A Novel Search Engine to Uncover Potential Victims for APT Investigations

Shun-Te Liu[1,2], Yi-Ming Chen[1], and Shiou-Jing Lin[2]

[1] Department of Information Management, National Central University, Taoyuan, Taiwan, R.O.C
[2] Information & Communication Security Lab, TL, Chunghwa Telecom Co., Ltd. Taoyuan, Taiwan, R.O.C
{rogerliu,sjlin}@cht.com.tw, cym@cc.ncu.edu.tw

Abstract. Advanced Persistent Threats (APT) are sophisticated and target-oriented cyber attacks which often leverage customized malware and bot control techniques to control the victims for remotely accessing valuable information. As the APT malware samples are specific and few, the signature-based or learning-based approaches are weak to detect them. In this paper, we take a more flexible strategy: developing a search engine for APT investigators to quickly uncover the potential victims based on the attributes of a known APT victim. We test our approach in a real APT case happened in a large enterprise network consisting of several thousands of computers which run a commercial antivirus system. In our best effort to prove, the search engine can uncover the other unknown 33 victims which are infected by the APT malware. Finally, the search engine is implemented on Hadoop platform. In the case of 440GB data, it can return the queries in 2 seconds.

1 Introduction

The cyber attacks become more and more sophisticated. Recently, this kind of target-oriented, covered and long-term attacks is labeled as advanced persistent threat (APT) [1-5]. Much research considers that APTs are the sophisticated and target-oriented cyber attacks which often leverage customized malware and bot control techniques to remotely control the victims.[1-5]. The victims will become the stepping stones for the attackers to access valuable information inside the enterprise network [6]. Therefore, the sooner we find the APT malware-infected computers, the smaller the loss caused by the APTs.

HTTP requests log is a valuable data for determining APT malware [7]. As Web-related protocols are allowed almost everywhere, the APT malware is mostly equipped with remote-controlled ability under the HTTP-based command and control (C&C) infrastructure to facilitate the attacks on the intranet [8-10]. Although much research can detect bot-infected computers [11, 12] or detect bot behavior [13-16], they require more bot samples to train a feasible model. This is a big problem to these approaches because the APT malware samples are few and often customized.

C.-H. Hsu et al. (Eds.): NPC 2013, LNCS 8147, pp. 405–416, 2013.

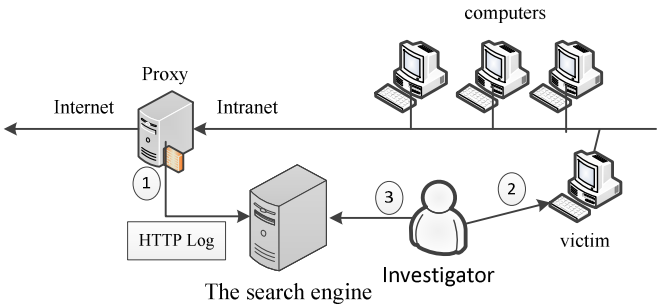

Fig. 1. The working concept of the search engine

In this paper, rather than detecting the APT malware, we take a more flexible strategy: developing a search engine for searching the potential victims to respond to APTs quickly. The working concept of our idea is shown in Fig 1. Each computer's HTTP requests are logged by proxy and sent to the search engine periodically. Once an APT malware is found, by giving the attributes of the malicious HTTP requests invoked by the APT malware, the search engine can search and rank the potential C&C servers and malware-infected computers from the historical HTTP requests. This approach is very useful for the APT investigation. We test our approach in a real APT case happened in a large enterprise network consisting of several thousands of computers, which run a commercial anti-virus system. The three known C&C servers are ranked in top 10 of the web sites. Meanwhile, in our best effort to prove, the search engine can find the other 18 C&C servers and the other 33 APT malware-infected computers. In addition, to process the huge volume of proxy logs quickly, the search engine is implemented on Hadoop platform. In the case of 440GB HTTP logs, it can return the queries in 2 seconds.

This paper contributes to network security in the following areas:

1. Propose a rank mechanism to rank the potential APT victims and C&C servers.
2. Develop a search engine on Hadoop platform to process the huge volume of HTTP requests.
3. Prove the usefulness of our approach in a real APT investigation case.

