Blacklist-based Malicious IP Traffic Detection

Ibrahim Ghafir and Vaclav Prenosil
Faculty of Informatics, Masaryk University
Brno, Czech Republic
ibrahim_ghafir@hotmail.com, prenosil@fi.muni.cz

***Abstract*—At present malicious software or malware has increased considerably to form a serious threat to Internet infrastructure. It becomes the major source of most malicious activities on the Internet such as direct attacks, (distributed) denial-of-service (DOS) activities and scanning. Infected machines may join a botnet and can be used as remote attack tools to perform malicious activities controlled by the botmaster. In this paper we present our methodology for detecting any connection to or from malicious IP address which is expected to be command and control (C&C) server. Our detection method is based on a blacklist of malicious IPs. This blacklist is formed based on different intelligence feeds at once. We process the network traffic and match the source and destination IP addresses of each connection with IP blacklist. The intelligence feeds are automatically updated each day and the detection is in the real time.**

***Keywords*—Cyber attacks, botnet, malicious IP, malware, intrusion detection system.**

I. Introduction

At present malicious software or malware has increased considerably to form a serious threat to Internet infrastructure. It becomes the major source of most malicious activities on the Internet such as direct attacks [1], (distributed) denial-of-service (DOS) activities [2] and scanning [3]. Nowadays hackers and cyber criminals are not motivated only by curiosity and fame seeking, they aim for illegal financial profit that encourages them to build more sophisticated malwares. Supported by availability of easy-to-use toolkits to create malware, this issue is likely to remain a challenge to users, governments and businesses.

By exploiting vulnerabilities of the servers, malware can be hosted on web servers. The mere visit of a website hosted on the exploited server causes to the user's computer to be infected, particularly when the user's operating system and/or internet browser are unpatched [4], this activity is also referred as a "drive-by attack" and the malware is known as a "bot malware". Infected machines may join a botnet and can be used as remote attack tools to perform malicious activities controlled by the botmaster. Botnets are networks established by a group of infected machines, called bots, that are compromised and controlled by the attackers that called botmasters [5], [6].

A bot can exploit vulnerable machines through Trojan insertion or direct exploitation, since it is a self-propagating application like other worms and viruses. What is different from previous generations of worms and viruses, bots are able to create command and control (C&C) channels between the infected machines and C&C servers [7]. The C&C servers are controlled by the botmasters and used to instruct and update the bots.

Botnet C&C traffic detection is not easy and forms a challenge for the current intrusion detection systems, because the volume of this traffic is low. Moreover, it is similar to normal traffic and follows the same protocol usage [8].

In this paper we present our methodology for detecting any connection to or from malicious IP address which is expected to be C&C server. Our detection method is based on a blacklist of malicious IPs. This blacklist is formed based on different intelligence feeds at once. Why rely on one vendor feed when we can use many? No single security team has all the answers. We process the network traffic and match the source and destination IP addresses of each connection with IP blacklist. The intelligence feeds are automatically updated each day and the detection is in the real time.

The remainder of this paper is organized as follows. Section 2 presents previous related work to malicious IP traffic detection. Our methodology and implementation of our detection method are explained in Section 3. Section 4 shows our results and section 5 concludes the paper.

II. Related Work

For years preventing users from connecting to malicious servers or websites has depended on blacklists and reputation systems. Blacklist-based techniques are the most common solutions to protect both client-side and network-side. Commercial security appliances and DNS block lists can be used for Network-side filtering (DNSBL) [9], [10], while the current web browsers contain client-side filtering [11].

Reputation systems are used to deny hosts from connecting to malicious websites. Reputation depends on different types of date present in each domain or address. Attributes extracted from link structures, WHOIS information and domains are used for reputation criteria. Antonakakis et al. [12] developed Notos, a dynamic reputation system. The system depends on many sources like AS information, border gateway protocol prefixes and DNS zones to gather information. Then, the collected information is used to model network and zone behaviors of malicious and benign domains. After that, a reputation score is calculated for each domain name based on those models.

