Efficient Detection of Real-World Botnets' Command and Control Channels Traffic

[1]Abdulmuneem Bashaiwth, [1]Basil AsSadhan, [2]Jalal Al-Muhtadi, and [1,3]Saleh Alshebeili
[1]Deptartment of Electrical Engineering, King Saud University, Saudi Arabia
[2]Deptarpment of Computer Science, King Saud University, Saudi Arabia
[3]KACST-TIC in RF and Photonics for the e-Society (RFTONICS), King Saud University
Riyadh, Saudi Arabia
basha-13@live.com, bsadhan@ksu.edu.sa, jalal@ksu.edu.sa, and dsaleh@ksu.edu.sa

ABSTRACT: *Botnets are a major security threat to today's Internet. Therefore, the detection of botnets has become a central task for network administrators. In this paper, we study the detection of botnets by monitoring and analyzing the Command and Control (C2) channels communication traffic. We note that this detection approach is effective as it detects a botnet before it engages in any harmful activities.*

We analyze real network traffic captured at King Saud University network by exploiting the periodic behavior in C2 traffic. We use periodograms to study the periodic behavior, and apply Walker's large sample test to detect whether the traffic has a significant periodic component or not, and, if it does, then it is bot traffic. We apply this test on two different days of KSU traffic. We show that the traffic in those days exhibit periodic behavior and report the source of that traffic as bot.

Keywords: Botnet detection, C2 traffic, Periodogram, Crypto-PAn Walker's large sample test.

Received: 2 August 2014, Revised 10 September 2014, Accepted 21 September 2014

1. Introduction

A botnet is a group of compromised computers (bots) that are controlled remotely by a single entity called a botmaster. Command and control (C2) communication channels are used by the botmaster to command and control bots in order to download the required information and codes for attacks execution [1]. Botnets are currently one of the major security threats that Internet users face. They are used to execute various malicious activities such as identity spoofing, password guessing, eavesdropping, DNS poisoning, Distributed Denial of Service (DDoS) attacks, E-mail spam, and phishing. Consequently, the detection of botnets has become an aim for network security administrators.

Botnets can be classified into two types, commonly called first and new generations [2, 3]. The first generation uses the Internet Relay Chat (IRC) service for its C2 communication channel. This type is also referred to as centralized botnets. The centralized C2 mechanism of such botnet has made it vulnerable to being detected and disabled because it has a single point of failure. The new generation of botnets are called Peer-to-Peer (P2P) based botnets. The P2P botnet does not suffer from a single point of failure, as it does not have a centralized C2 server. Instead, botnet's members contact each other through a mesh topology [3].

Regardless of the different structures and communication protocols used in many botnet variants, bots within a single botnet frequently contact each other through C2 communication channels every *T* seconds to receive commands, update data, and send keep-alive messages. Due to this behavior, these bots demonstrate similar traffic activities which result in temporal-spatial correlation[4]. Therefore, and as result of the pre-programmed manner in bots, periodic behavior arises in botnet C2 channels traffic.

Recent researches in botnet detection propose different techniques to achieve acceptable detection results. These techniques can be classified into two essential techniques; honeynets-based detection and passive network traffic monitoring-based detection [5, 6]. Honeynet-based techniques have been used to understand botnets' characteristics more than their detection. On the other hand, passive network traffic monitoring is helpful to identify the existence of botnets [6]. Network traffic monitoring allows the traffic at particular points in a network to be recorded, displayed in a useful format, and hence analyzed appropriately. In our work, we adopt traffic monitoring through the analysis of network traffic behavior.

Analysis of network traffic behavior is an effective method for botnet detection. It is performed by monitoring network traffic and noting unusual actions or departures from normal operation rather than the focus on packets' content, which we may or may not have access to, and even if we did, it might be encrypted. Furthermore, analysis of network traffic behavior allows the observation of a large amount of traffic with less processing time when compared to deep packet inspection [7].

Yu et al., [8], presented a method to detect botnets based on similarity measurement. This is achieved by computing the average Euclidean distance between streams of host features. Once few feature streams exhibit high similarity in their activities, the corresponding hosts are regarded as suspected bots. The authors extracted several features to construct the feature stream, example of these features include; total packet exchanged in flow, average bits per second for flow, and flow duration. In their work, they used the Discrete Fourier Transform (DFT) to avoid huge calculation among feature streams. Arshad et al., [9], followed a similar approach to the one used by Yu's, where they measured the similarity between NetFlows, [10], to detect botnets. They monitored the behavior of NetFlows and attacks simultaneously. To identify bot infected hosts, they clustered bots with similar NetFlows and attacks in different time windows, and then performed correlation techniques. The method is applied to real-world packet traces including normal traffic and several real-world botnet traces that included IRC-SdBot, IRC-SpyBot, HTTPBot-I, and HTTP-Bot-II.

