Communication Structures of Botnets with Case Studies

Nicholas Cornhill
University of Minnesota, Morris
600 East 4th Street
Morris, MN
cornh044@morris.umn.edu

ABSTRACT

Botnets are a serious security threat to both individuals and organizations that use the internet. Botnets have been implicated in the theft of personal information such as email names and passwords, account information, and bank information, as well as other illegal activities such as distributed denial of service (DDoS) attacks which can shut down internet services causing a loss in revenue. One common way to detect a botnet is to to perform graph analysis on a graph of the communication network to find a pattern that is characteristic of botnets.

For this graph analysis to occur, first it is necessary to understand how a botnet communicates internally. Detection is an important aspect in defending from attacks as well as shutting down the botnet. In this paper, I will discuss the two most common communication structures used by botnets: command and control, and peer-to-peer. I will also compare the trade offs of either system. I will finish by describing the communication structure of two botnets: the Miner and the Waledac botnets and discuss how they utilize aspects of both earlier structures.

Categories and Subject Descriptors

D.7.8 [**Networks**]: Network Types—*Peer-to-Peer* ; I.4 [**Security and Privacy**]: Intrusion/anomaly detection and malware mitigation

General Terms

Security

Keywords

Botnet, Communication Structures, Peer-to-Peer, Command and Control

1. INTRODUCTION

Botnets are networks of computers that have been infected with malicious software, or malware, called bots. Botnets allow the person who deployed the bot malware, called the botmaster, to issue commands to the computers on the botnet without its owners permission. These individual bots are organized by the botmaster in different ways to allow them to control the botnet as a single entity rather than controlling each one individually. The different ways the network can be set up are called communication structures.

These botnets are then used in several ways, including stealing private information from the user such as credit card, and social security numbers, as well as usernames and passwords for various accounts including social network accounts, email accounts, and even bank accounts. Botnets can also be used to perform tasks that either require large amounts of bandwidth or computing power, such as sending spam, distributed denial of service (DDoS) attacks, and in the case of the Miner botnet, mining bitcoins, which are a form of currency that has developed online.

This paper will focus on some of the communication structures botnets employ as well as discuss the communication structure employed by the Miner botnet, a recent botnet based in Germany and Russia which at its peak had at least 29,000 members, and the Waledac botnet which was a large botnet of around 100,000 bots that was publicly taken down by Microsoft. The next section, section 2 will introduce the basics of communication structures as well as their role in detecting botnets. Then I will explain the lifecycle of botnets that use the previously discussed communication structures. The third section will be a discussion of the lifecycle of the Miner botnet and the fourth will be a discussion of the lifecycle of the Waledac botnet. Then I will summarize the important points of the paper.

2. COMMUNICATION STRUCTURES

Communication structures dictate how the individual bots in the network communicate. This effects how the botmaster controls the botnet and how the security community can detect the botnet. Botmasters want their botnet to be easy to control and to respond quickly to their commands. To achieve that goal, the communication structure of botnets have become increasingly complex.

These communication structures however make patterns that can be used to detect the botnet. One of the most common ways this detection occurs is by using what are called *structured overlay topologies* [6]. If one was to graph a botnet where each bot was a node and a connection between two nodes occurred if two bots could communicate directly, then the overlay topology would be how that graph was structured. For example, a ring overlay would be one where each node of the graph has a two way connection with two other nodes.

The advantage of using a graph based technique to detect botnets is that because the only information that is used is *if* communication occurs between two nodes not *what* communication occurs. Encryption or other forms of hiding what data is passing between two computers do not affect the analysis [3, 6]. One disadvantage of this technique is that it can sometimes pick out P2P or other networks that are

not malicious but fit the overlay. Therefore, this detection technique must be paired with one that can verify that a computer is infected with a bot [6]. However, to get the overlays, we need to study the botnets themselves to find how they communicate.

There are two main kinds of communication structures that botnets use. The first is Command and Control (C&C), the second is Peer-to-Peer (P2P). C&C is a centralized structure where the botmaster sends commands, updates, or receives data from the botnet through one or more centralized locations [1]. A C&C structure allows information to move between the botnet and the botmaster quickly [1], but has the disadvantage that if the centralized location is removed or blocked the entire botnet is stopped [2].