The organization of this paper is described as follows: Section 2 describes the previous research in APTs and botnet detection. In Section 3, we propose a ranking mechanism to rank the websites. Section 4 describes the prototype of the search engine. Section 5 shows experimental results. Section 6 will conclude and describe the future work.

2 Previous Research

2.1 APT Characteristics

As the name implies, advanced persistent threat (APT) uses highly targeted method persistently for compromising the data security, but the definition of APT in academic

research is still unclear today. To understand that, we extract the APT characteristics, as shown in Table 1, described in the reports and the studies according to intrusion phases [17]. In reconnaissance phase, the targets are highly profiled including organization, people and computer environments before being attacked. The attackers do their best efforts to find out the weakness of their targets. In gaining access phase, the APT attackers use not only uncovered software vulnerabilities (zero day exploits) but also human weakness (social engineering attacks) to compromise the targets. This means that APT almost can bypass the conventional signature-based detection approaches. In maintaining access phase, the attackers often leverage bot control techniques to control the victims in the target's enterprise network. Finally, the objectives of the APT attackers are valuable information of their targets.

The research [18] analyzed a large corpus of targeted attacks identified by Symantec during the year 2011. The results show that only 5% malware used in APTs were identified by antivirus software. This means that preventing the APTs in gaining access phase is very difficult, especially when the zero day exploits are used in the attacks. Once the attack success, the signature-based intrusion detection approaches are often fail because they can't recognized the malware used. For these reasons, the reaction of APTs becomes more and more important for the enterprises to reduce the damages. Therefore, we focus on the characteristics of APT in maintaining phase and give APT a definition as follows:

Table 1. The APT characteristics extracted from the studies, where phase 1 is reconnaissance, phase 2 is gaining access, phase 3 is maintaining access and phase 4 is achieve objectives

Phase	Characteristics	[19]	[20]	[18]	[4]	[21]	[5]	[22]	[9]	[23]	[8]	[24]
1	Target-oriented	V	V	V		V	V	V			V	
	Highly profile	V	V	V	V			V		V		V
2	Zero day exploit	V	V	V	V	V		V	V	V	V	
	Coordinative	V	V	V	V	V	V	V	V			V
	Social engineering	V	V	V	V	V		V	V		V	
	Combine several attack skill	V	V	V	V	V	V	V	V			V
3	Slow and stealthy				V					V		
	Remote access	V	V	V	V			V	V	V	V	V
	Customized malware	V	V	V	V	V	V	V	V	V	V	
	Command and control server	V	V		V		V	V	V	V	V	
	Encrypt control traffic	V			V				V		V	
4	Valuable information	V	V	V	V	V	V	V		V	V	V
	SCADA control	V		V	V							
	Underground economy		V				V					
	Political								V			

APT is a sophisticated and target-oriented cyber attack which often leverage customized malware and bot control techniques to remotely access valuable information.

This definition indicates a key point that the APT attackers must maintain the access channels to control the victims remotely. In the cases of APT incidents, the investigations point out that these access channels are constructed by bot control techniques. The victims become the stepping stones for the attackers to access the valuable information inside the enterprise network [6]. Therefore, bot detection approaches may be applied to APT detection and investigation.

2.2 Bot Attributes and Detection

Based on the control methods, bot can be divided into several types such as IRC and HTTP bot. As web-related protocols are allowed almost everywhere, the APT malware is often equipped with remote- controlled ability under the HTTP-based C&C infrastructure to facilitate the attacks on the intranet[8, 9, 19]. Therefore, the HTTP bot detection approaches may be useful for detecting the APT malware.

In [11], the authors leverage the IRC nickname to detect bot contaminated hosts. They can also detect the HTTP bot by the common strings in URL of the bot servers. Based on the observation of the pre-programmed activities related to C&C, Botsniffer [16] capture the spatial-temporal correlation in network traffic and utilize statistical algorithms to detect botnets. Botzilla [13] capture malware traffic to detect the "phoning home" behavior. The phoning home traffic will be tokenized to generate the signature for detecting the malware-infected computer. In [15], the authors present a malware clustering system. They analyze the structural similarities among malicious HTTP traffic and automatically generate HTTP-based malware signatures for further detection. In [14], the authors focus on detecting C&C channels masquerading as web traffic. They use 2v-gram based anomaly detection approach to distinguish the C&C traffic from web traffic. The summary of these studies is shown in Table 2.