To detect IRC botnets' C2 traffic and distinguish it from IRC chat traffic, Ma et al., [11], proposed to analyze the characteristic of packet size sequences of the TCP conversation held between IRC bots and their C2 servers. They found that the TCP conversations within IRC botnets show a nature of approximate (quasi) periodicity, whereas the ones in IRC chat do not show periodicity. On the other hand, AsSadhan et al., [12], studied the periodic behavior in C2 traffic, they evaluated the periodogram of the traffic, then applied Walker's large sample test to detect whether the traffic has a significant periodic component or not.

All of the reviewed techniques share the following advantages; 1) They are independent of the structure and communication protocol used in the botnet, 2) They look for periodic behavior or similar activities in botnet traffic, 3) There is no need for any other a priori knowledge of the botnet behavior. The drawback of these techniques is that a botmaster may attempt to evade the detection of its C2 traffic periodic behavior by randomizing the update times. However, such evasion scheme will limit the efficiency of the exchange of C2 channel traffic between bots. As a result, a bot might not have the needed C2 updates on time, which might disturb the effectiveness of the attack carried out by the botnet.

The rest of this paper is organized as follows; in Section II we give an overview of KSU's network dataset and how we captured it. In Section III, we describe the architecture of our approach in botnet detection. Section IV presents and evaluates the results of the approach on various network traces. We present our conclusions in Section V.

2. Network Traffic Capturing

The first step to test a network traffic behavior analysis tool is to obtain packet traces. There are several approaches to obtain packet traces, and these traces differ in their characteristics depending on the approach used in obtaining them. Packet traces can be obtained by; 1) Generating traffic in an isolated experimental network environment (e.g., [13]); 2) Simulating network traffic (e.g.,[14]); 3) Capturing traffic from real world networks (e.g.,[15]). Both generating traffic in an isolated experimental network environment and simulating it provide packet traces that contain header and payload information. Network traffic generated in an isolated network is a better representation of real world network traffic than simulated traffic. However, it is not easy to either simulate or generate network traffic with all of the network applications that real world networks have [16]. Therefore, generated network traffic is only a good representation of individual applications traffic.

Using previously captured packet traces becomes an issue after few years pass since their capture as they become old and outdated. Tavallaee et al., [17], reviewed the state of experimental practices in the area of anomaly-based intrusion detection and surveyed 276 studies, containing 61 journals and 215 conference/workshop papers that were published during the period of 2000-2008. Based on their findings, it has been observed that most of the published work has been tested and validated using outdated datasets whose accuracy and ability to reflect current real-life conditions are questionable. Therefore, to avoid using outdated datasets we resort to capturing recent packet traces from King Saud University's (KSU) network.

To execute the capturing, we used Endace DAG 7.5G2 card [18]. The Endace DAG 7.5G2 card is used to capture traffic at the full line rate from the network into the memory of the host computer with zero packet loss. The DAG 7.5G2 card1 operates on a 4 lane PCIe bus and can be installed in any free 4 lane PCIe slot [19]. KSU network traffic was captured from one of the routers in KSU's network by mirroring the traffic of one of its ports into DAG7.5G2 card port through the Switched Port Analyzer (SPAN) of router. More than 11TB worth of traffic was captured using DAG 7.5G2 in native Endace Record Format (ERF) to cover all hours of the day for 50 days starting from Dec. 22, 2012 until Feb. 9, 2013. Approximately, more than 10,000 hosts were active inside KSU network during the capturing process.

3. Architecture Of Botnet Detection Approach

Figure 1 shows the architecture of our botnets detection approach. First, raw traffic is captured from the network's router as explained in SectionII. Then, packet traces are filtered out to reduce the volume of processing data; simultaneously the anonymization technique is performed on IP addresses of the filtered packets to maintain the privacy of the users. After that, packets are aggregated over appropriate time intervals and several count-features sequences, [12], are extracted within the aggregated intervals to produce discrete time sequences. Finally, statistical signal processing techniques are applied to the discrete time sequences to look for periodic behavior for the purpose of botnet detection. We next elaborate on these steps.

