Challenges and Limitations in Current Botnet Detection

Felix Brezo, Igor Santos, Pablo G. Bringas and José Luis del Val
S3lab, DeustoTech Computing
Universidad de Deusto, Avenida de las Universidades 24, 48007 - Bilbao
{felix.brezo, isantos, pablo.garcia.bringas, joseluis.delval}@deusto.es

Abstract

Botnets are an emerging phenomenon that is becoming one of the most significant threats to security. Its danger lies less in the malicious codes themselves, but in the support they provide to implement a wide branch of very different criminal practices which are quite more compromising than harming an isolated computer, such as distributed denial of service attacks (DDoS), phishing, online fraud, dissemination of malware, building servers for exchange of illegal material or sending spam (bulk mail). Therefore, the scientific community together with the different business-related corporations and public entities, should be aware of the need of developing mechanisms to improve their detection, analysis and deactivation. And these measures should be taken as soon as possible to stop the dissemination of a threat which impact factor and flexibility in perpetrating attacks commanding an army of hijacked computers (bots), makes them a tool capable of compromising even the most complex information systems. Thus, this article sets out the main lines of current research in this field and proposing solutions to detect its existence through the analysis of the communication channels (via HTTP, P2P, IRC...) and the variations in the traffic detected, as well as their propagation mechanisms.

1 Introduction

The botnet phenomenon (resulting from the combination of English voices *software robot networks*) is not accidental nor particularly new. The development of complex connected computer networks and the computational possibilities which provides its management, have not been overlooked neither for the computer professionals nor for malware writers, spammers and other cybercriminals. Therefore, it is a a hot topic that has concerned security agencies in the world for a long time and has forced some security agencies as Europol to prepare their professionals to deal with the threats that these networks bring in to privacy, anonymity and corporate security. In parallel, several police operations have been carried out such as the “Operation Bot Roast” [16] in 2007, by which the FBI detected a botnet compounded by more than a million compromised computers; or the one carried out in February 2010, which allowed the Spanish *Guardia Civil* in cooperation with Panda Labs, the dismantling of the “Botnet Butterfly” with 12.7 million computers committing users and companies of up to 190 countries [4].

However, it is precisely because of its connection with the problem of spam, that the fight against these networks gets more importance, as practically all the spam sent worldwide, is mailed through machines under direct control of spam operators using botnets to accomplish such objectives acting as botmasters. In fact, although in 2004 it was estimated that almost 70% of spam sent on these dates came from various networks of hijacked computers or botnets, according to SpamHaus[1], amongst 300 and 400 spammers are responsible for themselves of the 80% of global traffic of such malicious content , figures that Symantec in mid-2009 taxed at 85%, reaching the 90.4% of the global traffic of emails worldwide in May of that year.

Nowadays bots, which represent about 25% of all the malicious connection attempts [19], 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 to the proliferation of a wide range of variations, modifications and mutations of bots based on the specific purpose for which they are sought.

This document is structured as follows. Section 2 shows the chronology of the various stages that we can find for the development of a botnet. Section 3 puts together the different architectures through which a botnet can be deployed.

[1] http://www.spamhaus.org/rokso Register Of Known Spam Operations (ROKSO) Database 2009

Section 4 analyses the characteristics of the communication and control channels for botnet management. Section 5 delves into the different approaches that exist for detecting and tracking these networks, together with the advance of some lines to work on in the future. Section 6 defines the specific characteristics that a modern botnet detection architecture should include to achieve its goal as efficiently as possible. Finally, Section 7 brings down the conclusions to be extracted after analysing the state of the art produced in this document.

2 Chronology of a botnet development

Going deeper as regards their construction, we can distinguish three main stages in the ripening period of a botnet before it is fully functional.

1. **Candidates selection**. This is normally done through shared binary in other P2P networks *parasiting* [27] the P2P protocol implemented for traditional file transfers. However, the size of the botnet will be limited to the number of users connected to that particular server, something that has lead to new ways of propagation ranging from email to instant messaging passing through other means file sharing.

2. **Infection**. Using Trojans, worms and/or viruses and often aided by social engineering, the attacker seeks to exploit the inexperience of the less aware users to recruit new members. To ensure the success of the infection, malware writers usually promote their botnets offering some high demanding files such as videogame crackfiles, code generators, movies, images, etc., which are truly infected files.

