Parallel Botnet Detection System by Using GPU

Che-Lun Hung
Dept. of Computer Science & Communication Engineering
Providence University
Taichung, Taiwan
clhung@pu.edu.tw

Hsiao-Hsi Wang
Dept. of Computer Science and Information Management
Providence University
Taichung, Taiwan
hhwang@pu.edu.tw

***Abstract*—In recent years, botnet is one of the major threats to network security. Many approaches have been proposed to detect botnets by comparing bot features. Usually, these approaches adopt traffic reduction strategy as first step to reduce the flow to following strategies by filtering packets. With the rapid development of network hardware and software the network speed has reached to multi-gigabit. However, analyzing header and payload of every packet consumes huge amount of computational resources and is not suitable for many realistic situations. Although signature-based solutions are accurate, it is not possible to detect bot variants in real-time. In this study, we proposed a GPU-based botnet detection approach. The experimental results show that the network traffic reduction stage on GPU can achieve about 8x times over CPU based botnet detection tool. The proposed algorithm can used to improve the performance of botnet detection tools efficiently.**

Keywords-Botnet; Bot; GPU; Parallel Computing; TCP; Network Security

I. INTRODUCTION

With the rapid development of the network hardware and software, the network speed is enhanced to multi-gigabit. A variety of Internet services, such as web search engines, entertainments, and others, have been provided to people. Therefore, Internet security has become an important role to protect activities on Internet. Botnet has become one major threat to Internet users in recent years. A botnet consists of a large number of bots that are networked computers compromised by malicious attackers. Usually, an attacker controls the bots to launch various types of attacks such as phishing and spamming with a botnet, and thereby receives benefits from a variety of aspects such as economy and social security. Therefore, detecting bots and preventing users from being infected is critical to network security experts and researchers.

Most of methods detect bot's activities based on predefined patterns and signatures retrieved from well-known bots [1, 2, 3, 4, 5, 6]. Snort [1] acts as a network sniffer and analyses network traffic in order to detect attacks and other types of undesired traffic. Snort checks packet headers and packet payload to detect bot by using pattern-matching techniques against a database of bot signatures. Although signature-based approaches are able to detect bots accurately, it is difficult to be applied to detect botnet in real time. Nowadays, signature detection becomes more and more complex because the number of attacks is more than before, and the signatures also become sophisticated. In addition, the amount of packets is dramatically increasing due to the rapid growth of network traffic. Therefore, single computing machine is not sufficient to analyze the entire traffic on a high-speed link. Consequently, random packet losses are likely to occur if the traffic exceeds the process capacity of the botnet detection system.

To solve packet loss problem leaded by large amounts of traffic, one of the solutions is to increase the processing capacity of the botnet detection system. It can be either hardware [7] or software [8] solutions. However, the cost of these solutions is huge. The reasonable cost of available computing power to analyze the network traffic is critical. Therefore, parallel process is useful solution to improve the performance of the detection system.

In recent years, General-Purpose Graphics Processing Unit (GPGPU) has been applied to solve computation intensive problem of various domains [9-15]. Especially, GPGPU programming is useful for the scientific computing domains that involve a high level of numeric computation. The greatest benefit of GPGPU is that the processing units grow from many (few cores) to massive (over hundred cords) to enhance the computation power significantly. In 2006, NVIDIA proposed the Compute Unified Device Architecture (CUDA). CUDA adopts a new computing architecture named Single Instruction Multiple Threads (*SIMT*) [16]. *SIMT* allows thread to execute many instructions, facilitating decision-based execution that is not provide for by the common-model, Single Instruction Multiple Data (SIMD).

To improve the computational performance of the Botnet detection system, we propose a parallel detection system to rapidly detect bots by leveraging GPGPU in this paper. The experiment results demonstrate that the performance of the proposed system is superior to the CPU-based network traffic reduction strategy over 8x times. It presents that GPGPU is useful to enhance the performance of detection system.

The structure of this paper is as follows. Section 2 describes the proposed method. Section 3 presents the experiment results. We conclude with section 4, providing a brief summary and conclusion.