3.1 Anonymization of Packet Traces

Network administrators are typically unwilling to release their packet traces to the public due to the concern that user private information may be deduced from the trace. To make the traces publicly available, anonymization process has to be applied and only packets' header information can be released. In our work, we anonymize the IP addresses of KSU packet traces, where each distinct IP address appearing in the original trace is mapped to a distinct random address, thus the mapping process is performed one-to-one. The address anonymization technique is prefix-preserving, where two IP addresses sharing an n-bits prefix in the original IP address space will also share a n-bit prefix in the anonymized IP address space [20]. We use the *Cryptography based Prefix preserving Anonymization* (Crypto-PAn) tool to perform IP address anonymization. The Crypto-PAn tool is used to perform prefix-preserving anonymization, which uses the IP::Anonymous built-in Perl's module [21]. One of the important characteristics of Crypto-PAn is that it uses cryptography techniques that allows the owner of the traces to use a secret key to keep the anonymization process secret.

3.2 Preprocessing of Packet Traces

We use Perl, [22], to convert captured packets into Comma Separated Value (CSV) files. We start by recognizing the timestamp and data length fields between packets. Then we read the Ethernet frame header for each packet, where, we can recognize IP packets. After that, by parsing the IP header, we were able to get TCP packets. We focus on TCP packets since they constitute 75 – 85% of Internet's traffic. At KSU's dataset, TCP packets represent more than 90% of IP packets. For each TCP packet, we extract the following information and store them in CSV files:

- Time stamp

[1] Endace card works on Linux, FreeBSD, Windows Server 2003/2008, and Windows 7- 64 bit operating systems

- Source IP address
- Source port
- Destination IP address
- Destination port
- TCP flag (SYN, FIN, RST, or no flag is set)
- Data sequence number of the packet
- Data sequence number of the data expected in return
- Acknowledgment sequence number
- Receiver window
- Total length of the frame

We use the first five of the above features to produce discrete-time sequences. This is performed by aggregating TCP packets over an appropriate aggregation interval. Then count-features sequences (packet, byte, address, and port counts) are extracted from TCP packet header information. Statistical signal processing techniques can then be applied to discrete-time sequences for the purpose of botnet detection. The rest of above features will be used in our future work.

3.3 Methodology of Botnet Detection

We detect bots in network traffic through the detection of periodic behavior in the network traffic. This is done by analyzing the Power Spectral Density (PSD) of count-feature sequences extracted from the traffic. One of the most used tools to estimate PSD of signals are Periodograms [23]. The frequency components of a periodic signal exhibit high level of power at its fundamental frequency when its periodogram is calculated. Therefore, the periodogram of a periodic signal will have a high peak located at the fundamental frequency of the signal when compared to the mean of the periodogram.

After evaluating the periodogram and locating its peak, we test the significance of the peak compared to the mean of the periodogram. We use binary hypothesis testing, [24], and set up the null hypothesis H_0: the count-feature sequence is Gaussian against the alternative hypothesis H_1: the sequence has a periodic component at some unspecified frequency [12]. By using Walker's large sample test, [25], AsSadhan et. al, [12], sets the following statistic to test the peak's significance

$$g^*_x = \frac{\max_{0 \le k \le m-1}(P_{xx}[k])}{\frac{1}{2m}\sum_{k=0}^{m-1} (P_{xx}[k]}, \qquad (1)$$

where $P_{xx}[k]$ is the estimated periodogram of the traffic sequences, and m is the number of the periodogram ordinates at the positive frequencies.

Asymptotically, under H_0, we have for $z \ge 0$:

$$Pr\,[\,g^*_x > z\,] \sim (1 - exp\,(-z\,/\,2\,))^m\,. \qquad (2)$$

Under H_1, where the signal is periodic, the ratio g^*_x will be large. Therefore, we can use a one sided test and select the critical region $g^*_x > z_\alpha$, where z_α is selected so that the right hand side of (2) is equal to α, which is the false positive probability of the test, and the value of z_α would be :

$$z_\alpha = -2\ln\,(1 - \,(1 - \,a)^{1/m}). \qquad (3)$$

The value of α is selected based on how small we want the probability of false positive to be.

We reject H_0 if g^*_x is larger than z_α, and conclude that the sequence has a periodic component with a false positive probability of α, and that the periodic behavior is due to bot C2 traffic. If g^*_x is less than z_α we accept H_0 and conclude that the sequence does not have a periodic component.

