Detecting Covert Botnets Using Communication Patterns

Neal V. Sorensen, Sarah B. Sorensen, Kyle D. Feuz, Grigoriy Kerzhner and Chad D. Mano
Department of Computer Science
Utah State University
Logan, UT 84322, USA
Email: neal.sorensen[at]usu.edu, kyle.feuz[at]aggiemail.usu.edu,
gik[at]duke.edu, chad.mano[at]usu.edu

***Abstract*—Botnets are continuously evolving to evade detection. Current botnet detection systems are designed to be positioned on the edges of a private network. Covert botnets have been propsed which can evade this kind of detection. Switch-based monitoring as an addition to an edge-based detection system is required to detect these covert botnets. This paper presents an enhancement to a current switch monitoring system to include network traffic pattern detection so that covert botnets using encrypted communication can still be detected. This paper proposes several communication patterns that are indicative of covert botnets in a sub-network.**

I. Introduction

Botnets are the latest and most serious computer security threat [3], [5]. Botnets are a network of infected computers throughout the world that can be controlled by an attacker to do whatever he or she commands [1]. The attacker is called the botmaster and the infected machines are called bots. When botnets grow to a large size [4], their distributed attacking capabilities are not easily defended [15]. Large corporate systems and websites can be attacked and brought down easily by large botnets [13], resulting in huge losses. Botnets can also be used to collected sensitive data from their host systems [10].

Infected computers are compromised subtly and may execute attacker commands without the computer's owner even knowing. Botnets can become very large and persist because the people who's computers are infected don't even know it. Botnet detection is needed before the removal of these threats can even begin. Botnets started as simple, centralized command and control network structures that could easily be detected, but they are evolving to evade even the most advanced detection systems. Covert botnets that coordinate external communications within switched sub-networks have been proposed. These covert botnets can avoid perimeter detection systems, such as BotHunter [6], [11]. Complete switch monitoring coverage is required for the detection of these covert botnets.

In this paper we show how covert botnets in a sub-network can use encryption to evade and complicate detection. We then use network traffic analysis techniques to discover distinct sub-network botnet communication patterns that can be used for detection. These patterns are different than other patterns used by perimeter detection systems [7], because they are specific to covert bots in switched sub-networks. Just as binary signature scanning uses a set of signatures for detection, enhanced detection can scan for communication patterns in network traffic from a set of given, known botnet communication patterns. In this paper we propose to combine binary signature scanning with communication pattern scanning at switch monitors to provide a more thorough detection system which can provide greater detection ability and accuracy.

II. Related Work

Peer-to-peer botnets are the most advanced form of botnets found throughout the Internet today [5], [9], [15]. These botnets are very difficult to detect because they do not use a centralized command and control location. Peer-to-peer bots communicate with each other to distribute commands. To the best of our knowledge, BotHunter [6] is the most effective botnet detection system, even for peer-to-peer botnets. BotHunter uses infection dialogs to detect the presence of a bot in a network.

The authors of [11], [12] proposed a specialized peer-to-peer botnet that can evade BotHunter detection using covert bots. These covert bots in a sub-network coordinate external communications with each other to evade detection by the edge-based detection system, BotHunter. The authors of [11], [12] have also proposed a switch based botnet detection system, which is an extension to BotHunter, to detect these covert bots. This system uses monitors at each switch in a network to provide complete coverage of the network and monitors internal to internal botnet communication. The switch monitoring system correlates detection alerts and passes them to BotHunter which combines internal switch alerts with external Snort alerts. The author of [2] has implemented the covert subnet bot and the switch based detection system and proved that the covert bots can evade edge-based BotHunter detection and the extended monitoring system can detect the covert bots. Covert bots have not yet been detected in the wild, but they are possible threats.

By improving the covert botnet that was developed in [2] to use encrypted communication, the covert botnet will evade the current switch monitoring detection. The current switch monitoring system only uses signature scanning to detect bot communication; encrypted bot communication will cause this detection to fail. In this paper, we propose an enhancement to the switch monitoring system that can detect covert bots in a sub-network. The enhancement will be able to scan for network traffic communication patterns between the covert bots despite of encrypted communication. We also present several communication patterns that are used by covert bots using a token based model to communicate. These patterns can be detected by the enhanced switch monitoring system and lead to the detection of covert bots.

