A Framework for Understanding Botnets

Justin Leonard, Shouhuai Xu and Ravi Sandhu
Department of Computer Science, University of Texas at San Antonio
{jleonard,shxu}@cs.utsa.edu, ravi.sandhu@utsa.edu

Abstract

Botnets have become a severe threat to the cyberspace. However, existing studies are typically conducted in an ad hoc fashion, by demonstrating specific analysis on captured bot programs or bot communication mechanisms so as to suggest means to counter them. Although such studies are important, another perhaps even more important problem that is largely left unaddressed is: How should we build a unified framework that can help us understand botnets in a systematic fashion? In this paper we make a first step towards the goal by presenting a framework, which especially suggests a general architecture that could be coupled with certain advanced techniques that have not been exploited in existing botnets. The framework also suggests a set of attributes that can be used to measure and compare botnets. Moreover, the dynamic nature of botnets (e.g., a victim machine may be powered-off during some time intervals) implies that a botnet, and thus its attributes, are stochastic in nature. This means that a meaningful comparison between botnet attributes should be based on the concept of stochastic order.

1. Introduction

A bot is a compromised computer that can carry out the commands of its master, and bots are networked to form a botnet with a topology chosen by their master. Botnets have become a severe global Internet threat [3], [8], [10], [25], [26], [40], [33], [20], [9], [10]. Although there have been investigations on countering Botnets (cf., e.g., [22], [15]), the studies are typically conducted in an ad hoc fashion, by demonstrating specific analysis on captured bot programs or bot communication mechanisms so as to suggest means to counter them. Although such studies are important, a fundamental problem that is largely left unaddressed is: How should we build a unified framework that can help us understand botnets in a systematic, rather than in an ad hoc, fashion?

Our contributions. In this paper we make a first step towards the goal by presenting a framework for understanding botnets. The framework captures botnet lifetime cycles, botnet architecture, botnet command-and-control (C&C) mechanisms, and a set of botnet attributes. In particular, the framework offers insights into how some advanced techniques — such as anonymous communication channels, secret handshakes, and gossiping, which might not have been (widely) implemented in existing botnets — could be adopted in future botnets to make them more difficult to deal with. The framework suggests a dynamic graph model to abstract botnets, and the set of botnet attributes can be used to measure and compare botnets. The attributes proposed are: *robustness*, *resilience*, *sustainability*, *exposedness*, *bandwidth consumption*, *botnet size*, *botnet master goals*, and *botnet firepower*. The attributes can be combined to capture properties; for example, the often mentioned term "stealth" may be reflected by what we call robustness, resilience, sustainability, exposedness, and bandwidth consumption.

Related work. Studies on botnets have mainly focused on exploiting certain characteristics to detect botnets. There are mainly two approaches.

- The first approach, either host-based or network-based, aims to detect botnets *without infiltrating* them. Host-based detection includes signature-based IRC botnet detection systems such as [15], and behavior-based bot detection systems such as [29], which focuses on the way bots respond to data received over the network. Network-based detection aims to detect and possibly track botnets based on, for example, DNS lookup information [25], [26], flow data across large ISP [22] or local networks [31], [30], [24], network-level conversations within centralized botnets as visible from sampled traffic flows [27], email traces whereby the defender can map botnet membership by looking for multiple bots participating in the same spam email campaign [39], the correlation of IDS-driven dialog according to a user-defined "vertical" bot infection model (accommodating inbound scanning, exploit usage, egg downloading, outbound bot coordination dialog, and outbound attack propagation) [18], the correlation of "horizontal" (spatial-temporal) network anomalies of the hosts within a network (e.g., based on the observation that the bots should exhibit the same network behaviors) [19], [17], or the aggregation of centralized C&C traffic [37].
- The second approach aims to capture and analyze bot samples and then *infiltrate* into botnets. This approach has successfully tracked IRC-based botnets [14], [1], [18], and has very recently been extended to deal with

a class of P2P-based botnets (which use *unauthenticated* content-based publish/subscribe communications for C&C). In general, this approach consists of three steps: (1) Capture and analyze a bot so as to extract information such as IP addresses of initial peers, service ports, and application-specific connection information. (2) Infiltrate into the botnet so as to receive botnet commands and even identify, for example, the central IRC server. (3) Mitigate botnets by, for example, taking IRC server offline.