4. Results and Analysis

In addition to the C2 traffic generated by a given bot, the bot will have other background traffic generated by the legitimate

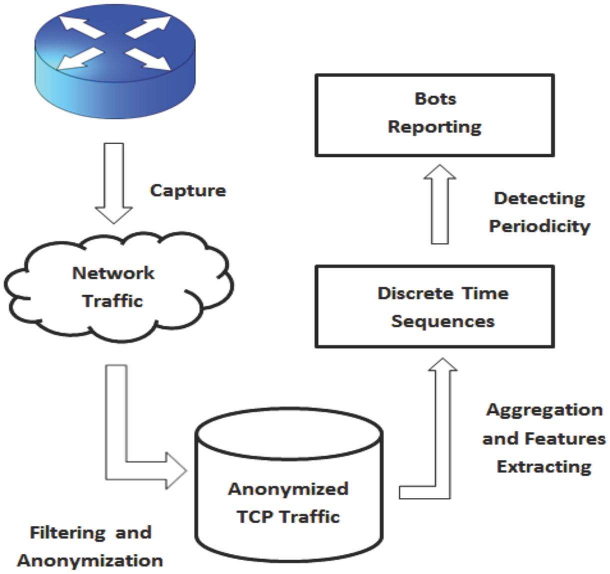

Figure 1. Architecture of botnet detection approach

user of the compromised machine. The background traffic can dominate the bot's traffic, especially if its volume is high enough and/or the C2 traffic has a long period or a small duty cycle [12]. This might result in hiding the periodic behavior of the C2 traffic; thus failing to detect the bot.

An alternative would be to filter out the traffic of every port number for a given host, then examine the traffic behavior of
iodic behavior or not. However, this is not scalable for a network that has many
ng the traffic of a given port number at every host, we test the port traffic for all
orm this on selected port numbers that are suspicious to be used by botmasters.

cket traces using port number 6667. We analyze the IRC traffic that originates from
ours period starting at 6AM on a given day until 6AM on the next day.

nd byte count sequences of IRC traffic for the packet traces captured on Thursday
0 seconds is used to extract the two count sequences from IRC traffic. The bottom
am for each sequence after subtracting its mean and normalizing it by its standard
ant peak located at 0.34 mHz, which corresponds to a period of 49 minutes. The
, which represent the signal's harmonics.

bility; thus the value of the threshold $z_1\%$ in (3) is equal to 23^2. We test the
x ($P_{xx}[k]$, for the two sequences by evaluating the ratio $g*_x$ in (1). We find it to be
arger than $z_1\%$, we conclude that the traffic exhibits periodic behavior and is due
plots in Figure 2 show that the C2 traffic is active for a very short time during the
period duration result in a very low duty cycle. This is because the IRC C2 traffic exchange involves very few packet that only represent the traffic originating from KSU's hosts and the return C2 traffic does not appear. This might be due to a firewall configuration that blocks such traffic.

The ratio $g*_x$ equals 44 for the periodograms of both of the address and port count sequences shown in Figure 3 for the same traffic. We note that the peaks of the periodogram of the address and port count sequences are greater than those of the packet and byte count sequences. This is because the number of distinct addresses or port numbers in the traffic has fewer fluctuations when compared to the number of packets or bytes.

[2] The number of ordinates at the positive frequencies of periodogram, m, used in evaluating $z_1\%$ is 1024.