3. **Propagation of the botnet**. If the new infected machine does not belong to any P2P network yet, permitting the contact with the botmaster, the technique of the *bootstrap* used in traditional P2P applications is applied: a list of the locations of those members of the botnet to be accessed is either hardcoded (as does Trojan.Peacomm [7]) or provided in a configuration file. This approach's main counterpart is that if this list falls into the hands of defenders, the botnet's integrity and its expansion would be entirely compromised. As an alternative, dynamically updated lists are handled, being updated by combination between neighbours generating new lists of members to contact without compromising the whole architecture of the network.

3 Network architectures

The main strength of botnets lies in the potential of having a fluid network of connected computers which can be controlled remotely. To achieve this goal, there are very different visions about how to deal with the communication problems between the entities in the botnet, which is why there exist distinct architectures:

- **Centralized**. Usually consisting of a central node distributing messages between network clients. They are characterised by:
  - Low latency due to the small number of hops required to transmit the orders from the botmaster.
  - Direct connection to order distribution nodes, which would compromise the security of the network in case of accidental detection of a node.
  - Implemented using different communication protocols, but most typically the IRC and HTPP.

- **P2P (Peer-to-peer)**. 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, making use of the facilities of traditional P2P networks which allow a high connection and disconnection ratios. Each node has greater structural complexity because all of them can act as both, client and server, being more difficult to intercept and study. Some examples of P2P network architectures are the aforementioned Trojan.Peacomm and Stormnet, but note that the fact that of having a member-to-member architecture is not needed to be related with the way its communication system commands are scheduled, being possible to find cases of P2P-based IRC botnets as it will be detailed below.

- **Unstructured**. This architecture leads to an extreme the concept of P2P networks, as it tries to make invisible the mere existence of the communication network to other infected systems, even in their own botnet. The idea is that whenever the botmaster wants to convey some kind of command, it will encrypt them, waiting for any random successful scans performed via Internet with any other bot in the botnet without committing network's security in case of the interception of a peer. The price of these interesting features lies in the increasing complexity of the network, which management gets more difficult due to the bigger latency of the messages sent, being less efficient when trying to carry out certain attacks.

4 Command & Control channels

The communication between bots has undergone major changes since the use of IRC clients, often created on the fly by malware writers to reach higher levels of anonymity and independence of some more complex point to point connections. 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. It is precisely this mechanism of effective communication between bots, which determines both, the network topology and its ability to avoid detection and ensure its robustness, differing these applications from the rest of the set of malware by opening Command & Control (C&C) communication channels. Those C&C channels can implement one of the following philosophies: pull or push.

4.1 Pull philosophy

The pull philosophy, also known as *command publishing/subscribing*, because of the way they actively obtain the new lists of instructions to perform published by the botmaster and communicated through a subscription service. Its use, specially common in centralized networks, may vary depending on the protocol through which the C&C channel is implemented in the botnet:

- 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, which exposes the communication channel nodes as the access to this web-page is lots of times, public.

- In those based on IRC, bots are also connected periodically to a specified 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).

- In the ones based in file sharing P2P networks, when trying to find a file, a query is transferred through the network, receiving a positive response if a member of the same botnet contains it, initiating then the communication protocol.

Special mention must be the *parasitic botnets*, 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. This philosophy uses *in-band-messages* (common P2P traffic) permits the sending and reception of commands to be confused with legitimate traffic within the network, difficulting their detection and complicating the analysis of traffic as this approach sues a pre-filtering process.

4.2 Push philosophy

Also known as *passive listening*, this philosophy waits until the nodes receive the instructions to be executed and then transmit them to the other members of the network, avoiding communication through regular connections with the central server and reducing the possibilities detection by traffic modelling output from the infected computer. The decision of what items will be informed of the new commands to execute can be simplified by trying to find a particular file (owned by all the members of the botnet), sending only to those members who owned the above file the appropriate directives, as they would have been identified as pals within the botnet. However, not having generated the lists of possible targets, the bot has to get in touch with their peers by opening a new channel that can only be decoded by the computer that is trying to communicate, opening a new line of study to detect the existence of a botnet through the analysis of these parallel connections.

5 Countermeasures

At the time of detection and tracking botnets, there are two approaches that implement two different techniques: the active one, usually based on *honeypots*[2] as the one developed in *The Honeynet Project* [23] and performing a passive monitoring network traffic.