The second kind of communication structure is a P2P structure, which is a decentralized network. Each bot in the botnet has a number of other bots it knows about and can interact with. Information is sent from the botmaster to a number of bots in the botnet, who then pass the message on to bots of which they are aware.

This has the advantage that if a number of bots are removed from the network the botnet still can operate. However, it is both difficult to maintain a P2P botnet as the communication structure is more complex and the latency of network is often very high with no guarantee of a message reaching all of the bots [2, 5]. For example, when the botmaster of the Waledac botnet attempted to update the botnet more than 30% of the botnet was not updated after two weeks [11]. However, it also makes the botmaster harder to detect and stop, as the botmaster does not need to remain at a fixed location. Instead, the botmaster can join and leave the network at any time and give instructions to any bot, which will then relay the message to the rest.

2.1 Lifecycle of a C&C Botnet

The first step in the lifecycle of a C&C botnet is to infect a vulnerable machine and get it to execute some malicious code, called the shellcode. Infecting a computer with shellcode is no different than infecting a computer with any other sort of malware, so it can be done in any number of ways, such as drive-by-download attacks where malicious software, often JavaScript, is run when browsing a webpage [4], or social networking services can be used [9]. Other common methods include worms, viruses, and trojans [2]. The shellcode commands the machine to contact one of the servers the botmaster had set up previously and download the actual bot code, which is set to run every time the machine is booted [1].

Once the bot code is downloaded, the bot is fully operational. It contacts the central communication point or points, which can be one or more IRC chatrooms [12], or websites, where it can receive commands and relay information back to the botmaster.

2.2 Lifecycle of a Peer-to-Peer Botnet

Similar to the C&C botnets, the first step is infection, which can occur via all the infection techniques a C&C botnet can use. However, due to the fact P2P botnets often piggyback on legitimate P2P networks, it is easier for them to recruit computers in the existing P2P network [14].

Once infected, the new bot must locate other individuals in the botnet so it can receive commands. The bot records bots of whom it is aware of in its *peer list*. How the peer

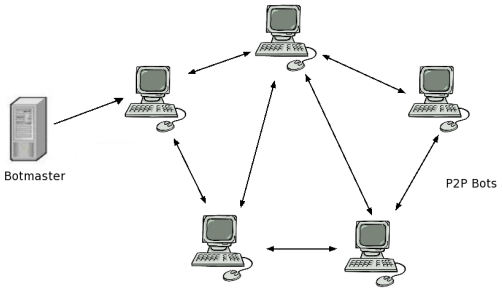

Figure 1: An illustration of a P2P botnet.

list is populated is specific to the botnet. If the botnet is piggybacking on a pre-existing P2P network, then it can simply use that network to find and/or communicate with other bots [14]. Otherwise, a common technique is for the bot to gather an initial list of bots from either the bot that infected it, a hardcoded list of bots, or a list of bots stored in some location on the internet. Once the initial list is populated, the new bot can use that list to find other bots in the network to diversify who it knows. This process is commonly referred to as *bootstrapping* [14]. Figure 1 shows an example P2P botnet once the bootstrapping procedure is completed.

Once the peer list is populated the bot stands by for orders. The bot can either wait passively for orders issued by the botmaster and pass them on to the bots in its peer list, which is called pushing. Or it can periodically ask bots in its peer list for any commands they have, which is called pulling. Which way the bot receives commands is specific to each botnet. However, there are several issues to consider. If a botnet pushes, there tends to be detectable spikes in network activity when commands are issued. However, if a botnet pulls, then there will be regular requests for commands being sent out from each bot which is another pattern that can be spotted. Overall, the decision is a trade-off that depends on the specific needs of the botnet.

2.3 Botnet Detection

One of the main ways one can detect a botnet is by finding patterns in internet traffic caused by the botnet using an overlay. The communication structure plays a large role in which patterns a botnet generates and thus, which overlays will be effective. Command and control networks tend to have large numbers of communications being sent to and from the C&C servers, as shown in Figure 2. This is the main weakness of a C&C botnet. Not only is the network traffic very obvious to observers but it clearly identifies most if not all of the bots in the botnet. More critically it reveals the C&C servers, which are the weak point in the network. Once the C&C servers are located it is possible to coordinate with law enforcement or the DNS servers to either stop the centralized servers or block traffic to and from them [16].