Outline. The rest of the paper is organized as follows. Section 2 discusses botnet system model. Section 3 propose a dynamic graph model for botnets. Section 4 introduces a set of attributes of botnet systems. Section 5 summarizes the paper.

2. Botnet System Model

System participants. We abstract the participants in botnet systems into three categories: vulnerable computers, botnet masters, and defenders. Vulnerable computers are those computers whose vulnerabilities may allow the botnet masters to recruit them to form botnets. Botnet masters aim to recruit vulnerable computers to form botnets. The masters would be typically motivated by economic incentives. The defenders intend to prevent, detect and track bots, and trace them back to the botnet masters. They may be casted by law-enforcement such as FBI, etc. From a computation perspective, we can model all the participants as probabilistic polynomial-time algorithms.

Botnet lifetime cycle. We classify the lifetime cycle of a botnet into *formation*, *C&C*, *attack*, and *post-attack* phases. During the formation phase, a master compromises vulnerable machines that then become members of a botnet. When the master needs to instruct the bots to launch attacks, the commands are sent to the bots via a C&C mechanism. After the bots launch attacks, some bots may be exposed to the defender and may get cured (e.g., the exploited vulnerabilities are patched). Thus, the master may recruit more bots, which may be merged with the bots that were not exposed during the last attack phase to form a new botnet. The newly formed botnet is then instructed via C&C to launch attacks, and so on.

Botnet architecture. Most existing Botnets use the Internet Relay Chat (IRC) protocol [21] to facilitate their C&C communications. IRC is a protocol originally designed for supporting communications, and has an (arguably) centralized architecture. This provides a means for detecting malicious Botnets based on their communication patterns. For example, if most IRC channels involving honest users who do not deploy encryptions to protect their communications, the appearance of encrypted communications would suggest that the IRC channel may be exploited by a Botnet. Advanced architectures, such as those based on peer-to-peer communication protocols, are emerging [8], [16], [34].

In our general botnet architecture shown in Figure 1, there is a *master-controller-intruder-bot* hierarchy (more or fewer layers are also possible). Specifically, there is a botnet

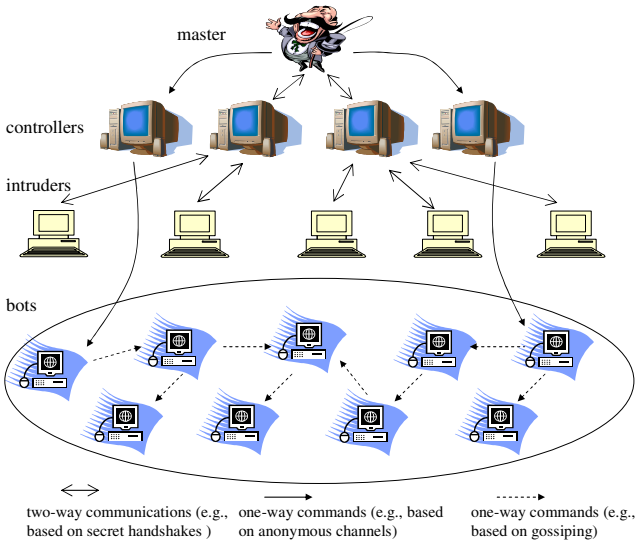

Figure 1. An example botnet architecture

master, namely the human attacker we ultimately want to catch. The master may deploy some methods to compromise vulnerable machines, which then become *controllers*. There may be two types of controllers: one type is responsible for recruiting *intruders*, and the other type is responsible for forwarding commands to the *bots*. The intruders are the compromised machines that intend to compromise more machines, which then become bots. The bots are the compromised machines that will launch actual attacks according to the master's command. Note that both the channels between the master and the controllers responsible for recruiting intruders, and the channels between the controllers responsible for recruiting intruders and the intruders, are two-way. This is because the intruders need to report to the controllers the information about the recruited bots, and the controllers need to report to the master the information about the bots. Once an intruder has recruited some number of bots, the intruder may cease to function by securely erasing its internal state information. This is for the purpose of (1) preventing the intruders from being detected as much as possible, and (2) preventing the detection of the bots because exposure of the intruders may lead to the exposure of the bots they recruited. Another possibility is that the intruder downgrades its role into an average bot.