Thus, the active method tries to detect and then get access to a botnet simulating the operation of a bot. However, given the nature of the connection, this behaviour can be detected by botnet operators who may take further measures to prevent future incursions.

In contrast, the passive pathway is based on the study of information flows in the network environment, without establishing direct communication links with the members of the botnet and observing the network traffic that reaches a *darknet* [21]. Although the term *darknet* is commonly used to define a series of networks and technologies that enable users to copy and share digital content without being possible to know certain details of what downloads have been made or the origin of the users who perpetrated it, Seewald et al. [21] in his publication used it to define a range of IPs that are not available for real machines and that in case of being accessed may correspond to malicious connections.

The main advantage of the purely passive implementation of this approach is that its existence cannot be detected by the botmaster as no communication with the members of

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

the botnet is performed, what could make them useful for the study of botnets as well as for the establishment of warning and prevention mechanisms in case of detecting contamination of a network. However, it was found that slightly more flexible approaches like ACK packet forwarding to accept reaching connections [1], significantly increases the volume of information to analyse while so it does the risk of being detected by the botmaster.

In this sense some of the main possible outlines of future work in the detection of botnets are determined by the detection of certain symptoms which are common to most of the infections.

- **Detection of DNS traffic** can be a good starting point when analysing the connection attempts of each entity of the botnet to the network itself, but it will only be effective if the communication C&C channel is known in beforehand, working on the same line as it has been done to detect intrusions in, for instance, corporate and public networks.
- At **local network** levels, it is interesting the placing of honeypots as it is quite common to find attempts of infection in those machines belonging to the same LAN where the hijacked bot is operating.
- **Malfunction of the machine itself** can also be some of the main symptoms of an infection, especially if the appropriate measures have not been taken. Some of the symptoms may include the following:
  - **Unusual or unknown processes** running on *background*.
  - **Unusually slow Internet connection** as a symptom of participating in a DDoS[3] or sending bulk mail or spam.
  - **Strange browser behaviour** such as changing the homepage, the appearance of pop-ups or the inclusion of non-installed tool-bars by the user.
  - Addition of **unknown files** to the list of programs allowed to access the internet or to run on system start-up.
  - **Unknown network connections** established to or from the computer.

5.1 Detection of botnets: outline of work

Once differentiated the two main approaches of analysis of botnets, in this section we proceed to enumerate some of the documented detection methods already used in the past, as well as its limitations and possible areas for expansion of knowledge. These methods include both, a more traditional approach such as detection by means of signatures and one more focused on the analysis of traffic based on the characteristics of the information sent and received.

5.2 Detection by means of signatures

Effective and still in use, detection using signatures is currently facing the challenges posed by modern techniques capable of avoiding polymorphic systems. However, passive monitoring has still shown some successful stories:

- Goebel et al. [6] have managed to combine lists of regular expressions in IRC nicknames (previously labelled as suspects) along with an analysis of n-grams to study whether a particular conversation made through unusual channels of communication belongs in a compromised computer. However, despite the problems related to those new generations of *applets* that connect to IRC clients using automatically generated names with specific patterns that could add some extra false positives, its main limitation is that it is strictly necessary to label these nicks (or at least a part of them) as likely to play malicious behaviour.
- Similarly, Blinkey et al. [17] contrasted a list of IPs of a specific IRC channel, with those IPs who were permanently scanning so as to identify malicious behaviour and to isolate these connection attempts.

5.3 Detection of cooperative behaviour

Typically, studies going on this path are oriented towards the pursuit of activities relating the infected machines, with the ultimate goal of using this information to optimize the tracking techniques. There are two types of traffic to detect: those responses to requests from the botmaster (with partial or total results, including the current status of the operation) and the traffic used to perform the task requested by the botmaster itself, understood as purely malicious activities such as sending spam, massive connections to certain servers in a DDoS, etc. The idea is to make statistical attacks seeking behaviour profiles that differ from that performed by a non-infected standard user, so as to select those network nodes more likely to be part of a botnet. The ultimate goal is to detect (1) botnets based on the push philosophy by identifying large numbers of connection attempts result of random scans and (2) computers responsible for a large number of connections that may correspond to nodes from which centrally managing network. Some examples are:

- Analysis of the information flow through certain IRC ports [18] depending on the particular characteristics of the implementation of the botnet. In this regard, it

[3] DDoS/Distributed Denial of Service attack refers to a Distributed Denial of Service attacks

was reported that in some ports (6667, associated to the control of botnets infected by SDBot) almost 35% of the traffic routed through it was caused by the botnet [3] so as to carry out its activities.