Table 2 shows that the above approaches focus on detection rather than investigation. Meanwhile, except Botzilla, these approaches require a lot of malware samples to train a feasible model for detection. This is a big problem for these approaches because the APT malware samples are often few and specific. Furthermore, Botzilla leverages network level traffic for bot detection. It's almost impossible to retrospect the data for APT investigation because the size of historical network level traffic is much larger than that of application level traffic. Therefore, it is required a new approach to deal with this problem.

Table 2. The summary of HTTP bot detection approaches

Item	**[11]**	**[16]**	**[13]**	**[15]**	**[14]**
Objective	Bot detection	Bot detection	Bot detection	Bot detection	C&C detection
Traffic level	Application	Application	Network	Application	Application
Match function	RE	NG	NG	NG or RE	NG
Require many-malware samples	Yes	Yes	No	Yes	Yes

RE: Regular Expression NG: N-Gram.

3 The Ranking Mechanism

3.1 Overview

In the enterprise network, HTTP proxy acts as an intermediary for HTTP requests from clients seeking resources from the websites. The structure of the logged HTTP requests is illustrated in Fig 2. We leverage HTTP logs to rank the websites by the probability of being C&C servers. The probability is determined based on two observations: 1) C&C servers often contain much few information than legitimate websites. Therefore, the higher diversity of a web site is, the higher probability it is a legitimate server, and vice versa, 2) to pretend the user behavior, the malware often actively invokes HTTP requests to the C&C servers to acquire the commands for the further actions. Therefore, to rank the websites, two scores are introduced: reversed diversity score (d) and continuity score (p) of a website.

3.2 Reversed Diversity Score

Diversity score is a quantitative measure that increases when the number of types into which a set of entities has been classified increases. To estimate the diversity score of a website, the entities can be file types of the web pages. However, the logs only provide the web pages requested by the computers, not all the web pages of a website. In this case, as it looks likely the sample survey in ecology and information science, the popular diversity index Shannon-Wiener (H') [25] are used to determine the diversity score of a website.

Let HTTP requests R consist of a set H of hostname, a set G of web pages, a set F of file types and a set S of source IP. Let the number of web pages and number of web pages with file type j of a website i be G_i and $g_{i,j}$. The diversity of the website i is calculated by Shannon-Wiener (H') as follows:

$$H'_i = -\sum_{j=1}^{F} f_j \log f_j \text{, where } f_j = g_{i,j}/G_i. \quad (1)$$

H' value is ranged from 0 to 4.5. As the higher diversity often means higher probability of being a legitimate website, the probability (d) of a website being a C&C server is calculated by reversing the diversity:

$$d_i = \frac{4.5 - H'_i}{4.5} \quad (2)$$

For the example of Table 3, the H' value and the reversed diversity score d of the three websites are:

$\boldsymbol{H'}_1$=-(0.99/(log0.99) + 0.01(log0.01)) = 0.056 $\quad$ $\boldsymbol{d}_1$ = (4.5-0.056)/4.5 = 0.987

$\boldsymbol{H'}_2$=-(0.5/(log0.5) + 0.5(log0.5)) = 0.693 $\quad$ $\boldsymbol{d}_2$ = (4.5-0.693)/4.5 = 0.846

$\boldsymbol{H'}_3$=-(0.33/(log0.33)+0.33/(log0.33)+0.33/(log0.33))= 1.1 $\quad$ $\boldsymbol{d}_3$ = (4.5-1.1)/4.5 = 0.755

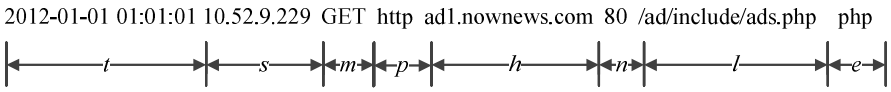

Fig. 2. HTTP log structure, where t is the timestamp, s is source IP, m is method, h is hostname, p is protocol, n is port, l is path and e is web page's file type

Table 3. An example of three websites that have three file types and the number of corresponding files

Website	HTML	ASP	JPG
W_1	99	1	0
W_2	50	50	0
W_3	33	33	33

3.3 Continuity Score

The continuity score p measures how often a website is connected by a computer. It increases when the frequency of the HTTP requests to a website increases. In this paper, we leverage the histogram approach to calculate the continuity score of a websites.