Note that the above architecture is general because the IRC-based architecture can be seen as a simplified version of this general architecture because in the IRC-based case, there may be one special controller that serves as an IRC

server for facilitating C&C, and the intruders may be casted by average bots.

Botnet C&C mechanisms. It is clear that a botnet master wants to protect himself from being detected. There are many ways for this purpose. The simplest mechanism is to send instructions on plaintext channels; this is also the easiest to detect and thus we will not consider it any further. Another example is that the master can let the intruders send encrypted (e.g., using a public key such that the corresponding private key is known only to the master) messages to some public bulletin board, from which the master may retrieve the ciphertext messages. Yet another example is that there are multiple IP multicast channels that everyone — not just the bots — can listen to, and that different bots may listen to different multicast channels. Then, the commands can be sent via encryptions using some group-wise encryption keys, and thus can only be decrypted by the intended bots. In this case an appropriate key management scheme may be utilized by the botnet master, and the keys may be distributed to the bots when they are recruited.

Even more sophisticated methods are possible. Examples are based on abusing advanced techniques such as those briefly reviewed below.

- Anonymous communication channels: Anonymous communication channels, introduced by Chaum [6], [7], can hide the identity of a sender. There have been many efficient methods for constructing anonymous communication channels such as [28], [13], which are particularly relevant because they are appropriate for anonymous peer-to-peer communications. It is clearly possible that an attacker can abuse such channels to implement botnet C&C without fear of being detected. Another possibility is that a botnet master directly abuses some anonymous communication channels to send commands to the bots. However, this may increase the likelihood that the master being caught or detected because of the large volume of communication traffic (e.g., the over use of anonymous communication channels).
- Secret handshakes: Secret handshakes, introduced by Balfanz et al. [2], are a privacy-preserving authentication method realizing the following functionality: Two users (e.g., FBI agents) authenticate each other in a way that no one reveals its own membership (or credential) unless the peer's legitimacy was already ensured of. Secret handshake schemes can be built using one-time credentials [2], [5]), using special credentials such as group signatures and group key management [32], or using PKI-like key infrastructures [35], [36]. This technique can be abused to implement secret communications without exposing their identities to any eavesdropper (e.g., the defender).
- Gossiping: Gossiping, introduced by Demers et al. [11], has become a popular method for implementing multicast. It offers high scalability as well as reliability guarantee [4], [12]. In this paradigm, a process (or node) sends the message in question to a number of other processes (called "gossip partners" or "fan-out") appropriately selected from a set of processes (called "views"); each process that receives the message also sends the message to a randomly selected subset of processes, and so forth. Recently, we found [23] that the most desirable definition of "views" would be that the view of a node consists of its one- or multiple-hop neighbors (we call "physical views"). The variant gossip technique investigated in [23] accommodates both "the number of times (or rounds) a node gossips a message" and "physical views" mentioned above. This makes it very attractive for realizing Botnet C&C because a bot only needs to know a very small number of other bots.

3. Dynamic Graph Model for Botnets

We now present a dynamic graph model for botnets. Specifically, at time t, a botnet can be modeled as a directed graph $G_t = (V_t, E_t)$ where V_t is the vertex set consisting of the bots, and E_t is the set of arcs that reflect the "knows" relation, namely that $(u, v) \in E$ implies that bot u knows the identity or address of bot v, and that exposure of u to the defender may cause the exposure of v. Note that undirected graphs can be seen as a special case of directed graphs. Note also that G_t may correspond to the botnet formation process (e.g., if bot u recruited bot v, then $(u, v) \in E$), or may be carefully designed by the botnet master in an off-line fashion (e.g., the master instructs the bots to form a carefully selected topology at the end of the formation phase).

The dynamic modeling of botnets is general due to the following reasons.