Felegyhazi et al. [13] proposed domain-based proactive blacklisting. It uses name server and registration information to predict malicious domains based on a small set of other known malicious domains. Ma et al. [14] presented a supervised learning approach for categorizing URLs as malicious or benign. This approach is based on host-based features and lexical structure of URLs like geographical information and WHOIS records.

978-1-4799-8553-1/15/$31.00 © 2015 IEEE

Chiba et al. [15] proposed a mechanism for detecting connections related to web-based malware. It is able to prevent hosts from connecting even to unknown malicious websites based on attributes extracted from structures of IP addresses. Based on a machine learning technique, they developed a scalable and lightweight detection scheme. Their method benefits from the empirical observation that IP addresses are more settled than other metrics such as DNS and URLs. While the strings that form IP addresses are not often changing, domain names and URLs are more changeable, i.e., IPv4 address space is mapped onto 4-bytes strings.

By monitoring only the time-to-live (TTL) value in the internet protocol (IP) header, a new approach for detecting malicious packets was proposed by Yamada and Goto [16]. This approach depends on the fact that the usual number of routers, through which an IP packet passes to a destination host, is less than 30. Based on their monitoring, they found that some IP packets have an unusual TTL value that is decreased by more than 30 increments from the initial TTL value. Special software, most often, has generated those packets. IP packets with abnormal TTL values are considered to be malicious. Some other methods based on machine learning, with deep packet inspections or numerous signature files, have been proposed for detecting malicious packets [17], [18].

With regards to botnets, by using a multilayer Feed-Forward Neural network, an HTTP based botnet detection system was presented by Kirubavathi et al. [19]. They utilized some features related to TCP connections for botnet detection. Those features were extracted in specific time intervals based on the fact that web-based botnets periodically make a web request from the C&C web server to be updated and instructed. They usually do not keep a continuous connection with the C&C server. Stateful-SBB, a co-evolutionary system for detecting automatically generated malicious (botnet) domain names was proposed by Haddadi et al. [20]. They showed that botnet C&C domain names, which are located in the network packet payload, could be distinguished by their system.

Francois et al. [21] presented a NetFlow monitoring framework to track communication patterns by using a simple host dependency model. This framework is able to detect similar botnet behavioral patterns based on clustering techniques and linkage analysis. A botnet detection system based on flow intervals was proposed by Zhao et al. [22]. For botnet detection, some flow attributes were utilized with decision tree classifiers and Bayesian networks. Those attributes were extracted from packet headers.

III. Methodology

In this section we propose our methodology for detecting any connection to or from malicious IP. Our detection method is based on a blacklist of malicious IPs. As it is shown in Figure 1, we process the network traffic and match the source and destination IP addresses for each connection with IP blacklist. The blacklist is automatically updated each day based on different intelligence feeds at once and the detection is in the real time. We have implemented our detection method on top of Bro [23], [24], which is a passive, open-source network traffic analyzer.

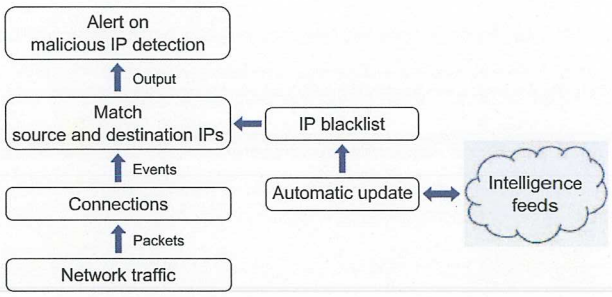

Fig. 1. Methodology of malicious IP traffic detection.