Let a period of time be L, which is divided into k bins. The count function $C_s(i,j)$ is equal to 1 if the website i appears in the HTTP requests, which is located on j bin and invoked by computer s , otherwise it is equal to 0. Let M_s be the number of non-zero bins on computer s. The continuity score of website i is defined as follows:

$$p_{s,i} = \frac{C_i}{M_s} \quad \text{,where } C_i = \sum_{j=1}^{k} C_s(i,j) \tag{3}$$

For the example of Table 4, the timeline is divided into six parts. As the computer has no any HTTP request in t_5, M_s is equal to be 5. The continuity score of the websites is:

$p_{s,1} = (1+1+0+0+0)/5 = 0.4$
$p_{s,2} = (1+1+1+1+0)/5 = 0.8$
$p_{s,3} = (0+0+1+1+1)/5 = 0.6$

3.4 Ranking the Websites

The websites are ranked by the probability of being C&C servers. The higher the reversed diversity score and continuity score of a website, the higher probability it is a

Table 4. An example of calculating continuity score

W	t_1	t_2	t_3	t_4	t_5	t_6
W_1	1	1	0	0	0	0
W_2	1	1	1	1	0	0
W_3	0	0	1	1	0	1

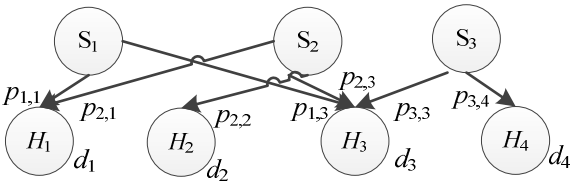

Fig. 3. Example of HTTP requests

C&C server. Meanwhile, the continuity score of a website is provided from the computers that have connected to it. We average them to be the final continuity score of a website. Therefore, the ranking score of a website *i* is defined as:

$$W_i = w_d \cdot d_i + w_p \cdot (\sum_{s=1}^{S} p_{s,i}) / S' \tag{4}$$

where w_d and w_p are the weight of reversed diversity score and continuity score and *S'* is the number of the non-zero $p_{s,i}$.

Fig. 3 is an example of HTTP requests, three source IP S_1, S_2 and S_3 connect to websites H_1, H_2, H_3 and H_4, each of which reversed diversity score is d_1, d_2, d_3, and d_4. As S_1 connects to website H_1 and H_3, it will give the continuity score $p_{1,1}$ and $p_{1,3}$ to H_1 and H_3 individually. S_2 connects to H_1, H_2 and H_3, so S_2 will give three continuity scores to the three websites. Therefore, the ranking score of H_1 and H_2 will be:

$$W_1 = w_d \cdot d_1 + w_p \cdot (p_{1,1} + p_{2,1}) / 2$$
$$W_2 = w_d \cdot d_2 + w_p \cdot (p_{2,2}) / 1$$

4 Design and Implementation

4.1 Design Overview

To realize the goal of responding to APTs quickly, two design issues are considered: 1) how to calculate the ranking score by E.q (4) quickly from the huge volume of HTTP logs (more than 30 GB per day); 2) how to extract the hit HTTP requests quickly when searching. To solving the first issue, we implement the ranking mechanism as MapReduce [26] jobs on Hadoop platform [27]. The second issue is solved by Lucene [28]. The high level working architecture of the system is shown in Fig. 4.

At first, the HTTP logs are duplicated, one for indexing and another for ranking. For indexing, the logs are filtered by a white list (known as legitimate website), the residual logs are indexed as the structure in Fig. 2 by Lucene with the keywords "time," "ip," "method," "protocol," "hostname," "port," "path," and "type". The querying mechanism is also completed by Lucence. The user can input the query statement with the keywords to look for searching the specific HTTP requests. Meanwhile, hostname and path provides "begins with," "ends with," "contains," and "equal to" operators, the other keywords only provide an "equal to" operator.