- With respect to its lifetime $[T_1, T_2]$, a botnet can be understood as a sequence of graphs, namely $\{G_t = (V_t, E_t)\}_{t\in[T_1,T_2]}$. This is because the vertex set V_t is dynamically changing because some bots (i.e., the corresponding victim computers) may be powered-off, or some bots may actually be detected and cured by the defender. Therefore, it makes a good sense to specify a time factor, meaning that we are specific to a snapshot of a botnet.
- Denote by $\mathcal{A}$ the master of a dynamic botnet $\{G_t = (V_t, E_t)\}_{t=0,1,...}$, where $\mathcal{A} \notin V_t$ for any t. $\mathcal{A}$ can directly communicate with some *entry bot(s)*, perhaps via some out-of-botnet anonymous communication channels that do *not* belong to G_t. The entry bots then forward $\mathcal{A}$'s commands to the bots they know (with respect to the aforementioned "knows" relation) via some in-botnet communication channels, and so on. We may assume that the communications between the bots

in G_t can be under the control of law-enforcement, even if they are based on anonymous channels. This is reasonable because it is likely that legislation in the future will regulate that such channels should provide a management interface to law-enforcement; without getting into further details, we mention that existing cryptographic techniques certainly suffice this purpose.

- The distinction between in-botnet channels, which can be under the control of law-enforcement, and out-of-botnet channels, which may not be under the control of law-enforcement, makes our model flexible. For example, the model accommodates the extreme case that every bot is an entry bot that may receive commands from $\mathcal{A}$ using out-of-botnet anonymous channels. Moreover, our model accommodates both current and future botnets. Specifically, our model not only accommodates well known stealthy C&C mechanisms, such as the aforementioned anonymous communication channels, but also accommodates more exotic P2P techniques such as gossiping [4] and secret handshakes [32]. Future botnets will likely take full advantage of P2P C&C techniques [8]. In particular, the technique of secret handshakes could be abused to fulfill C&C while achieving the best stealth because does not force $\mathcal{A}$ to over use out-of-botnet anonymous channels, which still could expose $\mathcal{A}$.

4. Botnet Attributes

The goal of identifying botnet attributes is to help measure and compare botnets or botnet architectures. Based on the above dynamic botnet graph model, the attributes should be deemed as *random variables*, rather than *deterministic numbers*. As a result, a meaningful comparison between two botnets or two botnet architectures, at least from a theoretic perspective, should be based on the notion of *stochastic order*. Specifically, we say random variable X is stochastically larger than random variable Y, denoted by $X \geq_{st} Y$, if for any r it holds that $\Pr[X > r] \geq \Pr[Y > r]$.

In order to specify the attributes, we denote by $\mathbb{G}$ the set of all possible botnets, by $\mathbb{V}$ the set of all possible bots, by $\mathbb{C}$ the set of all possible C&C mechanisms, and by $\mathbb{N}$ the set of positive integers.

Robustness. Intuitively, this attribute captures the minimal number of bots that need be detected by the defender, in order for the defender to trace all the bots associated with the same botnet. Since G_{t1}, the botnet graph at time $t1$, is typically different from G_{t2}, the botnet graph at time $t2$ where $t1 \neq t2$. The robustness of a botnet at time $t1$, denoted by $\mathsf{robustness}(G_{t1})$ is typically different from the robustness of the same botnet at time $t2$, denoted by $\mathsf{robustness}(G_{t2})$. Moreover, since G_t is a random variable, so is $\mathsf{robustness}(G_t)$.

Resilience. Intuitively, this attribute captures the consequences when some bots of a botnet $G_t = (V_t, E_t)$ at time t are exposed to the defender (who can then trace other bots based on the "knows" relation mentioned above). This attribute may be seen as a function $\mathcal{R}(\cdot,\cdot) : \mathbb{G} \times \mathbb{V} \rightarrow [0, 1]$. A good definition of this function should accommodate the following intuitions. At one extreme (i.e., worst resilience), G_t has a star topology (e.g., the current IRC-based botnets), which may cause $\mathcal{R}(G_t, \{u\}) = 0$ for any $u \in V_t$ because exposure of any bot u could lead the defender to trace to the hub, and thus all the other bots. At the other extreme (i.e., best resilience), $E_t = \emptyset$, meaning that each bot directly receives commands from $\mathcal{A}$ via an anonymous channel, or via the aforementioned secret handshake. For such G_t, it is clear that $\mathcal{R}(G_t, V^*) = 1$ for any $V^* \subset V_t$ because the defender may at least need to conduct on-line secret handshakes to "hunt" other bots in a one-by-one fashion. Since G_t is a random variable, so is $\mathcal{R}(\cdot,\cdot)$.