- Detection of recognition lookups by botmasters to determine the current status of the *blacklist*[4] of members.
- Calculation of the distance between the monitored data stream in suspicious machines and a predefined IRC communication model by analysing the traffic in the transport layer [10] with the already mentioned advantages of performing it from an entirely passive point of view (making invisible the analysis for the botmaster). This approach has succeeded to detect botnets that use encrypted communications with a false positive ratio less than 2%. However, there are still no studies confirming the efficiency of these systems with botnets controllers using HTTP or peer-to-peer CC channels.
- Classification of the IRC chats as human-made or bot-generated communication by using tools such as, for example, recording the fact of sending periodic queries or by disproportionate amount of information managed, taking into account those considerations highlighted by Dewes et al. [5] that in a standard IRC chat conversation the reader normally receives ten times more data he sends. Thus, by observing and profiling, abnormal talks could be detected allowing further study based solely on the volume of information flowing between two nodes.
- Gu et al. [8] made an interesting analysis of how to detect anomalous network traffic based on HTTP and/or IRC trying to find patterns of space-time similarities in the communications amongst the members of a botnet based on the preprogrammed nature of the bots. Those messages to be identified are those of request for system information, scanning the network and trying to identify machines performing communication of similar characteristics, so that if a number of machines running commands of identical characteristics are identified, we could say that a new botnet would have been detected. The most interesting feature in this senses resides in the fact of not needing to identify the content of the conversation evading even the use of certain encryption techniques, as it is based on the comparison of frequencies and temporal gaps by using Kolgomorov's complexity as proposed by Wehner [28] for the detection of worms.

This philosophy changes based on the methodology (push/pull) used by the botnet to be examined. There exist some case studies in HTPP and IRC (such as BotSniffer implemented as a plug-in for Airsnort [8]) as well as listings of the key commands that implement some bots such as Agobot, SpyBot or SDBot. However, even if these commands were coded to hide the real content of the messages, the activity of the bots can still be detected because of the need for these to answer to the requests made by the botmaster, a communication which is always done in a much more consistent and standardized way than the traditional human communications. That is why botnets taking into account current techniques of randomized pooling seeking to avoid (or at least confuse) these systems by modifying the time intervals between the interactions.

5.4 Offensive behaviour detection

This approach, more oriented to the detection of malicious activity on their own, assumes that botnets send massive amounts of information (spam, ghost connection attempts...) in relatively short periods of time. Thus, Xie et al. [30] uses the volume of data sent during these periods along with the information obtained from spam servers for tracking such contents before they cause real damage (for example, sending massive spam) and preventing the consequences by identifying the nodes that are responsible for carrying out this task.

5.5 Deactivation techniques

Once detected the existence of a botnet, for terminating its activities there are different outlines of work:

- Closing them physically by placing them in quarantine at an early stage of development to complete the process of disinfection of compromised nodes one by one.
- Closing the communication C&C channels to prevent them from spreading even more commands and, thereby, limiting its operational capabilities first and, then, their propagation ways.
- Using Index Poisoning Attack techniques, usually used by companies to prevent the redistribution of software, video and other content protected by copyright by including false records [13, 15]. Applying this to the realm of botnets, implies detecting those files that are searched by infected machines (using honeypots for example) so as to distribute afterwards files under the same key names with no information of any kind, making it more difficult for the nodes to communicate effectively with their pals [7]. However, there is literature capable of dodging this issue [25] which could isolate botnet attacks by preventing the spread of identity (and location) of its nodes through the use of a

[4] A *blacklist* is a registration of persons, entities or commands so as to deny them their privileges

standard public key model, alerting the scientific community about the problems that would involve an increase in the complexity of traditional communication channels in the next generation of botnets.

- Sybil attack techniques try to infiltrate a node in the list of bots to sabotage the communications from inside by modifying (and subsequent publication) of hashes, files or commands [26] with the advantage that changing some bits is sufficient enough to block the spread of the commands on the network as absolutely different files would be looked for.
- Commands or members *blacklisting* detected at earlier stages on suspicious communications.
- Hijacking of the network by sending commands to deactivate it. The downside is that many botnets currently use asymmetric encryption systems [24] to prevent the detection of one of the nodes that would compromise the security of the rest of the network.

