

[7] http://www.indetectables.net/foro/viewtopic.php?t=29086

[8] A honeypot is an entity created only to be attacked in order to detect and analyse new threats and the effects of these connections.

- **Cooperative behaviour**. The target is to make statistical attacks seeking behaviour profiles that differ from that performed by a non-infected user.
- **Offensive behaviour**. Assuming that botnets send massive amounts of information in relatively short periods of time, Xie et al. [28] uses the volume of data sent along with the information obtained from spam servers for tracking such contents before they cause real damage.
- **End-user tracking**. Authors like Ormerod et al. face this issue by defaming botnet toolkits through prosecuting the end-users of stolen credentials [29].

5 Conclusion

The communication between bots has undergone major changes since the use of IRC clients. Moreover, to facilitate the exponential growth of such networks, it is necessary to ensure that those commands sent by the botmaster are not lost without being received, interpreted and/or executed by any of the peers infected.

In the face of the alarming growth of malicious applications, which numbers have been beating every year [30], strong efforts will be needed to carry out more researching work in this field. In this way, a report presented by the computer security firm Kaspersky, as part of Technology Day 2011 showed that botnets are one of the most powerful threats in nowadays cyberspace, with perspectives of going on evolving “dramatically” in 2020 as they will incorporate more and more mobile devices with Internet connection.

At the same time, a new battlefield is appearing. The power and capabilities of mobile devices, tablets and smartphones are arising new security concerns. The amount of personal data stored in them and the increasing daily transactions that will be performed with such devices in the future, highlights them as a very attractive target for malware writers.

Although being dealing with a field whose expertise is still under development, we have proposed in this paper some outlines of work to cope with a phenomenon that could jeopardize all services connected to the network in the middle/long term. Thus, each and every organisation with access to the Internet must be prepared proactively, assuming, as part of the computer security protocols, that our systems may suffer in the future from hypothetical massive attacks linked to this new form of organized crime.

References

1. Lillington, K.: Time to talk: Anonymus speaks outs
2. InfoSecurity: Anonymus hacking group uses IRC channles to co-ordinate DDoS attacks. (2011)
3. Office, F.N.P.: Over 1 Million Potential Victims of Botnet Cyber Crime (2007)
4. Corrons, L.: Mariposa botnet (2010)
5. NATO/OTAN: Tackling new security challenges. Technical report (2011)
6. Lemos, R.: U. S. military to build botnets? **737** (2008)
7. Williamson, C.W.: Carpet bombing in cyberspace: Why America needs a military botnet

8. Trust, E.T.E.: Desactivando redes de ordenadores controlados por ciberdelincuentes para crear un internet ms seguro y fiable (2010)
9. Studer, R.: Economic and Technical Analysis of BotNets and Denial-of-Service Attacks. In: Communication Systems IV. University of Zurich, Department of Informatics (2011)
10. Cranton, T.: Cracking Down on Botnets. (2010)
11. Seiiler, J.: Entrance of Wikileaks Into Fourth Estate Creates Perils, Opportunities
12. Bloxham, A., Swinford, S.: WikiLeaks cyberwar: hackers planning revenge attack on Amazon
13. Zhuge, J., Holz, T., Han, X., Guo, J., Zou, W.: Characterizing the irc-based botnet phenomenon. In: Reihe Informatik, Pace University, White Plains, NY (2007)
14. Grizzard, J., Sharma, V., Nunnery, C., Kang, B., Dagon, D.: Peer-to-peer botnets: Overview and case study. In: Proceedings of the First USENIX Workshop on Hot Topics in Understanding Botnets. (2007)
15. Wang, P., Wu, L., Aslam, B., C. Zou, C.: An advanced hybrid peer-to-peer botnet. In: USENIX Workshop on Hot Topics in Understanding Botnets (HotBots07), 2007. (2007)
16. Wang, P., Wu, L., Aslam, B., C. Zou, C.: A systematic study on peer-to-peer botnets. In: Proceedings of 18th Internatonal Conference on Computer Communications and Networks, 2009. ICCCN 2009. (2009)
17. Naoumov, N., Ross, K.: Exploiting p2p systems for ddos attacks. (2009)
18. Nagaraja, S., Mittal, P., Hong, C.Y., Caesar, M., Borisov, N.: Botgrep: Finding p2p bots with structured graph analysis. (2010)
19. Binsalleeh, H., Ormerod, T., Boukhtouta, A., Sinha, P., Youssef, A., Debbabi, M., Wang, L.: On the analysis of the zeus botnet crimeware toolkit. In: Eighth Annual International Conference on Privacy Security and Trust (PST). (2010)
20. Seltzer, L.: Zeus Source Code Released
21. Ragan, S.: Overview: Inside the Zeus Trojans source code
22. Calles, J.A., Gonzàlez, P.: Troyano Flu b0.4 Windows. Manual de Usuario. (2011)
23. Nazario, J.: Twitter-based Botnet Command Channel. (2009)
24. Kartaltepe, E., Morales, J., Xu, S., Sandhu, R.: Social network-based botnet command-and-control: Emerging threats and countermeasures. In Zhou, J., Yung, M., eds.: Applied Cryptography and Network Security. Volume 6123 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2010) 511–528 10.1007/978-3-642-13708-2_30.
25. Spitzner, L.: The honeynet project: Trapping the hackers. IEEE Security & Privacy **1**(2) (2003) 15–23
26. Massi, J., Panda, S., Rajappa, G., Selvaraj, S., Swapana, R.: Botnet detection and mitigation. In: Student-Faculty Research Day, CSIS, Pace University, White Plains, NY (May 2010)
27. Goebel, J., Holz, T.: Rishi: Identify bot contaminated hosts by irc nickname evaluation. In: Proceedings of the USENIX Workshop on Hot Topics in Understanding Botnets (HotBots). (2007)
28. Xie, Y., Yu, F., Achan, K., Panigrahy, R., Hulten, G., Osipkov, I.: Spamming botnets: Signatures and characteristics. ACM SIGCOMM Computer Communication Review **38**(4) (2008) 171–182
29. Ormerod, T., Wang, L., Debbabi, M., Youssef, A., Binsalleeh, H., Boukhtouta, A., Sinh, P.: Defaming botnet toolkits: A bottom-up approach to mitigating the threat. In: eCrime Researchers Summit (eCrime). (2010)
30. Gostev, A.: Kaspersky Security Bulletin. Malware Evolution 2010. Technical report, Karspersky Labs (February 2011)