978-1-4799-4860-4/14/$31.00 copyright 2014 IEEE
ICIS 2014, June 4-6, 2014, Taiyuan, China

II. RELATED WORK

Kolbitsch *et al.* [17] proposed a malware detection approach that is used to replace or complement traditional anti-virus software at the end host. They used behavior models to detect malicious program, and these models match flows of system call invocations. Although it can be used to detect variants of a malware program, it may fail on detection of unknown malware. The analysis results showed that the detection rate is only 64%.

Livadas *et al.* [18] developed a system to detect C&C traffic of IRC botnets by using machine learning-based classification techniques. The system contains two stages, distinguishing between IRC and non-IRC traffic and distinguishing between botnet and real IRC traffic. They use a Bayesian network classifier to make the classification balanced between false negative rates and false positive rates. Sadhan *et al.* [19] study the periodic behavior of C & C traffic, and then identify botnet C&C traffic by using this periodic behavior. The bots of experiments were from simulated bots, not bots in the real network. Therefore, the validness of the experiment results needs to be further verified by experience real world bots. Zeidanloo *et al.* [20] proposed a detection framework to detect P2P based and IRC based botnets. The difference between their framework from many other similar works is that there is no need for prior knowledge of Botnets such as Botnet signature.

This detection approach can identify both the C&C servers and infected hosts in the network. Our approach is based on the observa- tion that, because of the pre-programmed activities related to C&C, bots within the same botnet will likely demonstrate spatial-temporal correlation and similarity. For example, they engage in coordinated communication, propagation, and attack and fraudulent activities. Our prototype system, BotSniffer, can capture this spatial-temporal correlation in network traffic and utilize statistical algorithms to detect botnets with theoretical bounds on the false positive and false negative rates. We evaluated BotSniffer using many real-world network traces. The results show that BotSniffer can detect real-world botnets with high accuracy and has a very low false positive rate.

Choi *et al.* [21] proposed a botnet detection mechanism by monitoring of DNS traffic to identify botnets that are group activity in DNA queries sent from distributed bots. These bots can be grouped together by similarities of DNS requests and hence these group activities are able to be used to detect bots. Gu *et al.* [22] proposed an approach, ''Bot-Sniffer'', which uses network-based anomaly detection to identify bot hosts based on spatial–temporal correlation of collected network traces without any prior knowledge of signatures or C&C server addresses. Although the evaluation results showed their approach is able to achieve a high detection rate and low false positive rates, only one bot is used in the evaluation.

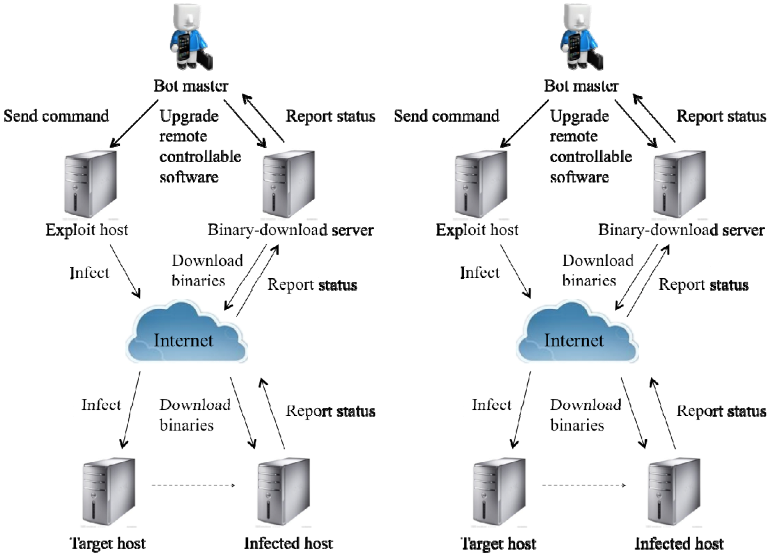
Park *et al.* [23] proposed an automated approach to generate semantic patterns for bot detection. This approach identifies one pattern that represents the important behavior of an entire class of bots, rather than of individual instances. In this approach, static analysis techniques are utilized to characterize bot behavior, and proposed to use hierarchical clustering of the resulting semantic patterns from a set of bot programs. It can achieve good detection rates more than 95%, but the effectiveness is limited when code obfuscation techniques are applied because their program is implemented by assembly code. In addition, signature-based detection limits its ability to detect bot variants and unknown bots.