Because the C&C servers are the weak point in a C&C botnet many techniques have been developed to either hide the servers or make them more difficult to take down. Encryption has been used to ensure that the bots in both C&C and P2P networks only respond to commands made by the botmaster and not an impostor. To avoid having their IP

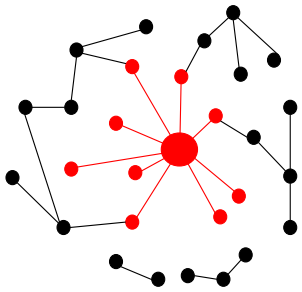

Figure 2: An example of what a C&C botnet might look like in a network. The small red nodes are nodes of the botnet, while the large one is the C&C server. Spotting the botnet is easy because the server node has many more connections than the surrounding nodes.

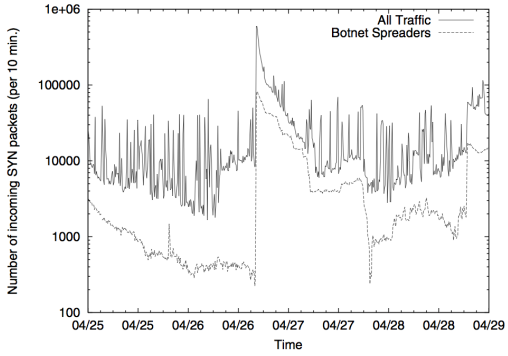

Figure 3: This is a graph of network activity on a network partially infected with a botnet. The spike about three quarters of the way through 4/26 is when the botmaster issued a command to search for new computers to infect. Image taken from [1]

addresses blocked, botmasters changed from hardcoding IP addresses into their botnet and instead used domain names to avoid losing control of the botnet in case their IP address was blacklisted.

In reaction to this, security organizations began to block domain names, which caused botmasters to employ a technique called *domain flux*. [12] A botnet that uses domain flux has an algorithm that generates a series of domains for the bot to contact for information, so that if any given domain name is blocked, the botnet will be able to contact the botmaster at the next domain name. Additionally, the botnet changes what domain they contact periodically to break the communication pattern that forms with that domain. Stone-Gross, *et. al* [12] managed to overcome domain flux by reverse engineering the bot code from a binary they found and predicting the domain names the botnet would use, then purchasing them before the botmaster could. They then set up their own C&C server at that domain and took control of the botnet.

Peer-to-Peer botnets were developed mainly to fix many of the issues C&C botnets had with detection via internet traffic. Because they are distributed, there is less traffic to or from any given point in the network. When a botmaster issues a command the information initially only travels to a few bots which then relay that command to the bots they know of, and those bots do the same. This causes their overlays to generally be more complicated than that of a C&C botnet although the use of overlays is still possible. However, other detection techniques can reveal these networks as well. A common pattern caused by P2P botnets is a cascade of packets passed through the botnet at once as shown in Figure 3. Often, when a botmaster issues a command, that command will be processed by the initial recipients and then passed on to everyone in their peer list immediately. Because each bot is almost always in multiple peer lists to support network robustness, when a botmaster issues a command more packets are sent within the network in total than with a C&C botnet which causes large recognizable spikes of activity within the network.

One way that these network spikes can be mitigated is to hide the traffic in pre-existing P2P networks. In what [14] called a *parasitic* botnet, all of the bots were members of the network before they were infected. This means that any traffic within the botnet can simply appear to be continued membership of the original P2P network. However, this limits the number of possible bots to be a subset of the entire network that are vulnerable to infection. The other kind of P2P network is what [14] called a *leeching* P2P botnet, where the bots join a pre-existing P2P network and depend upon it for its communication channels. This has the advantage of not limiting the potential members of the botnet to the degree the parasitic P2P network did. However, if the botnet becomes large the number of members in the P2P network could skyrocket which may alert members of the internet security community to its existence.