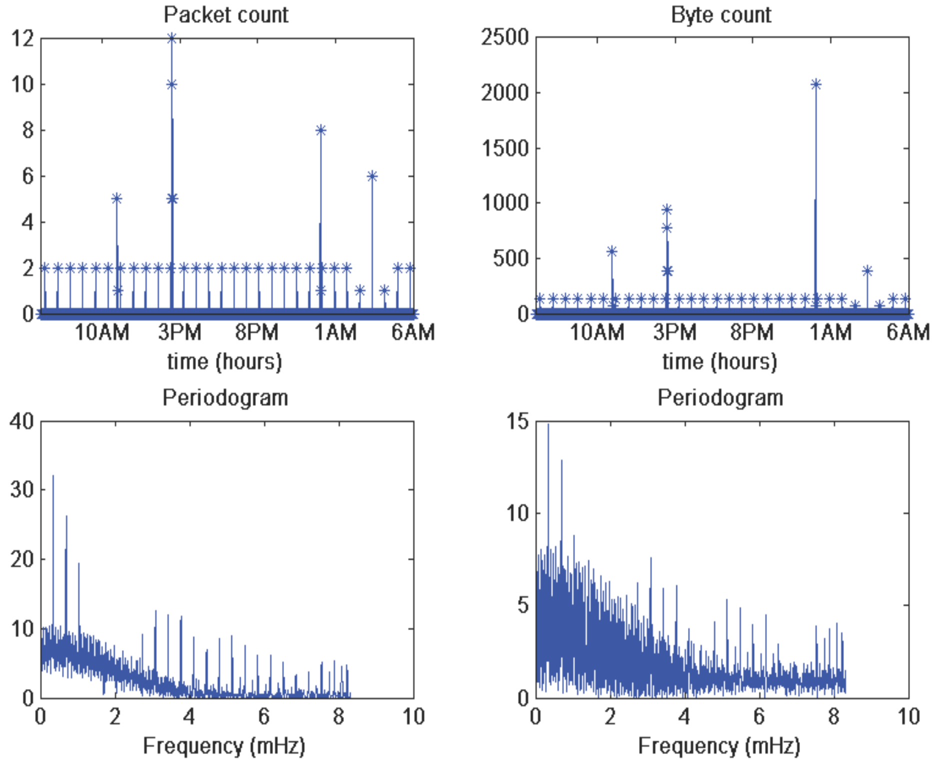

Figure 2. Left plots show the packet count for IRC traffic and its one sided periodogram. Right plots show the byte count for the same traffic and its one sided periodogram. The aggregation interval for both count sequences is 60 s.

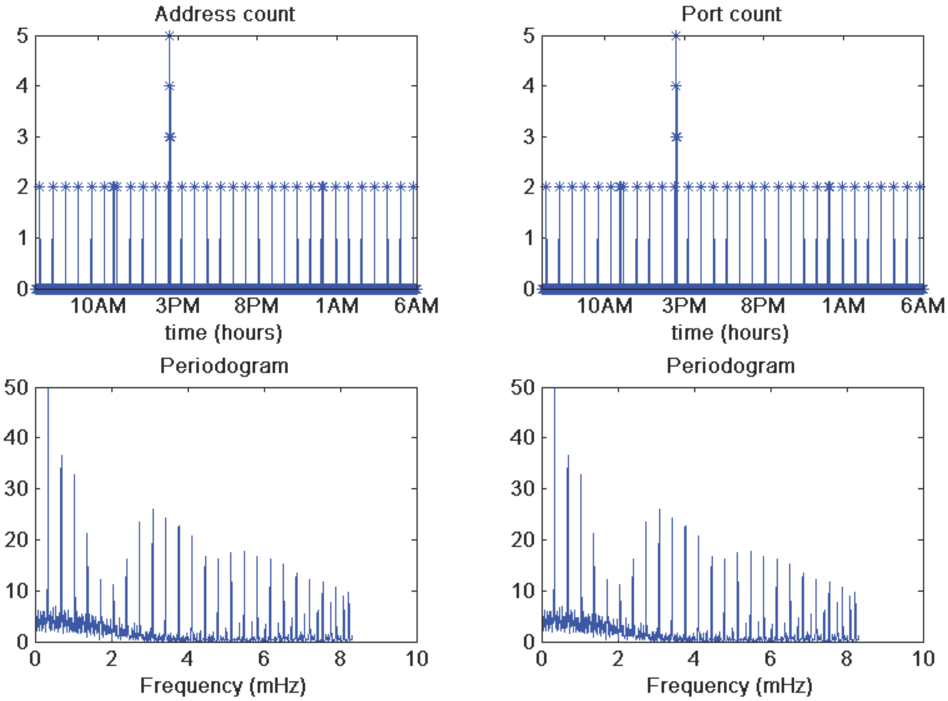

Figure 3. Left plots show the address count for IRC traffic and its one sided periodogram. Right plots show the port count for the same traffic and its one sided periodogram. The aggregation interval for both count sequences is 60 s

We can also see from Figures 2 and 3 that byte-count sequence follows packet-count sequence, in addition, the portcount is similar to address-count sequence. Therefore, in the rest of our results, we will concentrate only on packet and address counts sequences.

Next we identify which KSU host(s) was responsible for the periodic behavior. This is done by searching through all KSU's hosts and observing the number of transferred packet on port 6667. The host(s) that generates a significant amount of packets when compared to the number of IRC packets treated as suspicious, and their traffic is further analyzed.