Note that the above *robustness* and *resilience* are, in a sense, two sides of the same coin. On one hand, the former captures the minimal number of exposed bots necessarily needed in order to trace all the bots belonging to V_t. Suppose $V' \subseteq V_t$ is such a set of bots. On the other hand, the latter captures the consequence when a subset of bots are exposed. Suppose $V^* \subseteq V_t$ is such a set of bots. In the case $|V'| > |V^*|$, it is clear that exposure of V^* will not lead to the exposure of all the bots in V_t. However, even if $|V'| \leq |V^*|$, it does not necessarily mean that exposure of bots in V^* will necessarily lead to the exposure of all the bots in V_t. This is because some exposed bots may be "redundant" in the sense that their exposure can be caused by a few other bots. In contrast, exposure of bots in V' always leads to the exposure of all the bots in V_t.

Sustainability. This attribute captures the interaction between a botnet master and the defender, namely that some bots may be eventually detected by the defender and thus eliminated from a botnet, and then the master may recruit more bots to replace the eliminated bots. A good definition of *sustainability* should capture the effect due to the elimination of different bots. For example, the effect of eliminating a bot that "knows" many other bots is different from the effect of eliminating a bot that "knows" a few other bots in terms of the connectivity of the remaining botnet, or in terms of the size of the maximal connected component of the remaining botnet.

Since the criteria for optimization are specific to the concrete needs (e.g., connectivity or something else), we do not aim to emulate all possible definitions. Instead, we give an example by assuming that the criterion is based on the size of connected bots. In this case, we can define sustainability, denoted by S_n, as

$$S_n \stackrel{def}{=} \frac{\min_{\forall V' \subset V_t, |V'|=n|} \left(\max_{V'' \subseteq V_t - V', V'' \text{ is connected}} |V''|\right)}{|V| - n},$$

which stands for the ratio of the size of the minimum among the maximal connected component possible after eliminating $n < |V_t|$ bots. We say $G_t = (V_t, E_t)$ has optimal sustainability if $S_n = 1$ for any $1 \leq n \leq |V_t|$, and has a good sustainability if $S_n \geq 1 - \epsilon$ for any $1 \leq n \leq |V_t|$ and some desired $0 \leq \epsilon \leq 1$.

Exposedness. Intuitively, this attribute captures the probability that a bot is exposed to the defender because of its C&C activities when there are no already-exposed bots. Depending upon the C&C mechanisms, this attribute may be the *worst-case* probability that a bot is exposed due to its participation in fulfilling C&C. Let d be the defender capability, which may be treated as a threshold probability d of exposure. In other words, if a bot's exposedness is above the detection threshold, then bot is deemed exposed. A botnet $G_t = (V_t, E_t)$ has a low exposedness if the C&C assures that the exposedness of any bot is below the detection threshold d.

Bandwidth consumption. This attribute captures the bandwidth consumption caused by a C&C mechanism with respect to a botnet $\{G_t = (V_t, E_t)\}_{t \in [T_1, T_2]}$. Therefore, this attribute may be seen as a function $\mathbb{G} \times \mathbb{C} \rightarrow \mathbb{N}$. Moreover, this metric may capture the total or average number of messages incurred by $C \in \mathbb{C}$ over a botnet.

Note that one difference between bandwidth consumption and the above exposedness is that they capture two different aspects of the defender's capability, namely the detection based on seeing many C&C communications vs. the detection based on seeing the activity of a specific bot (i.e., the actual victim machine).

Botnet size. Given that V_t is a random variable, botnet size $|V_t|$ is also a random variable. It may be more meaningful to define the size of a botnet as the number of bots that actually participated in an attack at a specific time. Therefore, when we compare two botnets, in terms of their sizes, it may be more meaningful to compare the random variables corresponding to the number of bots that actually participate in an attack.