Yu *et al.* [24] proposed online botnet detection based on an incremental discrete Fourier transform approach. They used "feature streams" to describe raw network traffic, and then the feature streams originated from different hosts are compared with the known feature streams. The suspicious bot activities hence are detected if they are similar to the known feature streams. Discrete Fourier Transform (DFT) technique is adopted to represent network traffic as feature streams and detecting bots. However, two shortcomings are existed in this work. First, representation of network traffic as feature streams and detection of using DFT is a computation-intensive work, and therefore it is difficult to detect bot activities in real time application. Second, the error rates (false positive rates) are not significant low by their experiment results.

A number of studies are intent of detecting and presenting specific type of bots. Perdisci *et al.* [25] proposed to detect bots that leverage HTTP channels to communicate. Hsu *et al.* [26] and Lin *et al.* [27] proposed a approach to detect a specific bot, called fast-flux bot, which intends to extend the life time of malicious web or Internet services with dynamic domain names of using short TTLs. It is possible that bots can evade detection when bot detection mechanism did not associate bots' communications with the corresponding hosts [28]. Huang [29] proposed a detection method to detect bots based on failures generated from bots because a bot never generate a failure and then it could be missed from detection.

Wang *et al.* [30] proposed a bot detection method based on fuzzy pattern recognition approach. In this solution, the network traffic is reduced by a traffic reduction algorithm to filter the packets that are not needed to be processed in the next steps. Several membership functions have included in the fuzzy pattern recognition techniques to compute the probability of being bot activities from aggregated DNS and TCP traffic in the proposed method. Although the proposed method is able to achieve high detection rates, the detection accuracy can be improved by adopting more sophisticated membership functions.

Wang *et al.* [31] proposed a solution that the detection is made based on external behaviors. Because they utilized behavior detection model, the code obfuscation does not prevent a bot from being detected and bot variants and event unknown bots can be detected. This approach adopts a fuzzy pattern recognition approach that included even more sophisticated membership functions than their pervious work



(a) Infection phases (b)Attack phases

Figure 1. A bot's activity behavior.

[30]. Therefore, the detection rates are improved and the error rates are reduced significantly. In addition, this approach adopted concept of parallel computing to dispatch detection tasks to multiple servers and detect bots in parallel. The efficiency of the detector can be improved in proportion to the number of allocated detection servers.

III. Method

Usually, botnet has two common phases of bot behavior, infection and attack phases. Figure 1 shows these two phases. In the infection phase, a bot master attempts to intrude in a node as a victim and this victim then will become a bot. When a victim is intruded by the bot master, the remote controllable software (bot software) is downloaded and installed in the victim. After the bot software has been successfully installed and configured in the victim, the infection phase is finished and the second phase, the attack phase, then starts. In the attack phase, the bot software reports the status of the infected host to the bot master, receives commands from the bot master, and executes the commands. These commands include launching distributed denial of services, setting up phishing sites, relaying malicious traffic, and sending spam mails. Actually, behavior-based approaches can detect bots in the infection phase, the attack phase, or both phases, depending on the detection strategy adopted by the algorithms.

Most of the proposed botnet detection approaches has a very important stage, traffic reduction, to reduce the data set to the meaningful subset of flows to speed later stages in these approaches. However, these approaches take longer to filter a packet set captured off a giga-bit network interface than it took the set to arrive, making them infeasible for real-time traffic reduction. Therefore, we integrate GPU based packet filtering approach [13] into botnet detection tool [30] to enhance the performance of the tool. Figure 2 shows the proposed architecture.