3. THE MINER BOTNET

3.1 Background for Miner Botnet

The Miner botnet was a botnet whose first appearance can traced to December 20th, 2010 and died by February 2012. It received large amounts of media attention in August and September, 2011, when it attacked a number of German and Russian websites. The botnet was initially used for pay-for-install malware, where the botmaster would be paid money to install malware on the infected machines. Later, the botnet was updated to block access to Russian social networking sites VKontakte.ru and Odnoklassniki.ru. At around this time, it was also updated with the ability to perform distributed denial of service attacks. It was not until sometime around late May 2011 that the Miner botnet was updated to mine bitcoins, functionality for which it was later named [9].

3.2 Bitcoins

Bitcoins are a digital currency initially proposed by Satoshi Nakamoto in 2008. The Miner botnet was one of the first botnets to mine bitcoins. Bitcoins have no central authority to grant them value or validate transactions. Instead, their value is determined by what value people give the coins. Transactions are verified using a proof-of-work system in which a certain amount of calculations must be performed by a network of nodes to verify the transaction. The specifics are beyond the scope of this paper, but can be found in the original paper authored by Satoshi Nakamoto [7]. A similar proof-of-work system is used to generate coins. A certain

amount of processing power must be expended to generate a coin. As the number of coins generated increases, exponentially more processing power must be expended to generate more coins.

3.3 Lifecycle of the Miner Botnet

There is evidence of the existence of the Miner botnet starting in December, 2010, and it survived until February, 2012 [8]. Evidence suggests that the Miner botnet was initially completely command and control oriented and was later altered in July 2011 to function as a peer-to-peer network with some aspects of the old C&C remaining. Although there is evidence of the Miner botnet existing as a C&C network there are no binaries of the bot from that time nor analysis of that version of the botnet. I will discuss how network version 1999 operated, which was the most current version of Miner as of September 12, 2011. However, the only known infection technique it used has been attacks through Facebook and VKontakte. Miner would use stolen accounts on the social networks to send messages with a link to a fake YouTube video. When the victims attempted to watch the video it would prompt them to install a new version of the Adobe Flash plugin, but instead would install the shellcode for the botnet. This malware would then copy itself into either the root directory or one of its subfolders and restarts itself as a service to be executed on startup.

As previously mentioned, the Miner botnet was P2P network with aspects of a C&C structure as of version 1999. The network can be divided into four tiers as shown in figure 4. These four tiers are sorted by relevance to the botmaster where tier one is most relevant and tier four is least.

Tier one is the master servers controlled directly by the botmaster as well as a number of proxy servers used to hide the master servers. The proxy servers are reachable through both hard-coded IP addresses and domain names. The proxy servers then forward any data they receive to the master servers. In turn, the master servers communicate through the proxy servers to the next tier down, the distribution servers.

The distribution servers are the main aspect of the C&C network that remains. They manage the lower tiers on a broad scale, keeping track of infected computers, managing network connectivity by assigning peer lists to the bots, and distributing commands and updates to part or all of the botnet. Like the master and proxy servers, the distribution servers are maintained by the botmaster, but are contacted through the first tier.

The third tier of the botnet is the P2P bots, which are bots that have a direct connection to the internet. These P2P bots both serve as workers to mine bitcoins, execute DDoS attacks, or perform any other action the botmaster commands, as well as redistributing commands or updates they receive from the distribution server. This layer can also perform some network maintenance on itself as well as distribute commands and updates issued directly from the botmaster in case the upper tiers are removed. This tier occasionally contacts the master server to obtain the location of any distribution servers that have been added or moved.

The last tier consists of bots which are not directly connected to the internet, most likely because they are on a private network. They act solely as workers, but not as redistributors. However, as these machines are not directly

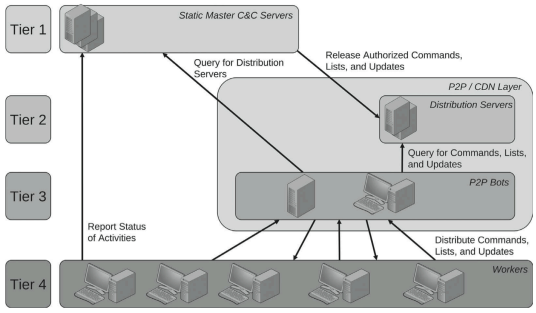

Figure 4: The communication structure of the Miner botnet, image from [9]

connected to the internet, they are also more difficult to track [11].