III. Evading Basic Signature Scanning

The implementation of the covert bots in [2] uses TCP connections and raw Ethernet frames for passing messages and binaries in order to coordinate the bots in the switched network. A token based model from [12] is used as the main structure for coordination and communication. One bot, the token bot, is used at a time to make external communications. The token bot distributes the binaries that it received externally to the other bots in the sub-network. The token bot assignment is rotated strategically among the covert bots so that the internal to external detector, BotHunter, cannot detect the presence of any bots in the network. The message and binary exchanges are done without encryption. The switch monitoring system scans the network traffic and can detect these unencrypted communications by scanning for known signatures in the bot binaries that are downloaded and distributed among the covert bots. The switch monitoring system sends alerts to BotHunter to detect the covert bots.

We added symmetric key, block cipher encryption to the TCP and raw Ethernet sockets used in the covert bots to encrypt all messages and binary being sent to other internal bots. The same kind of encryption could be added to any other kind of socket, such as UDP, that is used for bot communication. On the receiving end of the sockets, decryption was added to decrypt all internal, incoming messages or binaries. We also salted each encryption with the IP or MAC address of the receiving bot. Other salts or encryption keys can be used and even changed periodically so that the encryption will remain secure and internal to the covert bots. By using a different salt with each encryption, the same binary sent from one bot to the other bots will be different and will complicate the detection of the bots. Other individualized encryption schemes can be used for the same purpose, such as one presented in [15]. As shown in Figure 1, each binary will appear differently as it is transferred to each covert bot from the token bot. Each bot contains its own decryption capabilities, so no decryption stubs will be present in the binaries. This will eliminate any possible signature scanning of decryption stubs.

Once the encryption was added to the sockets in the covert bots, the bots were run and tested to ensure the same and proper function as before the encryption was added. Only the sockets had to do the extra work to communicate, the core bot code was left unchanged. Adding encrypted sockets to bots is a simple and easy modification that bot creators can use to make detection significantly more difficult. Any recognizable signatures in our covert bot's binary became completely hidden from signature scanners and the covert botnet operated fully without any detection. External to internal encryption techniques could also be developed for botnets, but our focus is on switch monitoring which is aimed at detecting internal to internal bot communication.

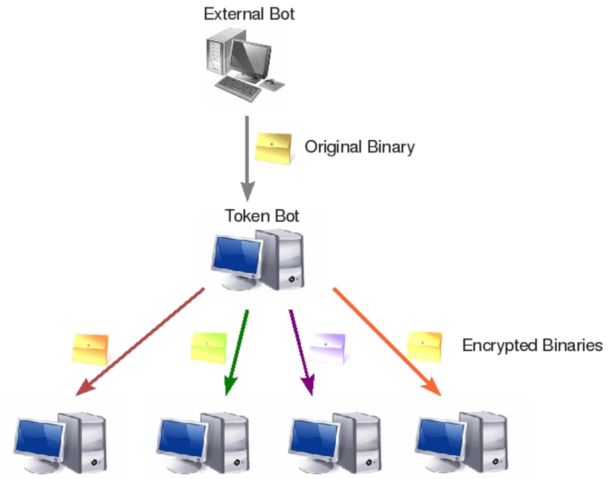


Fig. 1. Encrypted Binary Distributions Among Covert Bots

Encrypted botnet communication can also reduce the ability to mitigate or disrupt a botnet. Attempts have been made to mitigate peer-to-peer botnets by injecting commands into botnets [8]. Encryption provides the means for authentication into the botnet communication. Without proper encrypted commands or messages, any communication injected into the botnet would be ignored and the botnet would not be disturbed unless communication channels were flooded with traffic. Flooding the communication channels would only be a temporary disruption.