A. *Botnet Detection*

Wang et al. [30] proposed a behavior-based botnet detection in parallel is to detect bot activities from the network packet trace. Their approach splits the process of the trace into five stages, traffic reduction, feature extraction, data partitioning, DNS detection, and TCP detection. The first two stages are used to reduce the amount of packets that has to be processed by remaining stages. In this approach, it adopted a parallel detection process that is to dispatch workloads to multiple detectors in the third stage. In the last two stages, it detects the bots that generate DNS queries and TCP requests.

Although Wang's approach adopted parallel computing to enhance the performance of their approach, the parallel computing part in their approach still can be improved. In this paper, we propose a parallel botnet detection approach based on Wang's approach by adopting GPU. Figure 2 illustrates the proposed architecture. In the first stage, the packet is filtered by GPU-Based packet filtering mechanism. The second stage is to extract botnet features. The third stage is to detect botnet in parallel by GPU.

B. *GPU-based Botnet Detection*

With rapid growth of GPU hardware architecture, GPU has become increasingly more powerful and ubiquitous. Therefore, many computation intensive applications have begun developed on GPU. The GPUs parallel mode is that the threads execute the same instructions on multiple data simultaneously. Both of graphical and general-purpose applications thus are faced with parallelization challenges on GPU.

NVIDIA proposed the Compute Unified Device Architecture (CUDA) SDK to assist developers to create general-purpose applications that run on GPUs. A CUDA programs typically consist of a component that runs on the host (CPU), and a computationally intensive component called the kernel that runs in parallel on the GPU

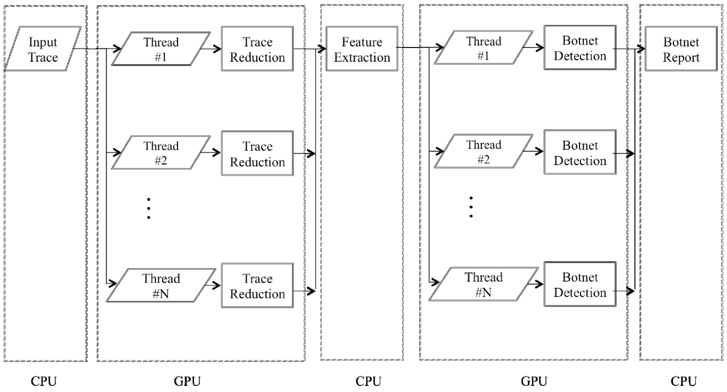


Figure 2. The architecture of the proposed botnet detection on GPU.

[16]. Input data for the kernel must be copied to the GPU's on-board memory from host's main memory through the PCI-E bus prior to invoking the kernel, and output data also should be written to the GPU's memory first and then copy to main memory. All memory used by the kernel should be pre-allocated.

Kernel executes a collection of threads that computes a result for a small segment of data. The hierarchy model for multiple threads is grid, thread block and thread from top to bottom. A grid is composed of multiple thread blocks and it is usually a one or two dimension grid. A given thread block is composed of threads, and this given block can be positioned within the grid. Therefore, each thread can calculate the data that it own, and write the output to memory on GPU. Each block is executed by a single multiprocessor, which allows all threads within the block to communicate through shared memory shared by the threads in the same block. There are three stages in the proposed architecture, traffic reduction, feature extraction, and botnet detection.

Figure 3 illustrates the architecture of GPU-based network traffic reduction stage to reduce the number of packets that the detection system has to examine. The packet filtering patterns are stored in constant memory and register files, and the packets are stored in global memory. A thread process a packet data according the filtering pattern. A bot's activities usually start from DNS lookups, because it often has to receive new commands from a bot master or command and control (C&C) servers. All of these servers have a list of domain names in bot software. The bot attempts to interact with each of the returned IP addresses from DNS queries in the bot master's or the C&C servers' IP addresses list. The most bots interact with the bot master or the C&C servers using TCP connections. Therefore, this stage is to examine TCP packet and remove DNS request/response packet or a packet with known source/destination IP addresses in the filtering IP list. The steps of GPU-based traffic reduction stage are shown as below:

- Coping packet header to GPU device memory
  The IP list is stored in constant memory, and the packet data is stored in global memory.
- Checking the packet headers
  A thread process a packet data according the IP list.
- Copy packet headers to main memory
  The results are copied back to main memory.
- Filtering packet
  The packet has been indicated from suspect IP list, and it will not be passed to next stage.