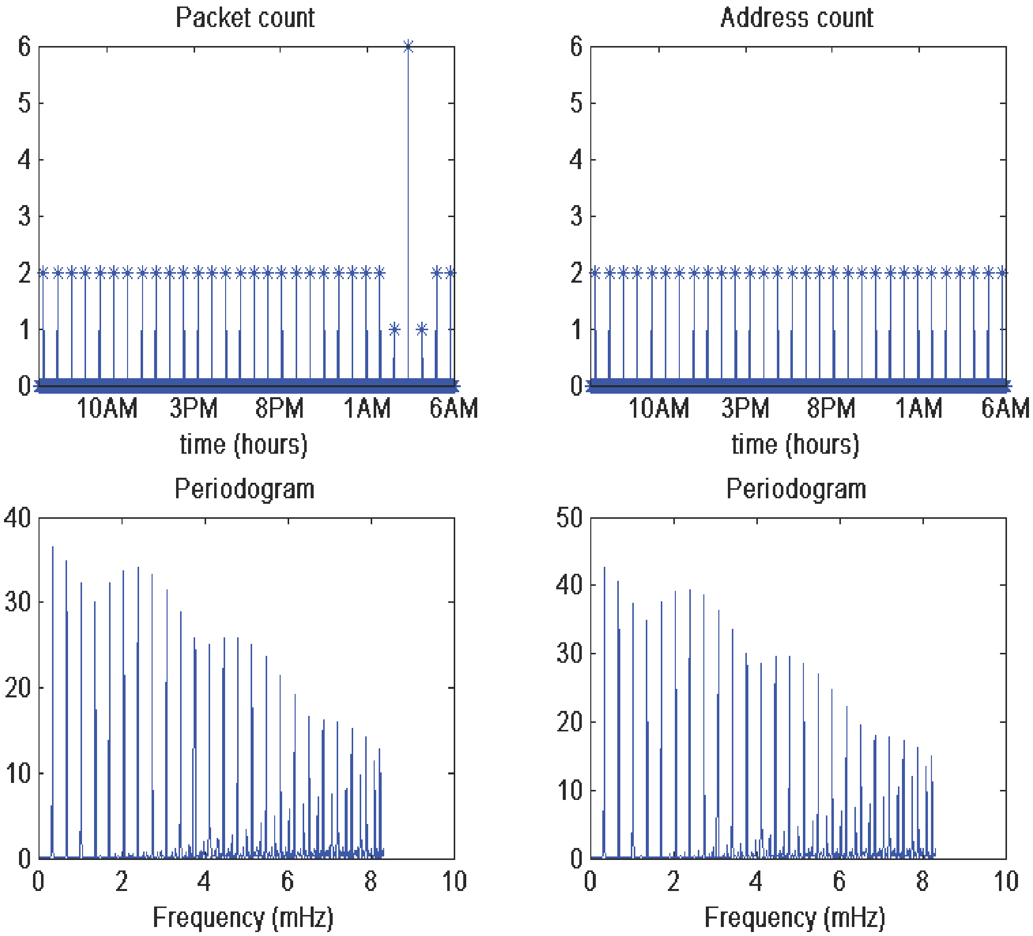

Figure 4: Left plots show the packet count for IRC traffic from a single host and its one sided periodogram. Right plots show the address count for the same traffic and its one sided periodogram. The aggregation interval for both count sequences is 60 s

Figure 4 shows the packet and address count sequences of a suspected host's IRC traffic and their periodogram at the same day. We use the same aggregation interval of 60 seconds and extract the packet and address count sequences within this aggregation interval. We can see a significant peak located at the same frequency in Figures 2 and 3, which corresponds to a period of 49 minutes. The ratio $g*_x$ is 45 for both packet and address count sequences. We notice that this value (in the case of testing the IRC traffic of single suspected host) is higher than the values of the test ratio we get in case of testing the IRC traffic of all internal hosts in the network (25 and 44 for packet and address count sequences, respectively). This is due to the presence of legitimate IRC traffic mixed with the bot's C2 traffic in case of testing the whole IRC traffic. Since this legitimate traffic does not exhibits periodic behavior, it lowers the value of the periodogram's fundamental peak.

While testing the port traffic for all internal hosts in KSU's network, we also find that the suspected KSU's host exhibits periodic behavior on port number 6970. Port 6970 may be used by Trojans as backdoor to allow a remote user access the host.