IV. Discovering Sub-network Covert Bot Communication Patterns

The best way for us to determine what kind of network traffic patterns are generated by a covert botnet in a switched sub-network was to collect traffic data from the encrypted botnet used in the previous section. We ran the botnet consisting of seven active bots for 105 minutes and collected all network traffic the bots generated. During this time, the covert bots made about 380 external downloads and distributed the them among the other bots. This is a high rate of external downloads; the bot were allowed to run as quickly as they could. In addition to binary distributions, a significant amount of messages were sent between the bots to coordinate themselves within the switched network.

Bots taking advantage of a token based communication model need to be closely coordinated and synchronized. A lot of bot communication traffic was generated for analysis. In order to correlate IP traffic with non-IP traffic, we created an IP to MAC address table. This table is generated when IP and ARP packets are parsed. When raw Ethernet packets are processed, which don't contain IP addresses, they can be obtained from the IP to MAC table which was generated from previous traffic. This will help us correlate the traffic from different network layers that the covert bots may use.

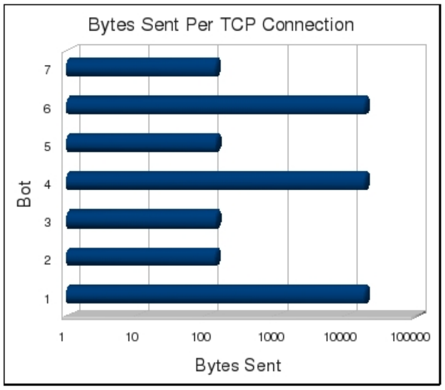


Fig. 2. TCP Traffic Pattern Data From Covert Bots

To analyze the network traffic data, we created scripts to parse through the data. The traffic was parsed packet by packet and we logged packet information sent in between every device. From the logged data, we could perform queries to obtain specific information about a certain protocol or traffic generated by one device or between a set of devices. Traffic only needed to be collected, parsed, and logged once. Different patterns can be identified independently by querying the same traffic logs. In many locally switched networks, such as a university computer lab, it is uncommon for systems to communicate directly with each other. The presence of even small network traffic between systems in the switched network may be sign of questionable activity [11].

Patterns can be protocol independent to provide a more general coverage or protocol dependent for more specific cases, like those found in [7]. More specific patterns can increase detection accuracy, but could also lower detection rates. Patterns may be specific enough to include a certain port or a pattern of ports used by a covert botnet. Our covert bot implementation used hard-coded port numbers, but individualized port assignments can be used [15] to make detection more difficult.

A. *TCP Connection Patterns*

TCP connection patterns were some of the most obvious traffic patterns generated by the bots. Within 15 seconds, 6 different systems made TCP connections to the same system and received the same amount of data from that system. This is obviously traffic generated by token bot distributing downloaded data to its peers. This pattern repeated and a different device became the distributing device. Another interesting pattern from the TCP traffic was that each bot always sent similar sized binary each time all the other bots connected to it. Figure 2 shows the amount of data sent by each bot per TCP connection. Bots 1, 4, and 6 appear to be the distributors of bot binary updates because of the large amount of data they sent each time. Bots 2 and 7 sent the same sized data, and bots 3 and 5 sent the same sized data. The two sets of smaller sized data could be distributions of peer lists and command lists. These TCP traffic patterns are unique patterns that can be used with other patterns to detect the covert bots.