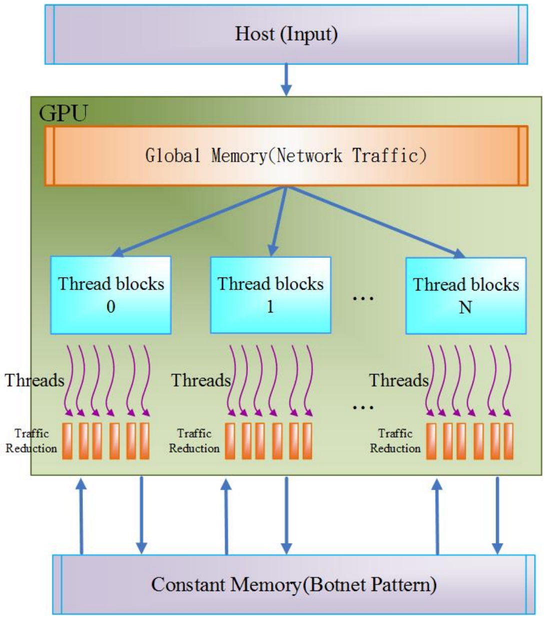


Figure 3. Architecture of GPU-based network graffic reduction stage.

In the feature extraction stage, two features, TCP and DNS, are observed in [31]. For the DNS feature, a bot often sends DNS queries regularly in a period of time. For TCP feature, For TCP features, we also observed that a bot would setup regular network flows to the bot master or the C&C server. The packets are collected for a period of time, and then both of these two features are extracted from these packets.

In the botnet detection stage, two phases, TCP and DNS, are utilized to detect bots. These two phases adopt the features extracted in feature extraction stage to detect bots. In the proposed method, a thread on GPU examines a packet according to TCP and DNS features. The steps of GPU-based botnet detection stage are shown as below:

- Coping TCP and DNS features to GPU device memory
  The TCP and DNS features are stored in constant memory, and the packet data is stored in global memory.
- Checking the packet data
  A thread processes a packet data according the features.
- Copy packet headers to main memory
  The results are copied back to main memory.
- Reporting bot
  If a host is detected as a bot, then it is reported.

IV. EXPERIMENT

In the proposed approach, we have implemented GPU-Based network traffic reduction stage on NVIDIA GeForceGTS 450 graphics card (Fermi architecture) and installed in a PC with an Intel i3 540 3.2 GHz CPU and 8GB DDRIII-1333 RAM running the Linux operating system. The testing packet data set is 65 million packets with the random source address, destination address, source port, destination address and protocol for the experiments.

The proposed approach is implemented in four different versions. In the first version, packet data is stored in global memory and IP list is stored in constant memory. In the second version, packet data is stored in texture memory and IP list is stored in constant memory. The third version is the same as the first version in memory usage but the data transfer way is zerocopy and the fourth version is the same as the second version in memory usage but the data transfer way is zerocopy. The results show that the GPU-based RFC algorithm can achieve 8x~13x speedup over CPU-based RFC algorithm in Fig. 4.

The performance of the first stage is enhanced dramatically. It can be expected that the total performance of the proposed approach can be improved significantly by using GPU parallel model.

V. CONCLUSION

In Recent years, botnet has become one major threat to Internet users. Meanwhile, many botnet detection approaches are proposed to detect bots. Most of these methods detect bot's activities based on predefined patterns and signatures retrieved from well-known bots. Nowadays, signature detection becomes more and more complex than before because the signatures become sophisticated. In addition, the amount of packets is dramatically increasing due to the rapid growth of network traffic. Therefore, single computing machine is not sufficient to analyze the entire traffic on a high-speed link. Consequently, random packet losses are likely to occur if the traffic exceeds the process capacity of the botnet detection system.