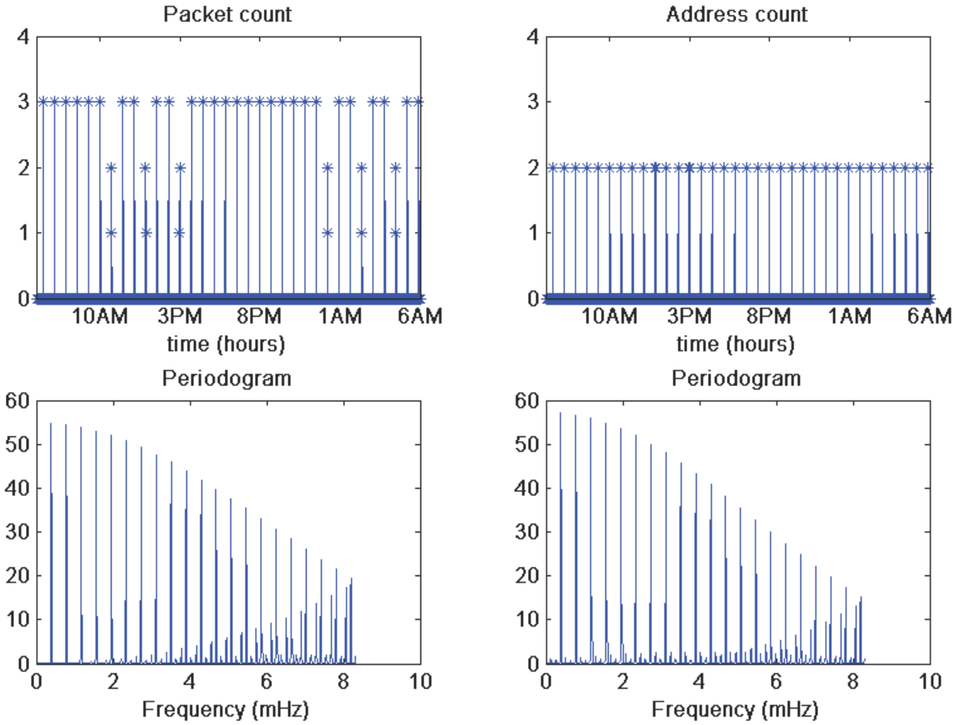

Figure 5. Left plots show the packet count for the traffic on port 6970 from a single host and its one sided periodogram. Right plots show the address count for the same traffic and its one sided periodogram. The aggregation interval for both count sequences is 60 s

Figure 5 shows the packet and address count sequences and their periodograms for port 6970 traffic of the suspected host on Monday Dec. 31, 2012. Periodograms of both sequences show a significant peak located at 0.4 mHz, which corresponds to a period of 42 minutes. The values of the ratio test are 54 and 60 for the periodograms of packet and address count sequences, respectively. Both these values are greater than threshold test. Therefore, we can conclude that this suspected host has another periodic behavior on port 6970. We note that the whole traffic of KSU's hosts on port 6970 on that day also exhibits periodic behavior (figures are not shown).

5. Conclusions

We detect botnets by detecting periodic behavior of bots' C2 communication traffic. This is done through testing the significance of the maximum ordinate of the periodogram of the packet, byte, address, and port count sequences. Since we note the similarity between the packet and byte count sequences in Figure 2, and between the address and port count sequences in Figure 3. We only consider the packet and address count sequences.

We analyze the IRC traffic and the traffic on port number 6970 of all hosts in KSU's network and find it to exhibit periodic behavior. We then examine the traffic of the suspected single bot generating the majority of this traffic and show that it exhibits a stronger period behavior as well in Figure 4-5. We note that since these port numbers are rarely used, legitimate traffic on these port number generate background traffic with low volume. This enables us to analyze the whole traffic of KSU and still be able to detect the periodic behavior. Therefore, we can conclude that the technique presented here can effectively be used to detect botnets in nowadays traffic as it was before when applied to datasets captured few years back.

Acknowldgemtns

We extend our appreciation to the Saudi National Science, Technology, and Innovation Plan (NSTIP) for funding this work through Research Project No. 10-INF1279-02.

References

[1] Gu, G., Yegneswaran, V., Porras, P., Stoll, J. Lee, W. (2009). Active Botnet Probing to Identify Obscure Command and Control Channels, *In*: *Computer Security Applications Conference*, 2009. *ACSAC '09*. *Annual*, Honolulu, HI, USA, Dec.7-11, 2009.

[2] Ha, D., Yan, G., Eidenbenz, S., Ngo, H. (2009). On the Effectiveness of Structural Detection and Defense Against P2P-based Botnets, *In*: *IEEE/IFIP International Conference on Dependable Systems & Networks (DSN)*, Lisbon, Portugal, June 29 – July 2, 2009.

[3] Zeidanloo, H. R., Zadeh, M. J., Safari, M., Zamani, M.(2010). A Taxonomy of Botnet Detection Techniques, *In*: *the 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT)*, Chengdu, China, July 9-11.