Botnet master goals. A botnet master may utilize a botnet to launch various attacks. Representatives are:

- Making a revenue by directly using botnets to launch attacks: This is perhaps the currently most popular attack scenario, where the master uses botnets to launch attacks such as distributed denial-of-service, spam, or blackmailing the victim users for ransom by encrypting their data on disk [38].
- Making a revenue by selling botnets per se: In this case, a botnet master sells the control of a botnet to other malicious attackers. There may have been such a underground market.
- Making a revenue by selling critical and private data: In this case, a botnet master is mainly interested in collecting and harvesting critical data, such as cryptographic keys, passwords, email address books, email communications, working documents. The harvested critical data may be further sold to other malicious participants who have the interest to break the security and privacy of the victim users.

Botnet firepower. One simple way to define the firepower of a botnet is to treat it as the number of bots that actually participated in an attack, namely *botnet size* defined above. However, the firepower should also depend on the resource a bot, namely the corresponding victim machine, can dedicate to launch the attack in question. Depending on the goal of the botnet master, this may be defined as the bandwidths the bots can dedicate to launch distributed denial-of-service attacks, or as the number of valid private keys that the botnet has harvested. Note that the harvest of many private keys would cause catastrophic consequences such as there being a large number of disputes and lawsuits because of the digital signatures generated using the compromised private keys, or because of the adversaries being able to arbitrarily impersonate the victim users.

5. Conclusion and Future Work

We presented a framework for understanding botnets in a systematic, rather than in an ad hoc, fashion. The framework suggests a general architecture that could be coupled with certain advanced techniques, and also suggests a set of important attributes that can be used to measure and compare botnets. Moreover, we point out, the first time, that the dynamic nature of botnets implies that a botnet, and thus its attributes, are stochastic in nature.

There are many research problems that remain to be explored within the framework. A class of problems can be stated as follows: Given a botnet $\{G_t = (V_t, E_t)\}_t$ and a defender capability d, how can we quantitatively measure the attribute A of botnet $\{G_t\}$ under the defender capability d? The other line of research is to refine the above framework to incorporate, for example, sophisticated defense strategies that may turn the defender capability d into stochastic $\{d_t\}_t$ as well.

Acknowledgment

The authors are partially supported by a grant from the State of Texas Emerging Technology Fund.

References

[1] P. Bacher, T. Holz, M. Kotter, and G. Wicherski. Know your enemy: Tracking botnets. http://www.honeynet.org/papers/bots/, dated on 13 March 2005.

[2] D. Balfanz, G. Durfee, N. Shankar, D. Smetters, J. Staddon, and H. Wong. Secret handshakes from pairing-based key agreements. In *IEEE Symposium on Security and Privacy*, 2003.

[3] P. Barford and V. Yegneswaran. An inside look at botnets. In *Workshop on Malware Detection*, 2006.

[4] K. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, and Y. Minsky. Bimodal multicast. *ACM Trans. Comput. Syst.*, 17(2):41–88, 1999.

[5] C. Castelluccia, S. Jarecki, and G. Tsudik. Secret handshakes from ca-oblivious encryption. In *Asiacrypt'04*.

[6] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *CACM*, 84–88, 1981.

[7] D. Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of Cryptology*, 1:65–75, 1988.

[8] E. Cooke, F. Jahanian, and D. McPherson. The zombie roundup: Understanding, detecting, and disrupting botnets. In *Proc. SRUTI'05*, 2005.

[9] D. Dagon, G. Gu, C. Lee, and W. Lee. A taxonomy of botnet structures. In *ACSAC'07*.

[10] D. Dagon, C. Zou, and W. Lee. Modeling botnet propagation using time zones. In *Proc. NDSS'06*, 2006.

[11] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, and D. Terry. Epidemic algorithms for replicated database maintenance. In *PODC*, pages 1–12, 1987.

[12] P. Eugster, R. Guerraoui, S. Handurukande, P. Kouznetsov, and A. Kermarrec. Lightweight probabilistic broadcast. *ACM Trans. Comput. Syst.*, 21(4):341–374, 2003.