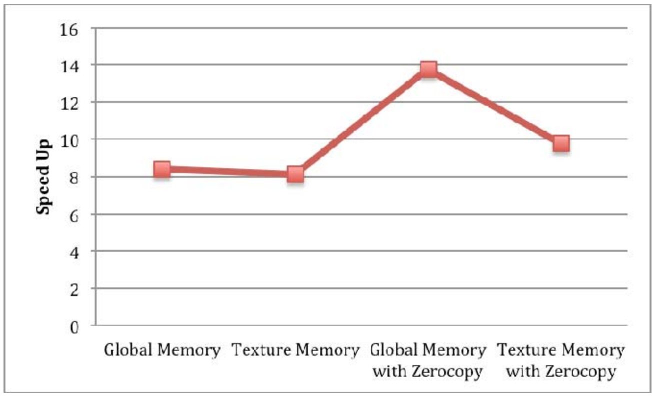


Figure 4. The performance comparison between GPU-based and CPU-based network traffic reduction stages.

To improve the computational performance of the Botnet detection system, we propose a parallel detection system to rapidly detect bots by leveraging GPGPU. In the paper, we only implement network traffic reduction stage on GPU. The experiment results demonstrate that the performance of the proposed system is superior to the CPU-based network traffic reduction over 8x times. It presents that GPGPU is useful to enhance the performance of detection system. In the future, we will implement other stage on GPU, to improve the performance of the proposed algorithm in advance.

ACKNOWLEDGMENT

This research was supported by the National Science Council under the Grants NSC-102-2221-E-126-004.

REFERENCES

[1] M. Roesch, "Snort - Lightweight intrusion detection for networks," Proceedings of the 13th USENIX Conference on System Administration, 1999.

[2] V. Paxson, "Bro: A system for detecting network intruders in real-time," Computer Networks, vol. 31(, pp. 2435–2463, 1999.

[3] W. Lu, M. Tavallaee, G. Rammidi and A. A. Ghorbani, "Bot- cop: An online botnet traffic classifier," Proceedings of the 7th IEEE Annual Communications Networks and Services Research Conference, 2009.

[4] F. Alserhani, M. Akhlaq, I. U. Awan and A. J. Cullen, "Detec- tion of coordinated attacks using alert correlation model," Proceedings of IEEE International Conference on Progress in Informatics and Computing, 2010.

[5] M. Szymczyk, "Detecting botnets in computer net- works using multi-agent technology," Proceedings of the 4th International Conference on Dependability of Computer Systems, 2009.

[6] L. Braun, G. Munz and G. Carle, "Packet sampling for worm and botnet detection in TCP connections," Proceedings of IEEE Network Operations and Management Symposium (NOMS), 2010.

[7] S. Fide and S. Jenks, "A Survey of String Matching Approaches in Hardware," Dept. of Electrical Engineering and Computer Science, University of California, Irvine, Tech. Rep. TR SPDS 06-01, Mar. 2006.

[8] M. Colajanni and M. Marchetti, "A Parallel Architecture for Stateful Intrusion Detection in High Traffic Networks," in Proc. of Workshop on Monitoring, Attack Detection and Mitigation (MonAM) 2006, Tu □bingen, Germany, Sep. 2006.

[9] Y. C. Chen and J. S. Yeh, "Preference utility mining of web navigation patterns," Proc. IET international conference on Frontier Computing, pp. 49–54, 2010.

[10] C. Y. Lin, C. L. Hung and Y. C. Hu, “A Re-sequencing Tool for High-throughput Long Reads Based on UNImarker with Non-Overlapping iNterval Indexing Strategy,” INFORMATION-An International Interdisciplinary Journal, vol.16, pp. 827–832, 2013.

[11] Y. R. Chen, C. L. Hung, Y. S. Lin, C. Y. Lin, T. L. Lee and K. Z. Lee, "Parallel UPGMA Algorithm on Graphics Processing Units Using CUDA," , Proc. IEEE International Conference on High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems, pp. 849–854, 2012.

[12] C. L. Hung, C. Y. Lin and H. H. Wang, "An efficient parallel-network packet pattern-matching approach using GPUs," , Journal of Systems Architecture, 2014.