The communication structure the Miner botnet uses can neither be classified as a pure C&C or P2P structure. It has a number of servers maintained by the botmaster in the first two tiers. These are useful because it gives bots a guaranteed location they can contact to receive updates to the bot and are used to add some stability to the botnet because they are nodes that are consistently contactable and generally do not change locations. Meanwhile, those servers are not required for the botnet to operate. The second tier can be completely removed and the botnet can still operate, using the third tier instead. This gives the botnet a measure of robustness. If a member of the security community attempted to take down the botnet by blocking the second tier's IP address with the domain name server then the botnet would still function allowing the botmaster time to connect a new IP address to that domain name. Even if everything is up and working, this structure has the advantage of stealth over a C&C based botnet. Bots have the option to contact either one of the servers in the second tier or a bot in the third tier. This means communication is distributed between multiple entities instead of solely the C&C servers, making the botnet harder to detect.

3.4 Death of the Miner Botnet

The Miner botnet died due to neglect. The last update issued to it occurred on December 2011 and the botmasters performed no new maintenance after that point. The P2P layer was no longer traversable because each bot did not know of any or enough other bots to move through the network around late February 2012. This occurred due to network churn. Network churn is just the natural changing of IP addresses and how computers are connected to the network that occurs over time due to various reasons. Normally network churn would not be a problem, if a bot started losing connectivity to neighbors it could ask the distribution servers for a new peer list. As the distribution servers were not active, the peer lists could not be repopulated and the nodes of the network lost connectivity [9, 8].

4. THE WALEDAC BOTNET

4.1 History of the Waledac Botnet

The Waledac Botnet was originally released in December, 2007 [10], and was taken down in 2010 by Microsoft [15]. It

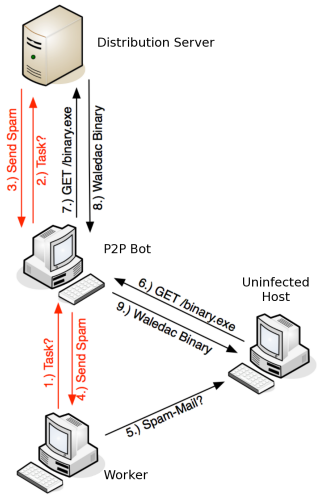

Figure 5: How Waledac infects a new host. Arrows depict communication and are numbered according to when the communication occurs. Image taken from [11]

can be seen as a predecessor to the more well known Storm Botnet in terms of communication structure although it can be argued that Waledac is less centralized than the Storm botnet. Although it is technically considered a P2P network it does have aspects of a C&C style botnet and functions similarly to the Miner botnet.

4.2 Waledac's Communication Structure

Waledac's communication structure is essentially the same as the Miner botnet's although they differ in the specifics. At the topmost layer is the master C&C server. It is from there that the botmaster controls the botnet. Below that, there are a number of distribution servers which act as stable servers that can be used to aid in the distribution of binaries and updates. Then, below those are two more layers which are the bots. The upper level of bots, the P2P bots, are infected hosts that are accessible from the internet and act as part workers, part distribution servers. Then, at the bottom are the infected hosts that cannot be accessed by the internet, or the workers.

4.3 The Lifecycle of the Waledac Botnet

4.3.1 Infection and Bootstrapping Procedures

The Waledac botnet is mostly a P2P network with aspects of a C&C botnet. It functions very similarly to the Miner botnet in its partial use of C&C servers. The process of infection is depicted in Figure 5. The only infection vector Waledac had was via spam. Steps 1 through 4 is the worker asking what it should do. Step 5 is where it sends the spam which would include a link to another website. That link would go to the shellcode for Waledac which would then be downloaded and run by the host. Running the shellcode causes step 6 to occur, which is the host computer requesting the actual binary for the Waledac bot. Steps 6 and 7 are that request for the binary being passed up to the distribution servers and steps 8 and 9 are that binary being sent to the host. Once it has downloaded and installed the binary, it is a bot [11].