6 Experimental morphology

The main ways of experimentation in this field must include the release of some versions of bots as (Rbot, Spybot or SdBot) whose source code has been previously modified with the aim of: (1) scan the traffic and connections in a controlled environment (usually consisting of virtual machines) to later (2) test the software in a real environment after the corresponding period of evidence collection, analysis and study of the exchange of information intercepted. In this regard, we recommend include the following in a reliable system for botnet detection:

- **Content filtering**. Given the large amount of traffic that is handled nowadays, performing filtering operations is essential to focus the analysis on those communications with more overtones of being potentially suspicious. In this regard, such filtering can be done in two ways:
  - Protocol filtering, discarding those protocols that are not C&C communications-oriented such as UDP and ICMP.
  - Inclusion of *soft whitelists* which dynamically ensure the reliability of the source of the communication, marking as safe the traffic coming from those nodes.
- **Review of handshaking**. It tries to identify the origin of the IRC, HTTP and P2P communications by detecting their handshaking routines. It takes into account the special features of these protocols when establishing the connections and negotiating acceptable parameters for equipment and systems at both ends of the communication channel, including data transfer rates, alphabet coding, parity, protocols, hardware features, etc.
- **Scan semantic messages**. Scanning private messages and post topics in order to detect sending commands, keeping in mind at this point the involved legal issues related to privacy.
- **Detection of unusual commands**. Although this should not be used by itself as a single detecting element, in certain environments it may be of interest to detect traffic that uses non-standard protocols in a standard user (such as MX DNS or SMTP) and, thus, labelling it as suspicious traffic in order to filter the content for later analysis.
- **Grouping clients**. The idea is that, by having a server with large number of client connections, the chances of coming across traffic from a botnet also increase.
- **Analysis of n-grams and n-sequences**. With the success of these systems in other fields of malware detection [11, 20], it would be interesting for any botnet detection system based on content or user commands coming from such networks.

7 Conclusion

With recent attacks on several credit card servers as a retaliatory measure by the constraints brought by these companies to Wikileaks [22, 2], it has been shown that even in 2011 the main channel of transmission of commands between nodes in a network of infected computers remain the IRC chat channel (used for the transmission of orders in the attacks by *Anonymous* [14, 9]).

That is why the main monitoring techniques must continue to focus their efforts on the development of interfaces on IRC chat channels. However, some extra layers should be taken into account in the design of a reliable botnet detection system, to fight the proliferation of other C&C techniques such as chat (or private messages) in the P2P servers or static HTML pages connections (like one used by the open-source Prablinha project[5]).

In this regard, the main paths of experimentation in this field must include the release of some versions of bots (Rbot, Spybot or SdBot) which source code has been previously modified with the aim of: scanning the traffic and

[5] Prablinha is a project trying to teach some details related to botnet configuration and deployment using .NET frameworks. Different versions have been published exploiting IRC and HTTP/HTTPS channels. More information can be found in http://itsm3.com/Aplicaciones/prablinha

connections in a controlled environment (usually consisting of virtual machines) to, later, testing the software in a real environment after the corresponding period of evidence collection, analysis and study of the exchange of information intercepted has been completed.

In the face of such an alarming growth of malicious applications which numbers have been beating every year, and given the direct threat of hijacking computers to personal, corporate and governmental security, public collaboration is needed by the scientific community to keep on carrying out more researching work in this field. Botnets will be used in the future as tools to gather information about military [12] and economical targets as suggested by Colonel Charles W. Williamson III [29], leading to the need of improving current analysis and deactivation techniques.

Although being dealing with a field whose expertise is still under development, we have proposed in this paper some outlines of work to start 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] M. Bailey, E. Cooke, F. Jahanian, J. Nazario, D. Watson, et al. The Internet Motion Sensor: A distributed blackhole monitoring system. In *Proceedings of the 12th ISOC Symposium on Network and Distributed Systems Security (SNDSS)*, pages 167–179. Citeseer, 2005.

[2] A. Bloxham and S. Swinford. WikiLeaks cyberwar: hackers planning revenge attack on Amazon.

[3] C. Borghello. Botnets, redes organizadas para el crimen. 2007.

[4] L. Corrons. Mariposa botnet, 2010.