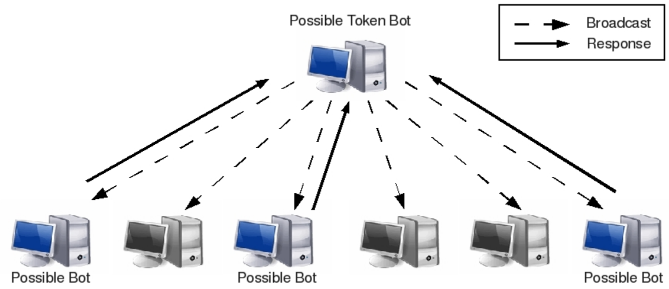


Fig. 3. Covert Bot Broadcast Traffic

B. *Broadcast and Response Patterns*

Broadcasts were the next recognized traffic patterns generated by the bots. When a system would send out a broadcast, it would get six quick responses each time. These were not common traffic broadcasts, such as ICMP or ARP packets. They were raw, unknown protocol Ethernet packets and responses. These broadcast patterns were repeated often. Other covert botnets could be implemented using common broadcasts with encrypted messages specific for bots to respond to. Frequent broadcast traffic and responses to the broadcasts are not evidence enough to ensure a botnet exists in the network, but it is a pattern that raises suspicion of a covert botnet infection. Figure 3 illustrates this traffic pattern.

C. *Dialog Traffic Patterns*

We define dialog as connectionless communication between two systems. This type of communication can be identified by detecting several same-protocol packet exchanges between two systems within a short amount of time. We found from the covert botnet traffic that within 30 seconds, each bot had the same kind of dialog with at least two other bots in the botnet. By forming graphs of dialogs within 30 second time frames, we were able to form connected graphs that contained each bot in the botnet. Members of these graphs can be considered as possible bots. This pattern allows us to detect communications using unknown protocols as well. We also recognized other traffic dialog patterns in the covert botnet by monitoring the number of same-protocol packets exchanged between systems per minute and the number of same-protocol bytes sent between systems per minute. Thresholds could be set on these kinds of measurements to raise alerts of possible botnet communication. Highly delayed dialog may prevent the detection of certain dialog patterns.

D. *Other Possible Patterns*

There are many ways that covert bots can communicate with each other in a sub-network. We have only analyzed one implementation of a botnet using a token based model. The token based model can also be implemented differently to produce different traffic patterns than the ones we detected. Other models can be implemented in various ways and each one will generate different, detectable traffic patterns. One possible traffic pattern is a circular traffic pattern. Instead of a bot connecting with or talking with all other bots, it can pass a message or binary along to its neighbor only. Figure

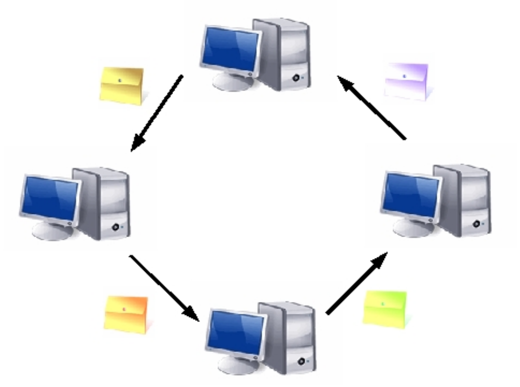


Fig. 4. Circular Bot Communication Pattern

4 illustrates this kind of pattern. The data passed around in a circle could be recognized by its common size each time it is passed to the next bot. Detection of this pattern can be done by finding simple cycles in a graph of systems sending data to each other. This pattern could be combined with other patterns to detect the bots.

Apart from bot communication, patterns of infection spreading can also be detected in a sub-network. A bot may noisily scan other systems in the sub-network to find vulnerabilities. Exploit patterns from one internal system to another can be detected from switch monitored traffic. As shown in [12], these internal to internal patterns can be combined with other external alerts to detect the presence of a bot with BotHunter. Our covert bot implementation did not use scans or common exploits to infect other systems, so these patterns would not be detected in our covert bot network traffic, but they should be monitored.

V. Enhanced Detection Using Communication Patterns

When detecting bots by communication patterns, covert bots can be detected by groups or individually. Bot spreading patterns would raise detection alerts for only one bot in the network. Communication patterns between covert bots exist between a set of bots, and don't originate from one bot. A disadvantage of covert bots, but an advantage to the detection of them, is all of them in sub-network may be detected at once.

The detection of one communication pattern in network traffic is not enough evidence to raise an alert of a botnet infection. These patterns can often be seen individually on a network, but they aren't caused by covert botnets. To reduce false negative detections, we will use a weighted pattern threshold system similar to the one used by BotHunter [6]. If we detect a combination of communication patterns, rather than just one, produced by the same group of systems, we can have a greater assurance that they are part of a botnet. Groups of only two bots produce too many false negative alerts. We will set the minimum amount of bots in a sub-network group to be three.