[13] C. L. Hung, Y. L. Lin, K. C. Li, H. H. Wang and S. W. Guo, "Efficient GPGPU-Based Parallel Packet Classification," Proc. IEEE International Conference on Trust, Security and Privacy in Computing and Communications, pp. 1367–1374, 2011.

[14] C. Y. Lin, C. Y. Tang, S. T. Li, Y. L Lin, and C. L. Hung, "CUDA-FRESCO: Frequency-Based RE-Sequencing Tool Based on CO-clustering Segmentation by GPU," Proc. IEEE International Conference on High Performance Computing and Communications, pp. 857–862, 2011.

[15] M. LCharalambous, P. Trancoso and A. Stamatakis, “Initial Experiences Porting a Bioinformatics Application to a Graphics Processor,” Proc. the 10th Panhellenic Conference on Informatics, LNCS, 415–425, 2005.

[16] Nvidia cuda c best practices guide, version 4., March 2011.

[17] C. Kolbitsch, P. M. Comparetti, C. Kruegel, E. Kirda, X. Zhou and X. Wang, “Effective and efficient malware detection at the end host,” Proceedings of 18th USENIX Security Symposium, USENIX Association, pp. 351–366, 2009.

[18] C. Livadas, R. Walsh, D. Lapsley and W. T. Strayer, “Usilng machine learning technliques to identify botnet traffic,” Proceedings of the 31st IEEE Conference on Local Computer Networks, pp. 967–974, 2006.

[19] B. Sadhan, J. M. F. Moura and D. Lapsley, “Periodic behavior in botnet command and control channels traffic,” Proceedings of the 28th IEEE Conference on Global Telecommunications, pp. 2157–2162, 2009.

[20] H. R. Zeidanloo and A. B. Manaf, “Botnet Detection by Monitoring Similar Communication Patterns,” International Journal of Computer Science and Information Security, Vol. 7, pp. 36-45, 2010.

[21] H. Choi, H. Lee, H. Lee and H. Kim, “Botnet detection by monitoring group activities in DNS traffic,” Proceedings of the 7th IEEE International Conference on Computer and Information Technology, pp. 715–720, 2007.

[22] G. Gu, J. Zhang and W. Lee, “Botsniffer: Detecting botnet command and control channels in network traffic,” Proceedings of Network and Distributed System Security Symposium, 2008.

[23] Y. Park, Q. Zhang, D. Reeves D, Mulukutla, “Antibot: Clustering common semantic patterns for bot detection,” Proceedings of the 34th IEEE Annual Computer Software and Applications Conference, 2010.

[24] X. Yu,X. Dong, G. Yu,Y. Qin,D. Yue and Y. Zhao, “Online botnet detection based on incremental discrete fourier transform,” Journal of Networks, vol. 5, pp. 568– 576, 2010.

[25] R. Perdisci, D. Ariu and G. Giacinto, “Scalable fine- grained behavioral clustering of http-based malware,” Computer Networks, vol. 57, pp. 487–500, 2013.

[26] C. H. Hsu, C. Y. Huang, K. T. Chen, “Fast-flux bot detection in real time,” The 13th International Symposium on Recent Advances in Intrusion Detection, 2010.

[27] H. T. Lin, Y. Y. Lin and J. W. Chiang, “Genetic-based real-time fast-flux service networks detection,” Computer Networks, vol. 57, pp. 501–513, 2013.

[28] B. Shirley, L. Babu and C. Mano, “Bot detection evasion: a case study on local-host alert correlation bot detection methods,”

[29] C. Y. Huang, “Effective bot host detection based on network failure models,” Computer Networks, vol. 57, pp. 514–525, 2013.

[30] K. Wang, C. Y. Huang, S. J. Lin and Y. D. Lin, “A fuzzy pattern-based filtering algorithm for botnet detection,” Computer Networks, vol. 55, pp. 3275– 3286, 2011.

[31] K. Wang, C. Y. Hung, L. Y. Tsai and Y. D. Lin, “Behavior-based Botnet Detection in Parallel,” SECURITY AND COMMUNICATION NETWORKS, 2013.