[4] Gu, G., Zhang, J., Lee, W (2008). BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic, *In*: *the* 15th *Network and Distributed System Security Symposium (NDSS'08)*, San Diego, CA, USA, Feb. 10–13.

[5] Kaur, S., Verma, A. (2013). Ontological Engineering Approach Towards Botnet Detection in Network Forensics, *International Journal of Computers and Technology (IJCT)*, 10, Sep.

[6] Zeidanloo, H. R., Manaf, A. B. (2010). Botnet Detection by Monitoring Similar Communication Patterns, *International Journal of Computer Science and Information Security* (IJCSIS), 7.

[7] Liu, J., Huang, J. (2010). Broadband Network Traffic Analysis and Study in Various Types of Applications, *In:the International Conference on Intelligent Control and Information Processing (ICICIP)*, Dalian, China, Aug. 13-15.

[8] Yu, X., Dong, X., Yu, G.. Qin, Y, Yue, D., Zhao, Y. (2010). Online Botnet Detection Based on Incremental Discrete Fourier Transform, *Journal of Networks*, 5, May.

[9] Arshad, S., Abbaspour, M., Kharrazi, M., Sanatkar, M. (2011). An Anomaly-Based Botnet Detection Approach for Identifying Stealthy Botnets, *In:* Computer Applications and Industrial Electronics (ICCAIE), 4-7 Dec.

[10] CISCO. (2011). *NetFlow Version 9 Flow-Record Format*. Available: http://www.cisco.com/en/US/technologies/tk648/tk362/technologies_white_paper09186a00800a3db9.html

[11] Ma, X., Guan, X., Tao, J., Zheng, Q., Guo, Y., Liu, L., Zhao, S. (2010). A Novel IRC Botnet Detection Method Based on Packet Size Sequence, *In:* The IEEE International Conference Communications (ICC), Cape Town, South Africa, 23-27 May.

[12] AsSadhan, B., Moura, M. F. (2013). An Efficient Method to Detect Periodic Behavior in Botnet Traffic by Analyzing Control Plane Traffic, *Journal of Advanced Research, Available online at: http://dx.doi.org/10.1016/j.jare.2013.11.005*.

[13] Jackson, A. W., Lapsley, D., Jones, C., Zatko, M., Golubitsky, C., Strayer, W. T. (2009). SLINGbot: A System for Live Investigation of Next Generation Botnets, *In*: *Cybersecurity Application and Technologies Conference for Homeland Security* (CATCH), Washington, DC, USA, Mar. 3-4.

[14] Lippmann, R., Haines, J., Fried, D., Korba, J. , Das, K. (2000). The 1999 DARPA Off-Line Intrusion Detection Evaluation, In *the* 3rd *International Workshop on Recent Advances in Intrusion Detection (RAID)*, New York, NY, USA, Oct. 2-4.

[15] LBNL/ICSI. (2013). *LBNL/ICSI Enterprise Tracing Project*. Available: http://www.icir.org/enterprise-tracing

[16] AsSadhan, B. (2009). Network Traffic Analysis Through Statistical Signal Processing Methods, Carnegie Mellon University, Pittsburgh, PA, USA, .

[17] Tavallaee, M., Stakhanova, N., Ghorbani, A. A. (2012). Toward Credible Evaluation of Anomaly-Based Intrusion-Detection Methods, *IEEE Transactions on the Systems, Man, and Cybernetics, part c: applications and reviews*, 40, Sep.

[18] Endace, *DAG* 7.5G2 *Card User Guide*, Version May 10, 2012.

[19] Endace, *Network Tapping Technical Overview*, Version Aug. 14, 2009.

[20] Pei, Z. , Xiao-hong, H., Min-qi, L., Chun-yu, N., Yan, M (2010). Fast restorable prefix-preserving IP address anonymization for IPv4/IPv6, *The Journal of China Universities of Posts and Telecommunications*, 17, Dec. 2010.

[21] Kumar, S. A. (2012). Conficker Botnet Prevention: Crypto-Pan Algorithm, *International Journal of Engineering and Science*, 1, Dec.

[22] Lapworth,L. (2013). *The Perl Programming Language*. Available: http://www.perl.org

[23] Oppenheim, A., Schafer, R., Buck, J. (2010). Discrete-Time Signal Processing, 3 ed. Upper Saddle River, New Jersey: Prentice-Hall.

[24] Trees, H. V. (2013). Detection, Estimation, and Modulation Theory. Part 1, second ed. Hoboken, New Jersey: John Wiley & Sons.

[25] Priestley, M. B. (1982). *Spectral Analysis and Time Series*. Academic Press