Figure 2 explains the implementation of our detection method in Bro, first we start processing the network traffic. Bro is able to reduce the incoming packet stream into a series of higher-level *events*, so we can get *new_connection* event. This event is generated for every new connection and it is raised with the first packet of a previously unknown connection. Through this event we should check both connection sides (source and destination IPs) to detect if the connection is to or from malicious IP.

In the first case, detecting if the connection is to malicious IP, first we check if this connection is established by a host from our network, therefore we check the source IP address through *is_local_addr* function, this function returns true if an address corresponds to one of the defined local networks, false if not. Here we should define the subnet of our network. After that, we check if the destination IP address is in *t_ip_blacklist* table, this table contains malicious IPs which we get them from different intelligence feeds [25], [26], [27], [28], [29], [30], [31], [32], [33]. This blacklist is automatically updated each day as it is shown in Figure 3. When a match is found, that means that the connection is established to a malicious IP. Before we raise an alert we check if we got an alert about the same host during the last day, because we don't want to send many alerts about the same infected host during one day, therefore we check if the current IP (infected host) is existent in *t_suppress_ip_alert* table, this table contains all detected hosts during the last day. If the source IP address is not in *t_suppress_ip_alert* table, we can write the following information into *blacklist_detection_ip.log:*

timestamp = c$start_time
alert_type = "ip_alert"
connection = c$id
infected_host = cidorig_h
malicious_ip = cidresp_h

We send an alert email about malicious IP traffic detection to RT (Request Tracker) where the network security team can perform additional forensics and response to it [34].We generate an event, *ip_alert*, about malicious IP detection, this event can be used for alert correlation [35]. We should add the IP address of the current (infected) host into *t_suppress_ip_alert* table where it will stay for one day to be sure that we will not get another alert about the same infected host during one day.

In the second case, detecting if the connection is from malicious IP, we follow the same procedure of the first case

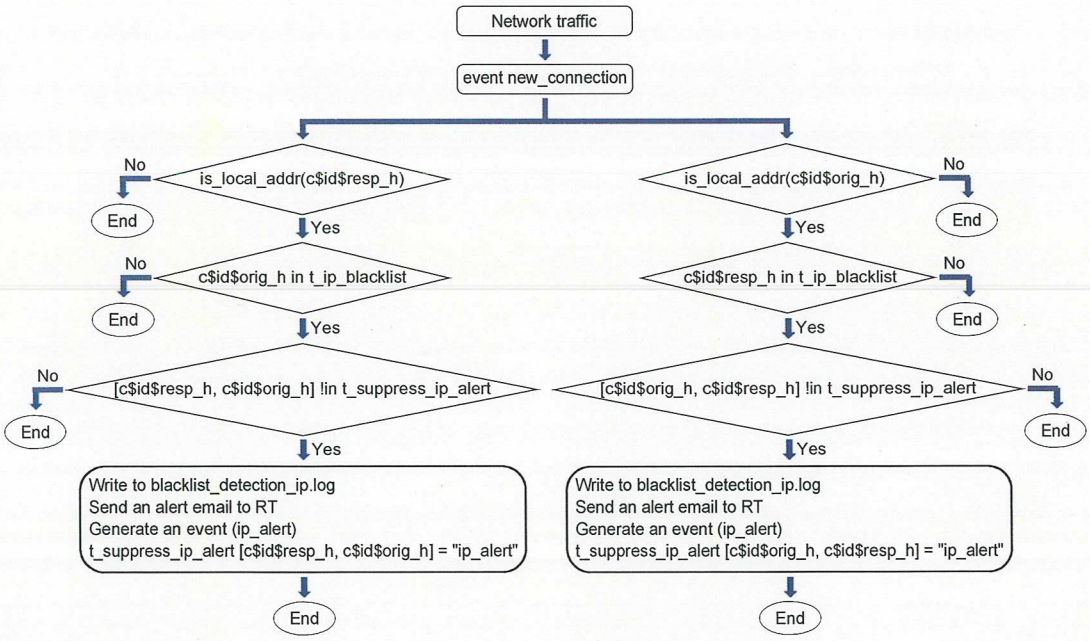

Fig. 2. Implementation of malicious IP traffic detection method.

but we pay attention to the source and destination IP addresses as it is explained in Figure 2.