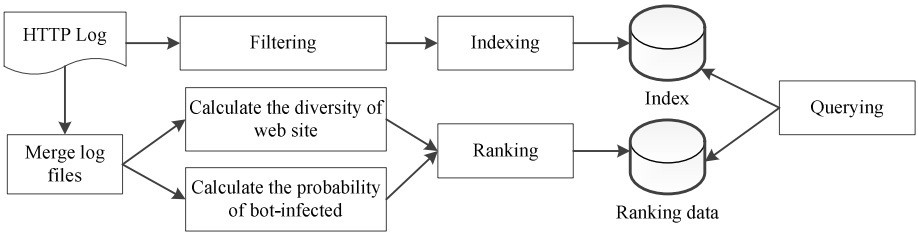

Fig. 4. High level working architecture

When users input a query to the system, the hostnames of the hit HTTP requests are extracted and ranked by the corresponding website's ranking score, and the system then shows the ranked results to the users.

4.2 The MapReduce Jobs

The ranking scores are calculated by the MapReduce jobs. In step 1, the map function extracts hostname, path and file type from the logs to be the keys. The reduce function in step 2 sorts and removes the duplicated keys. In step 3, as the files with the same file name but located in different directory are considered different files, the reduce function makes hostname and file type as the key, and the value starts from 0 and is added by one when reducer reads a record with the same key. The final value will be the number of files with the same file types of a websites. The reduce function in step 4 calculates the reversed diversity score of the websites by E.q (2) and output the results.

In step 5, the map function extracts timestamp, source IP and hostname to be the key, where timestamp is divided by the length of bin as described in section 3.3. The reduce function in step 6 sorts and removes the duplicated keys. In step 7, the reduce function ignores the hostname and makes timestamp and source IP as key to count the number of non-zero size bins M_s. Step 8 is another branch from step 6, it makes source IP and hostname as the key and calculates C_i. Step 9 refers to the outputs of step 7 and step 8 to calculate the continuity score of a website by E.q (3). Finally, step 10 calculates the ranking score by E.q (4) based on the results of step 4 and step 9.

5 Experiments and Evaluation

5.1 The Experiments Setup

To evaluate the effectiveness of our approach, we collected the proxy logs of the enterprise network consisting of several thousands of computers for two weeks, from October 11 to October 24, 2011, as the experimental data. The attributes of the experimental data are shown in Table 5. The experimental data consists of three C&C servers and the HTTP requests of five malware-infected computers which were found in December 14, 2011. Two reasons make us believe this is an APT attack: 1) the computer is a stepping stone and the footprints can be traced back to 8 months ago

Table 5. The attributes of the experimental data

Data Size	Source IP	Hostname	HTTP requests
440GB	19,633	267,962	273,195,451

and 2) the malware has the similar abilities, such as remote access, key logger, packet forward, DLL injection and so on, as described in [1, 6, 8, 29]. Meanwhile, as we don't know how many computers were infected actually in the data, in our best effort, we investigate all potential victims detected by our approach manually. The investigations are used for evaluating the accuracy of the search engine.

Finally, the performance of the search engine is a key point in this study. Five servers, one for the Hadoop master and four for the slaves, ran on CentOS 5.4 with a 2.26 GHz Intel Xeon CPU and 12 GB RAM. The version of Hadoop used was 1.0.1 and that of Lucene was 3.5. The log files for two weeks were fed to our system to evaluate the performance of indexes building, ranking scores calculation and query.

5.2 Determine the Weights

The weight w_d and w_p in Eq. (4) should be determined at first. We select the first week's logs to observe the reversed score and continuity score of the websites. We set L to be one week and k to be 1 hour. Fig. 5 depicts the distribution of the reversed diversity score and continuity score. Over 70% website's reversed diversity score is between 1~0.9. This is because 1) proxy servers can't log the details of HTTPS requests, the reversed diversity scores of all the HTTPS websites are equal to 1 and 2) some websites are connected for only a few times, so their reversed diversity scores are also equal to 1.Therefore, we ignored the HTTPS websites and the lower traffic websites (less than 10 HTTP requests).