[13] M. Freedman and R. Morris. Tarzan: a peer-to-peer anonymizing network layer. In *ACM CCS'02*.

[14] F. Freiling, T. Holz, and G. Wicherski. Botnet tracking: Exploring a root-cause methodology to prevent denial-of-service attaks. In *ESORICS'05*.

[15] J. Goebel and T. Holz. Rishi: Identify bot contaminated hosts by irc nickname evaluation. In *HotBot'07*.

[16] J. Grizzard, V. Sharma, C. Nunnery, B. Kang, and D. Dagon. Peer-to-peer botnets: Overview and case study. In *HotBot'07*.

[17] G. Gu, R. Perdisci, J. Zhang, and W. Lee. BotMiner: Clustering analysis of network traffic for protocol- and structure-independent botnet detection. In *USENIX Security'08*.

[18] G. Gu, P. Porras, V. Yegneswaran, M. Fong, and W. Lee. Bothunter: Detecting malware infection through ids-driven dialog correlation. In *USENIX Security'07*.

[19] G. Gu, J. Zhang, and W. Lee. Botsniffer: Detecting botnet command and control channels in network traffic. In *NDSS'08*.

[20] A. Hirt and J. Aycock. Anonymous and malicious. In *Proc. Virus Bulletin International Conference'05*, pages 2–8, 2005.

[21] C. Kalt. Internet relay chat: Architecture (ietf rfc 2810), April 2000.

[22] A. Karasaridis, B. Rexroad, and D. Hoeflin. Wide-scale botnet detection and characterization. *HotBot'07*.

[23] X. Li, P. Parker, and S. Xu. A Stochastic Characterization of a Fault-Tolerant Gossip Algorithm. In *HASE'07*.

[24] C. Livadas, R. Walsh, D. Lapsley, and W. Strayer. Using machine learning techniques to identify botnet traffic. In *WNS'06*.

[25] M. Rajab, J. Zarfoss, F. Monrose, and A. Terzis. A multifaceted approach to understanding the botnet phenomenon. In *IMC'06*.

[26] A. Ramachandran, N. Feamster, and D. Dagon. Revealing botnet membership using dnsbl counter-intelligence. In *Proc. SRUTI'06*.

[27] A. Ramachandran, S. Seetharaman, N. Feamster, and A. Lakhina. Monitoring stealthy network conversations with sampled traffic. Technical report, Georgia Institute of Technology, 2006.

[28] M. Reiter and A. Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.

[29] E. Stinson and J. Mitchell. Characterizing bots' remote control behavior. In *DIMVA 2007*.

[30] W. Strayer, D. Lapsley, R. Walsh, and C. Livadas. *Botnet Detection: Countering the Largest Security Threat*, In Advances in Information Security, 2008.

[31] W. Strayer, R. Walsh, C. Livadas, and D. Lapsley. Detecting Botnets with Tight Command and Control. *LCN'06*.

[32] G. Tsudik and S. Xu. A flexible framework for secret handshakes. In *PET'05*.

[33] R. Vogt, J. Aycock, and M. Jacobson. Army of botnets. In *Proc. NDSS'07*, 2007.

[34] P. Wang, S. Sparks, and C. Zou. An advanced hybrid peer-to-peer botnet. In *HotBot'07*.

[35] S. Xu and M. Yung. k-anonymous secret handshakes with reusable credentials. In *ACM CCS'04*.

[36] S. Xu and M. Yung. k-anonymous multi-part secret handshakes. In *FC'07*.

[37] T. Yen and M. Reiter. Traffic aggregation for malware detection. In *DIMVA'08*.

[38] A. Young and M. Yung. Cryptovirology: Extortion-based security threats and countermeasures. In *IEEE Symposium on Security and Privacy*, 1996.

[39] L. Zhuang, J. Dunagan, D. Simon, H. Wang, I. Osipkov, G. Hulten, and J. Tygar. Characterizing botnets from email spam records. In *LEET'08*.

[40] C. Zou and R. Cunningham. Honeypot-aware advanced botnet construction and maintenance. *DSN'06*.