For automatic update of *t_ip_blacklist* table which is used in our methodology, Figure 3 shows how it is done.

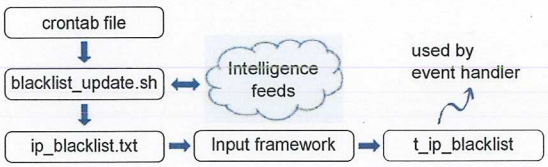

Fig. 3. Automatic update for IP blacklist.

We start from user *crontab file* which is configured to run *blacklist_update.sh* each day at 3:00 am, this shell script will connect through Internet to the intelligence feeds and download updated blacklist of malicious IPs into *ip_blacklist.txt* file. Then we have Input Framework which enables us to use the text file as an input for our methodology which is implemented on top of Bro, this framework will read *ip_blacklist.txt* file into *t_ip_blacklist* table and this table will be used by *event handler* as it is explained above. This automatic update is done without stopping network traffic live monitor.

IV. Evaluation and Results

We used three scenarios to evaluate our methodology. In the first one, we ran a script initiating connections to malicious IPs on a computer connected to the monitored network. In the second scenario, we applied our method on pcap files which contained malicious IP traffic. In the third scenario, we monitored the campus network for hosts connecting to malicious IPs.

In the first scenario, we installed a script on a user's computer of our network. The network was monitored by our detection method and the script was written to connect to random malicious IPs from the blacklist. In this scenario, we focused on the real-time detection feature. Once a malicious connection is detected, a report is sent to RT by our method.

The test consisted of the following steps. First, a connection to a randomly picked IP from the blacklist was established and the connection time was recorded with millisecond precision. Then, our detection method sent an RT ticket as soon as the malicious connection was detected. After the RT ticket was received, we recorded the time of arrival with millisecond precision. Based on both recorded time we calculated the detection delay. The average detection delay was 270 ms with standard deviation of 55 ms. Figure 4 shows the results.

In the second scenario, we applied our methodology on three pcap files contained malicious IP traffic. These pcap files were analyzed by the provider, so we used this fact to set the ground truth. The first pcap file contained traffic infected by the malware (*PizzaHut_Coupon.exe*) with MD5 hash *191a02952905cc0037753700636c3339* [36]. The infection was delivered by an email attachment sent by Asprox botnet which uses phishing emails and sends fake *Pizza Hut* emails with the subject line: *Free Pizza*. The second pcap file traffic was infected by the malware (*Label-CA-Toronto.exe*) with MD5 file hash *a6ba2cadc7c6891a5f437b212a18ac52* [37]. The infection was also delivered by Asprox botnet phishing email, but different malicious IP was involved. The third pcap file traffic was infected by a malware with MD5 file hash *dc5c71aef24a5899f63c3f9c15993697* [38]. The infection

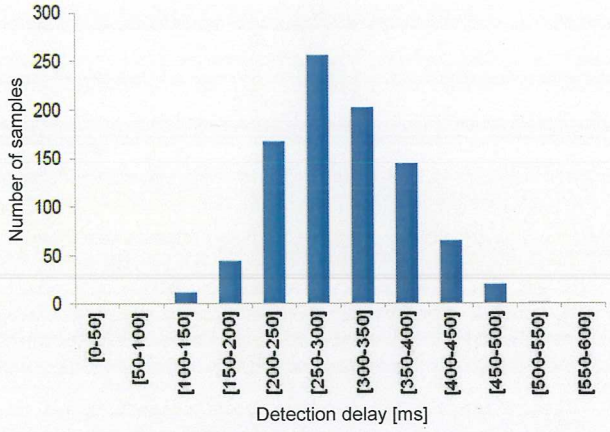

Fig. 4. Detection delay of our method.

was delivered by drive-by-download attack and five malicious IPs were involved.