The bot code then instructs the computer to determine if it is a worker or P2P bot by checking its IP address. If it is on a private network it proceeds to mine personal information from the computer and send that up to the P2P bot above it. Then it waits for further commands from the botmaster.

If the host computer is not on a private network, it becomes a P2P bot and its functionality is different. A P2P bot mines personal data like the worker did, but in addition it needs to connect with the rest of the botnet. It populates its initial peerlist from one hardcoded into the binary or one located at a location specified in the binary. Once it populates its peer list initially, it requests lists from those bots and uses the new peerlists that are sent to it to repopulate its own. Then it periodically repeats the operation to ensure it has a fresh peerlist.

4.3.2 Death of the Waledac Botnet

In February 2010, Microsoft, as well as several other organizations including Shadowserver, University of Mannheim [11], University of Bonn, the University of Washington, Symantec, and others [16] worked together to take down the Waledac botnet. The entire operation was known as Operation b49 and was one of the first major actions of a larger project called Project MARS (Microsoft Active Response for Security) which is a project Microsoft is heading to combat botnets.

Although the specifics of how Operation b49 took down the Waledac botnet have not been published, we can discus what techniques they may have used. From Williams' statement in [16] the takedown occurred on every level of the botnet. They disrupted the communication of the P2P bots, which also severed communication among the worker bots as they relied on the P2P bots to communicate. In addition, they disrupted the upper levels of communication as well to prevent the P2P bots from contacting the distribution servers.

In [16] Holz mentions that Microsoft's legal department had a hand in the takedown. The legal department won a court order which allowed them to block 277 domain names used by the botmaster to control the botnet [15]. This would have completely removed the two upper levels of the communication structure until the botmaster could purchase new domain names, reconnect to the botnet, and update them to communicate through the new domain names by propagating the update through the P2P network.

Guessing how Operation b49 disrupted communication on the lower levels is more difficult. In [16], Williams describes the way they disrupted communication on this level as "peer-to-peer communication disruption through technical countermeasures..." which is pretty broad. Sinclair *et al.* [10] describes how they would take down Waledac. They propose a technique that I will refer to as *peer-list poisoning* after the term *index poisoning* [14]. Peer-list poisoning is a technique where a bot is tricked into adding bots that do not exist to its peer list. Once enough bots have enough fake entries in their lists communication halts as messages no longer propagate through the network properly. Sinclair *et al.* [10] proposed that the peer lists of Waledac could be poisoned by adding fake P2P bots to the botnet that do not do anything except respond to requests for peer lists. When their peer list is requested from the fake bot it gives a list of IP addresses that point to nothing. If enough fake bots

send out enough fake peer lists the fake entries could propagate through the network significantly reducing its ability to function.

Taking down the Waledac botnet in this way would be a large amount of work and would require co-ordination from multiple organizations. Disrupting the upper levels took Microsoft's legal team to block the domain names, which was not a trivial undertaking. First, they had to identify all 277 different domain names the servers were using, then get the court order put through to get those blocked before they changed to a different domain. Then, they had to set up enough fake bots that they could sufficiently poison peer lists to stop communication. The Waledac botnet consisted of 70,000 [15] to 160,000 [10] bots. This means that the number of fake bots counted in the thousands. Finally, all of these attacks had to occur simultaneously because if either the upper or lower level of communication remained intact the botmaster could use it to re-organize the botnet.

5. CONCLUSIONS

Understanding how the communication structures of botnets work as well as their trade offs is critical to detecting and defending from botnets. Command and control botnets are easier to create and maintain but are easier to detect. Meanwhile, peer to peer botnets are more difficult to create and maintain, but have the advantage of being harder to detect. However, there are multiple techniques to mitigate these issues with either system that in turn also have trade offs, such as using domain flux in the case of C&C botnets or piggybacking off of pre-existing networks for P2P networks.

The Miner network takes aspects of both C&C and P2P networks to make a network that is easy to maintain due to the distribution servers acting as a somewhat centralized control center, handling peer list population and communication from the botmaster to the botnet which are difficult issues in a P2P network. Meanwhile, the Miner botnet also uses bots as distributed distribution centers which means there is less traffic coming from the distribution servers so they are harder to detect as well as being able to function even if the distribution servers are removed completely [9].