We observed from testing our pattern detection algorithms, that other systems often get included with pattern groups when looking for communication patterns. When looking for botnet groups, if a group that has participated in one kind of pattern is a subset of a group in a different pattern, we will identify the subset as a botnet group. A group of systems must be a part of two or more of the group patterns mentioned in section IV, before we will raise a detection alert on those systems. Certain communication patterns may require a different combination of patterns to raise detection alerts.

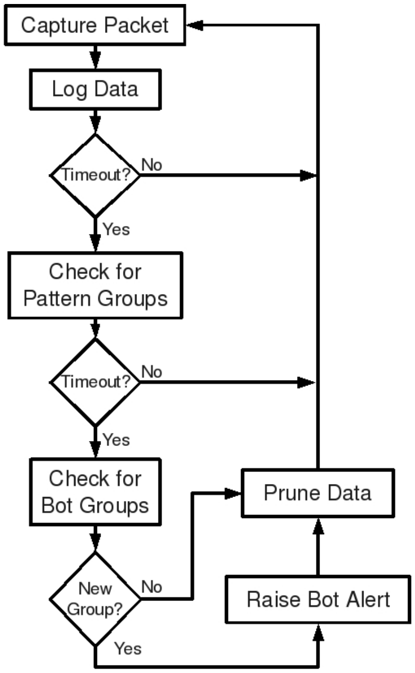


Fig. 5. Pattern Detection Flow Chart

In order to keep memory usage down and to enable the detection system to run quicker, we also implemented data pruning. The traffic data can be cleared out periodically at a chosen rate. This is needed if there are high volumes of traffic and if there are a lot of recognizable patterns in the traffic.

A single threaded implementation of our pattern detection system is illustrated by the flow chart in Figure 5. Timeouts are used to control how often we check for patterns and bot groups. These values can be adjusted depending on the available memory and processing. Logged data is used to search for communication patterns among computers. The groups formed from the patterns are then analyzed to form groups of bots in the network. A multi-threaded version of the system could improve packet capturing performance and communication pattern detection.

VI. Analysis

After implementing the communication pattern detection system mentioned above, we ran network traffic data through the detection system. The detection system was able to detect groups of bots over and over again as they communicated with each other. Each time the bots coordinated with each other, they created several kind of communication patterns which we detected. No systems that did not host bots were included

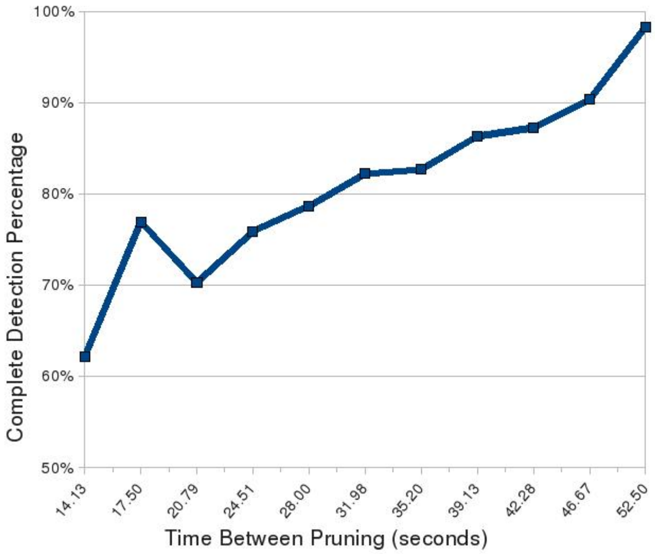
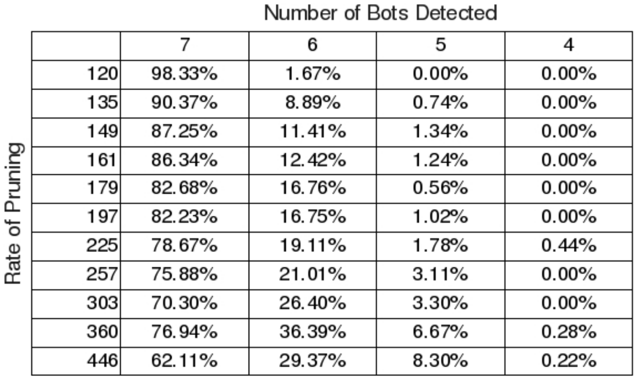