We set up our method that it consumed the pcap files and produced log files. We applied our detection method on the pcap files and it was successfully able to detect all malicious IPs which were involved in those infections. Note that we did not provide the ground truth blacklist to our detection method. Figures 5, 6, and 7 show part of *blacklist_detection_ip.log* produced by our detection method for each pcap file. The log file contains more information about the malicious connection than in the figures (like its source and destination ports) but the figures show only the interesting part of the log.

```
#fields  timestamp      alert_type  infected_host    malicious_ip
#types   time           string      addr             addr
1414537682.066774  ip_alert    192.168.56.255   192.168.56.101
1414537713.067728  ip_alert    85.12.29.172     192.168.56.101
#close 2014-11-29-20-40-04
```

Fig. 5. Part of the log produced by our method for the fist pcap file.

```
#fields  timestamp      alert_type  infected_host    malicious_ip
#types   time           string      addr             addr
1415437682.066018  ip_alert    104.152.888.175  128.13.136.177
1415437674.992953  ip_alert    128.13.136.8 .   128.13.136.177
#close 8514-11-89-85-42-41
```

Fig. 6. Part of the log produced by our method for the second pcap file.

```
#fields  timestamp      alert_type  infected_host    malicious_ip
#types   time           string      addr             addr
141155537681 27325  ip_alert    15. 81678 348147 1478 2181248 0
14115553518 3. 627  ip_alert    15. 81678 348147 1478 2181248 5
141155535189 10920  ip_alert    15. 81678 348147 758438918126
141155513. 846. . 70  ip_alert  15. 81678 348147 958 1008 158110
14115551978 72291   ip_alert    15. 81678 348147 998798978 1. 9
#close . 314-11-. 5-. 3-26-. 4
```

Fig. 7. Part of the log produced by our method for the third pcap file.

In the third scenario, we monitored the campus live traffic for malicious IPs detection. The method was set up to create a log file of detected malicious IPs. We set up a server hosting our detection method and passively analyzing the campus live traffic. The monitoring was performed for one month. We correlated the list of hosts involved in malicious IP traffic with results of a malicious domain detection method. As it is shown in Figure 8, 22 hosts were detected involved in malicious domain connections and 37 hosts were detected involved in malicious IP connections. Of these, 14 hosts were detected involved in both malicious IP and domain connections, which indicated that they were infected by malware.

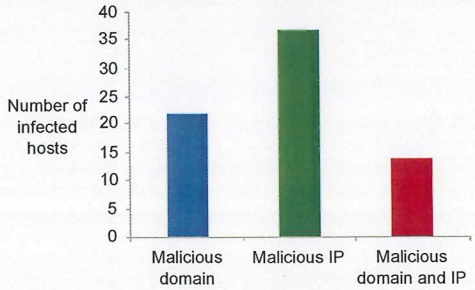

Fig. 8. Detected hosts by malicious domain and IP detection methods.

Our detection method also sent an alert email about each malicious IP detection to RT where the network security team can perform additional forensics and response to it. Figure 9 shows an example of *Bro_Malicious_IP* ticket which was sent by email to RT.

```
Greetings,

the security team CSIRT-MU detected involvement of the IP address
[redacted] into the following incident:

Incident type: Bro_Malicious_IP
Time of detection: 2014-11-27 18:25:37 +0100
IP address: [redacted]
Domain name: ---

Details of this incident can be found at this address:
https://reports.csirt.muni.cz/A4FE72DC-65A8-1C74-8627-5664BE78D342

Best regards,
CSIRT-MU, the security team of Masaryk University
http://www.muni.cz/csirt
        Date: Thu, 27 Nov 2014 18:25:55 +0100
```

Fig. 9. Bro_Malicious_IP ticket.