The takedown of the Waledac botnet shows how difficult these bots can be to take down. Because modern botnets often have backup systems in place multiple approaches must be used simultaneously. In addition, each approach must be comprehensive enough to stop communication on that level entirely. Microsoft had to take down nearly 300 distribution servers and thousands of fake bots had to be inserted into the Waledac network to poison the peer lists to a sufficient degree to stop communication on the P2P layer. In the future, I suspect sophisticated takedowns will become increasingly common as botnet technology develops further and the technology necessary to counter them will have to grow as well.

6. REFERENCES

[1] M. Abu Rajab, J. Zarfoss, F. Monrose, and A. Terzis. A multifaceted approach to understanding the botnet phenomenon. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, IMC '06, pages 41–52, New York, NY, USA, 2006. ACM.

[2] M. Bailey, E. Cooke, F. Jahanian, Y. Xu, and M. Karir. A survey of botnet technology and defenses. In *Conference For Homeland Security, 2009. CATCH '09. Cybersecurity Applications Technology*, pages 299 –304, march 2009.

[3] H. Choi, H. Lee, and H. Kim. Botgad: detecting botnets by capturing group activities in network traffic. In *Proceedings of the Fourth International ICST Conference on COMmunication System softWAre and middlewaRE*, COMSWARE '09, pages 2:1–2:8, New York, NY, USA, 2009. ACM.

[4] M. Cova, C. Kruegel, and G. Vigna. Detection and analysis of drive-by-download attacks and malicious JavaScript code. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 281–290, New York, NY, USA, 2010. ACM.

[5] M. Jelasity and V. Bilicki. Scalable stealth mode P2P overlays of very small constant degree. *ACM Trans. Auton. Adapt. Syst.*, 6(4):27:1–27:20, Oct. 2011.

[6] S. Nagaraja, P. Mittal, C.-Y. Hong, M. Caesar, and N. Borisov. Botgrep: finding p2p bots with structured graph analysis. In *Proceedings of the 19th USENIX conference on Security*, USENIX Security'10, pages 7–7, Berkeley, CA, USA, 2010. USENIX Association.

[7] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Consulted*, 1:2012, 2008.

[8] D. Plohmann. personal correspondence, Oct. 2012.

[9] D. Plohmann and E. Gerhards-Padilla. Case study of the Miner botnet. In *Cyber Conflict (CYCON), 2012 4th International Conference on*, pages 1 –16, june 2012.

[10] G. Sinclair, C. Nunnery, and B. Kang. The Waledac protocol: The how and why. In *Malicious and Unwanted Software (MALWARE), 2009 4th International Conference on*, pages 69–77. IEEE, 2009.

[11] B. Stock, J. GoЇL andbel, M. Engelberth, F. Freiling, and T. Holz. Walowdac - analysis of a peer-to-peer botnet. In *Computer Network Defense (EC2ND), 2009 European Conference on*, pages 13 –20, nov. 2009.

[12] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna. Your botnet is my botnet: analysis of a botnet takeover. In *Proceedings of the 16th ACM conference on Computer and communications security*, CCS '09, pages 635–647, New York, NY, USA, 2009. ACM.

[13] P. Wang, S. Sparks, and C. Zou. An advanced hybrid peer-to-peer botnet. *Dependable and Secure Computing, IEEE Transactions on*, 7(2):113 –127, april-june 2010.

[14] P. Wang, L. Wu, B. Aslam, and C. Zou. A systematic study on peer-to-peer botnets. In *Computer Communications and Networks, 2009. ICCCN 2009. Proceedings of 18th Internatonal Conference on*, pages 1 –8, aug. 2009.

[15] Wikipedia. Waledac botnet — wikipedia, the free encyclopedia, 2012. [Online; accessed 12-November-2012].

[16] J. Williams. What we know (and learned) from the Waledac takedown. http://blogs.technet.com/b/mmpc/archive/2010/03/15/what-we-know-and-learned-from-the-waledac-takedown.aspx, Mar. 2010.