Fig. 6. Percentages of Complete Bot Detection With Different Pruning Rates



Number of Bots Detected

Rate of Pruning	7	6	5	4
120	98.33%	1.67%	0.00%	0.00%
135	90.37%	8.89%	0.74%	0.00%
149	87.25%	11.41%	1.34%	0.00%
161	86.34%	12.42%	1.24%	0.00%
179	82.68%	16.76%	0.56%	0.00%
197	82.23%	16.75%	1.02%	0.00%
225	78.67%	19.11%	1.78%	0.44%
257	75.88%	21.01%	3.11%	0.00%
303	70.30%	26.40%	3.30%	0.00%
360	76.94%	36.39%	6.67%	0.28%
446	62.11%	29.37%	8.30%	0.22%

Fig. 7. Percentages of Bot Detection With Different Pruning Rates

in the detection, resulting in a zero false positive rate for our particular set of data.

We ran the detection system several times using different rates of pruning. Pruning keeps the detector's memory cleared up and helps it run faster. If we pruned our pattern data often, we detected a higher percentage of partial bot groups rather than the complete group of seven bots in the sub-network. As shown in Figures 6 and 7, if there is a slower frequency of data pruning, we get a greater percentage of detections of the complete sub-network botnet. It takes time for the bots to communicate, and if we prune data before communication patterns are completed, we will miss detecting some of the bots in the network.

We ran a wide variety of other network traffic capture files through our pattern detection system to check for false positives. Some of the capture files contained enterprise network traffic and common computer lab traffic. When we did not limit the our detection to our internal sub-network only, we had many false positive group alerts. These groups consisted of systems with ip addresses in different ranges that would normally not be found on the same sub-network. Our communication patterns are specific to sub-network traffic patterns, and we discovered through the false negative alerts that they are not applicable to a wide area network. Network management protocols also seemed to raise unwanted alerts in our testing. The false positive alerts did not occur as frequent as the actual covert botnet alerts. The legitimate alerts stood out much more because of the large number of them that were generated. We eliminated false positives and improved our detection system by making our communication patterns more specific. This involved limiting our patterns to the sub-network that we were monitoring only and by handling valid protocols that raise alerts.

Our pattern detection system isn't stand-alone when trying to detect bots. It will be combined with the switch-based detection system that is being developed [12]. The alerts generated by our pattern detection system on a switch will be passed to an event correlator, BotHunter. This complete network detection system is explained in section II. A correlator is what gets the overall picture of what is going on in the network and determines when to confirm if there are bots in the network. Our analysis is only one part of the detection system and does not represent results from the entire detection system.

With only looking for a combination of two of three communication patterns, we were able to detect all of the covert bots in the sub-network. Any partial group detections were detected significantly less times than the full group of bots, especially if we decreased how often we pruned the data. The pattern detection system can be improved by including more communication patterns that covert botnets may use. Our new method of detection proves to be effective and accurate after modifying it to eliminate false positives. By combining pattern detection with signature scanning in the switch detection system, covert bots can be detected more effectively.

VII. Summary and Future Work

We have shown how covert botnets can add encryption to evade detection by basic binary signature scanning. Encryption is a simple addition that heavily complicates botnet detection. To detect covert botnets in a sub-network, we have proposed a way to enhance switch-based monitors by including communication pattern detection. It did not take long to distinguish common communication patterns in covert botnet traffic. Detecting combinations of these communication patterns proved to be effective and accurate in detecting the presence of covert bots.