V. Conclusion and Future Work

In this paper we presented our methodology for detecting any connection to or from malicious IP address which is expected to be C&C server. Our detection method is based on a blacklist of malicious IPs. This blacklist is formed based on different intelligence feeds at once.

For future work, the output of this detection method will be correlated with the outputs of other detection methods to raise an alert on APT attack detection.

ACKNOWLEDGMENTS

This work has been supported by the project CYBER-2 funded by the Ministry of Defence of the Czech Republic under contract No. 1201 4 7110.

REFERENCES

[1] K. G. Anagnostakis, S. Sidiroglou, P. Akritidis, K. Xinidis, E. P. Markatos, and A. D. Keromytis, "Detecting targeted attacks using shadow honeypots." in *Usenix Security*, 2005.

[2] D. Moore, C. Shannon, D. J. Brown, G. M. Voelker, and S. Savage, "Inferring internet denial-of-service activity," *ACM Transactions on Computer Systems (TOCS)*, vol. 24, no. 2, pp. 115–139, 2006.

[3] S. Staniford, J. A. Hoagland, and J. M. McAlerney, "Practical automated detection of stealthy portscans," *Journal of Computer Security*, vol. 10, no. 1, pp. 105–136, 2002.

[4] K.-K. R. Choo, "The cyber threat landscape: Challenges and future research directions," *Computers & Security*, vol. 30, no. 8, pp. 719–731, 2011.

[5] P. Bacher, T. Holz, M. Kotter, and G. WICH-ERSKI, "Know your enemy: Tracking botnets, 2005," *URL http://www. honeynet. org/papers/bots*, vol. 4, pp. 24–33.

[6] H. Choi, H. Lee, and H. Kim, "Botgad: detecting botnets by capturing group activities in network traffic," in *Proceedings of the Fourth International ICST Conference on COMmunication System softWAre and middlewaRE*. ACM, 2009, p. 2.

[7] G. Gu, P. A. Porras, V. Yegneswaran, M. W. Fong, and W. Lee, "Bothunter: Detecting malware infection through ids-driven dialog correlation." in *USENIX Security*, vol. 7, 2007, pp. 1–16.

[8] G. Gu, J. Zhang, and W. Lee, "Botsniffer: Detecting botnet command and control channels in network traffic," 2008.

[9] Uribl, "Rss feeds," http://rss.uribl.com/, accessed: 29-11-2014.

[10] OpenDNS, "Protect your business with opendns umbrella," https://www.opendns.com/, accessed: 29-11-2014.

[11] Windows, "Smartscreen filter," http://windows.microsoft.com/en-US/internet-explorer/products/ie-9/features/smartscreen-filter/, accessed: 29-11-2014.

[12] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster, "Building a dynamic reputation system for dns." in *USENIX security symposium*, 2010, pp. 273–290.

[13] M. Felegyhazi, C. Kreibich, and V. Paxson, "On the potential of proactive domain blacklisting," in *Proceedings of the Third USENIX Workshop on Large-scale Exploits and Emergent Threats (LEET), San Jose, CA, USA*, 2010.

[14] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Beyond blacklists: learning to detect malicious web sites from suspicious urls," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009, pp. 1245–1254.

[15] D. Chiba, K. Tobe, T. Mori, and S. Goto, "Detecting malicious websites by learning ip address features," in *Applications and the Internet (SAINT), 2012 IEEE/IPSJ 12th International Symposium on*. IEEE, 2012, pp. 29–39.

[16] R. Yamada and S. Goto, "Using abnormal ttl values to detect malicious ip packets," *Proceedings of the Asia-Pacific Advanced Network*, vol. 34, pp. 27–34, 2013.

[17] S. Seufert and D. O'Brien, "Machine learning for automatic defence against distributed denial of service attacks," in *Communications, 2007. ICC'07. IEEE International Conference on*. IEEE, 2007, pp. 1217–1222.