[5] C. Dewes, A. Wichmann, and A. Feldmann. An analysis of internet chat systems. In *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement (ICM)*, pages 51–64, New York, NY, USA, 2003. ACM.

[6] J. Goebel and T. Holz. Rishi: Identify bot contaminated hosts by irc nickname evaluation. In *Proceedings of the USENIX Workshop on Hot Topics in Understanding Botnets (HotBots)*, 2007.

[7] J. Grizzard, V. Sharma, C. Nunnery, B. Kang, and D. Dagon. Peer-to-peer botnets: Overview and case study. In *Proceedings of the First USENIX Workshop on Hot Topics in Understanding Botnets*, 2007.

[8] G. Gu, J. Zhang, and W. Lee. BotSniffer: Detecting botnet command and control channels in network traffic. In *Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS08)*. Citeseer, 2008.

[9] InfoSecurity. Anonymus hacking group uses IRC channles to co-ordinate DDoS attacks. 2011.

[10] A. Karasaridis, B. Rexroad, and D. Hoeflin. Wide-scale botnet detection and characterization. In *Proceedings of the USENIX Workshop on Hot Topics in Understanding Botnets (HotBots)*, 2007.

[11] J. Kolter and M. Maloof. Learning to detect malicious executables in the wild. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 470–478. ACM, 2004.

[12] R. Lemos. U. S. military to build botnets? 737, 2008.

[13] J. Liang, N. Naoumov, and K. Ross. The index poisoning attack in p2p file sharing systems. In *IEEE INFOCOM*, volume 6. Citeseer, 2006.

[14] K. Lillington. Time to talk: Anonymus speaks outs.

[15] X. Lou and K. Hwang. Prevention of index-poisoning DDoS attacks in peer-to-peer file-sharing networks. *submitted to IEEE Trans. on Multimedia, Special Issue on Content Storage and Delivery in P2P Networks*, 2006.

[16] F. N. P. Office. Over 1 Million Potential Victims of Botnet Cyber Crime, 2007.

[17] S. Racine. Analysis of internet relay chat usage by ddos zombies. *Master's thesis, Swiss Federal Institute of Technology Zurich*, 2004.

[18] S. Racine, T. Dubendorfer, and B. Plattner. Analysis of Internet Relay Chat Usage by DDoS Zombies. 2004.

[19] A. Rajab et al. A multifaceted approach to understanding the botnet phenomenon. In *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, page 52. ACM, 2006.

[20] I. Santos, F. Brezo, J. Nieves, Y. Penya, B. Sanz, C. Laorden, and P. Bringas. Idea: Opcode-Sequence-Based Malware Detection. *Engineering Secure Software and Systems*, pages 35–43.

[21] A. Seewald and W. Gansterer. On the detection and identification of botnets. *Computers & Security*, 2009.

[22] J. Seiiler. Entrance of Wikileaks Into Fourth Estate Creates Perils, Opportunities, Benkler Says.

[23] L. Spitzner. The honeynet project: Trapping the hackers. *IEEE Security & Privacy*, 1(2):15–23, 2003.

[24] S. Staniford, V. Parxson, and N. Weaver. How to own the internet in your spare time. In *Proceedings of the 11th USENIX Security Symposium*, 2002.

[25] G. Starnberger, C. Kruegel, and E. Kirda. Overbot: a botnet protocol based on Kademlia. In *Proceedings of the 4th international conference on Security and privacy in communication netowrks*, page 13. ACM, 2008.

[26] R. Vogt, J. Aycock, and M. Jacobson. Army of botnets. In *Proceedings of the 2007 Network and Distr. System Sec. Symposium (NDSS 2007)*, pages 111–123. Citeseer, 2007.

[27] P. Wang, L. Wu, B. Aslam, and C. C. Zou. An advanced hybrid peer-to-peer botnet. In *USENIX Workshop on Hot Topics in Understanding Botnets (HotBots07), 2007*, 2007.

[28] S. Wehner. Analyzing worms and network traffic using compression. *Journal of Computer Sec.*, pages 303–320, 2007.

[29] C. W. Williamson. Carpet bombing in cyberspace: Why America needs a military botnet.

[30] Y. Xie, F. Yu, K. Achan, R. Panigrahy, G. Hulten, and I. Osipkov. Spamming botnets: Signatures and characteristics. *ACM SIGCOMM Computer Communication Review*, 38(4):171–182, 2008.