Future work includes improving covert botnets to make them harder to detect. This can be done by designing and implementing an external to internal encryption scheme to evade edge-based detection of binary transfers. Additional encryption for internal to internal bot communication and coordination can also be increased by changing message and binary sizes with random padding. This will make it difficult for detectors to recognize size related network traffic between bots.

Other future work will involve improving our switch-based detection system. It can be improved by finding additional communication patterns used by cover botnets. To identify other communication patterns, we will implement additional covert botnets based on different sub-network communication models [12]. By analyzing the network traffic generated by implementations of other covert botnet models, we will be able to form a more complete set of communication patterns for the detection system.

The switch-based detection system could also be improved by automatic communication pattern generation. An intelligent system could be developed to identify what communication patterns are indicative of covert botnets in a sub-network. Similar systems have been developed to automatically generate binary signatures [14]. There are a lot of challenges of distinguishing botnet traffic from legitimate, normal traffic. Botnets could use existing P2P protocols or imitate other common protocols or traffic in a sub-network. An automated system to generate background traffic would also aid in the testing of our developing detection system.

Botnet detection is getting better, but so are the botnets and detection needs to be enhanced. Our research can greatly benefit future detection tools and future research in relation to botnet detection. Our goal is to detect future threats through forward-looking research. Advanced covert bots may soon be seen in the wild, and we want to be prepared to detect them.

REFERENCES

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

[2] Lokesh Babu Ramesh Babu. Covert Botnet Implementation and Defense Against Covert Botnets. Master's thesis, Utah State University, April 2009.

[3] David Barroso. Botnets the silent threat. Technical report, 2007.

[4] David Dagon, Guofei Gu, and Christopher Lee. A taxonomy of botnet structures. In *Botnet Detection*, pages 143–164. 2008.

[5] Julian B. Grizzard, Vikram Sharma, Chris Nunnery, Brent ByungHoon Kang, and David Dagon. Peer-to-peer botnets: overview and case study. In *HotBots'07: Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*, pages 1–1, Berkeley, CA, USA, 2007. USENIX Association.

[6] Guofei Gu, Phillip Porras, Vinod Yegneswaran, Martin Fong, and Wenke Lee. BotHunter: Detecting malware infection through ids-driven dialog correlation. In *Proceedings of the 16th USENIX Security Symposium (Security'07)*, pages 1–16, August 2007.

[7] Guofei Gu, Junjie Zhang, and Wenke Lee. BotSniffer: Detecting botnet command and control channels in network traffic. In *Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS'08)*, February 2008.

[8] Thorsten Holz, Moritz Steiner, Frederic Dahl, Ernst Biersack, and Felix Freiling. Measurements and mitigation of peer-to-peer-based botnets: a case study on storm worm. In *LEET'08: Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, pages 1–9, Berkeley, CA, USA, 2008. USENIX Association.

[9] Phillip Porras, Hassen Sadi, Vinod Yegneswaran, Phillip Porras, Hassen Sadi, and Vinod Yegneswaran. A multi-perspective analysis of the storm (peacomm) worm, 2007.

[10] Ramneek Puri. Bots & botnet: An overview. Technical report, SANS Institute, 2003.

[11] Brandon Shirley and Chad D. Mano. A model for covert botnet communication in a private subnet. In *Networking*, pages 624–632, 2008.

[12] Brandon Shirley and Chad D. Mano. Sub-botnet coordination using tokens in a switched network. In *GLOBECOM*, pages 2169–2173, 2008.

[13] Vrizlynn L. L. Thing, Morris Sloman, and Naranker Dulay. A survey of bots used for distributed denial of service attacks. In *SEC*, pages 229–240, 2007.

[14] Ke Wang, Gabriela Cretu, and Salvatore J. Stolfo. Anomalous payload-based worm detection and signature generation. pages 227–246, 2005.

[15] P. Wang, S. Sparks, and C. Zou. An advanced hybrid peer-to-peer botnet. *Dependable and Secure Computing, IEEE Transactions on*, PP(99):1–1, 2003.