[18] K. Hwang, Y. Chen, and H. Liu, "Defending distributed systems against malicious intrusions and network anomalies," in *Parallel and Distributed Processing Symposium, 2005. Proceedings. 19th IEEE International*. IEEE, 2005, pp. 8–pp.

[19] G. K. Venkatesh and R. A. Nadarajan, "Http botnet detection using adaptive learning rate multilayer feed-forward neural network," in *Information Security Theory and Practice. Security, Privacy and Trust in Computing Systems and Ambient Intelligent Ecosystems*. Springer, 2012, pp. 38–48.

[20] F. Haddadi and A. N. Zincir-Heywood, "Analyzing string format-based classifiers for botnet detection: Gp and svm," in *Evolutionary Computation (CEC), 2013 IEEE Congress on*. IEEE, 2013, pp. 2626–2633.

[21] J. François, S. Wang, T. Engel *et al.*, "Bottrack: tracking botnets using netflow and pagerank," in *NETWORKING 2011*. Springer, 2011, pp. 1–14.

[22] D. Zhao, I. Traore, A. Ghorbani, B. Sayed, S. Saad, and W. Lu, "Peer to peer botnet detection based on flow intervals," in *Information Security and Privacy Research*. Springer, 2012, pp. 87–102.

[23] V. Paxson, "Bro: a system for detecting network intruders in real-time," *Computer networks*, vol. 31, no. 23, pp. 2435–2463, 1999.

[24] The-Bro-Project, "The bro network security monitor," https://www.bro.org/, accessed: 29-11-2014.

[25] Computer-Incident-Response-Center-Luxembourg, "Source and destination ip blocklist," http://misp.circl.lu/, accessed: 29-11-2014.

[26] Abuse.ch, "Spyeye ip blocklist," https://spyeyetracker.abuse.ch/blocklist.php?download=ipblocklist, accessed: 29-11-2014.

[27] "Palevo c and c ip blocklist," https://palevotracker.abuse.ch/blocklists.php?download=ipblocklist, accessed: 29-11-2014.

[28] "Zeus ip blocklist," https://zeustracker.abuse.ch/blocklist.php?download=ipblocklist, accessed: 29-11-2014.

[29] Malware-Domain-List, "Malware domain list, hostlist ip," http://www.malwaredomainlist.com/hostslist/ip.txt, accessed: 29-11-2014.

[30] Sentinel-IPS, "ci-badguys.txt," http://www.ciarmy.com/list/ci-badguys.txt, accessed: 29-11-2014.

[31] Internet-Storm-Center, "Ciarmy ip blacklist," http://feeds.dshield.org/top10-2.txt, accessed: 29-11-2014.

[32] OpenBL, "Abuse reporting and blacklisting," http://www.openbl.org/lists/base.txt, accessed: 29-11-2014.

[33] Emerging-Threats, "Compromised ips," http://rules.emergingthreats.net/blockrules/compromised-ips.txt, accessed: 29-11-2014.

[34] Best-Practical-Solutions, "Rt: Request tracker," https://www.bestpractical.com/rt/, accessed: 29-11-2014.

[35] Y. B. Leau, S. F. Tan, S. Manickam *et al.*, "A comparative study of alert correlations for intrusion detection," in *Proceedings-2013 International Conference on Advanced Computer Science Applications and Technologies, ACSAT 2013*. IEEE, 2014, pp. 85–88.

[36] Malware-Traffic-Analysis, "2014-10-28 - asprox botnet serving free pizza," http://malware-traffic-analysis.net/2014/10/28/index.html, accessed: 29-11-2014.

[37] "2014-09-11 - aspros botnet phishing campaign," http://malware-traffic-analysis.net/2014/09/11/index2.html, accessed: 29-11-2014.

[38] "2014-10-28 - asprox botnet serving free pizza," http://malware-traffic-analysis.net/2014/09/29/index.html, accessed: 29-11-2014.