The continuity score of most websites is lower than 0.1 (over 94%). The continuity score of the known C&C servers is between 0.8~1.0. However, the number of the websites with continuity score between 0.9 and 1.0 is larger than that of the websites with continuity score between 0.8 and 0.9. It is because some websites are connected

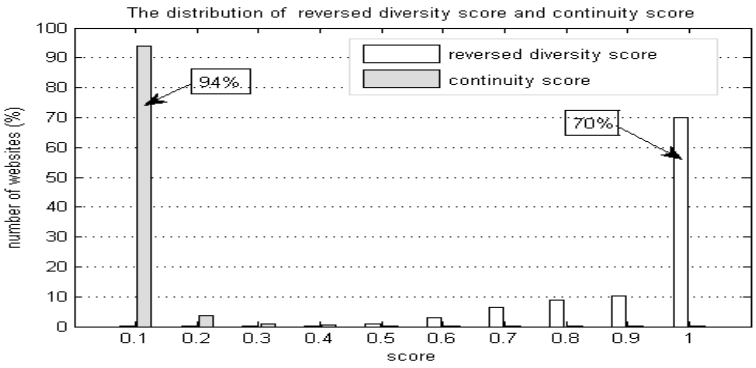

Fig. 5. The distribution of the reversed diversity score and continuity score

by a few clients and their IP also only connect to these websites. To deal with this problem, we remove the continuity score which is given from the source IPs with M_s<12, which means that in the two weeks the hours that a computer invoked HTTP requests are less than 12. Finally, as the impact of continuity score is larger than reversed diversity score, we set w_d to be 0.2 and w_p 0.8 in the further experiments.

5.3 Experiment Results

To evaluation the accuracy of the search engine, the experimental data is fed to the search engine. We set L to be two weeks and k to be one hour. The known malicious HTTP request r_m:=<*m=GET*, *p=http*, *n=443*, *l=/FC001/xxx*, *e*=- > invoked by APT malware are the references of the queries. To form the queries, we fix method, protocol and port and give three statements, where Q_1 = path contains "*FC001*", Q_2= type is equal to "-" and Q_3= any path and type. The ranking results in Table 6 show that the search engine can find the three known C&C servers by Q_1. Meanwhile, it also finds the other 6 unknown C&C servers. According to the found C&C servers, we find the other 11 potential victims which are real victims proved by investigations.

The search engine returns more websites by Q_2. Since the precise rate of being C&C server in top 20 websites is lower than the results by Q_1, Q_2 finds the other four C&C servers. The legitimate websites in top 20 are consisted of three types: web game, availability test of a website and flash video. In the test of Q_3, the search engine returns more websites than Q_1 and Q_2. Notably, the precise rate in top 20 websites of Q_3 is higher than that of Q_2. Meanwhile, the other five C&C servers are found by Q_3. This is because a new APT malware invokes the HTTP request *r'*$_m$:=<*GET*, *http*, *443*, */90ad.asp*, *asp*>, which is unknown before the experiments. For further investigation, 33 of 38 potential victims are proved being infected by the unknown malware.

In performance evaluation, the results in Table 7 show that the execution time of building index and calculating ranking score is more than 21 hours. This doesn't include the time for uploading log files to the servers. On average, the daily logs can be processed in 3.5 hours. We also test the system responding time by above three queries. The results show that the three queries can be completed in 2 second. The execution time of Q_2 and Q_3 are longer than that of Q_1. This is because the more websites hit, the more time the search engine requires to display them.

Table 6. The ranking results of the three queries

Queries	Q_1	Q_2	Q_3
Number of hit websites (excluding the three known C&C servers)	9	39	98
The rank of the three known C&C servers	1, 2, 7	1, 2, 7	1, 2, 10
The number of C&C servers in top 20 websites	9	13	18
The precise rate of C&C servers in top 20 websites	100%	65%	90%
The false positive rate of C&C servers in top 20 websites	0%	35%	10%
Number of potential victims	11	48	38
Number of real victims	11	24	33

Table 7. The performance of the search engine

MapReduce job	Execution Time
Building index	21 hours
Calculating ranking score	27 hours
Querying by Q_1	< 0.6 second
Querying by Q_2	< 1.6 second
Querying by Q_3	< 2 second

6 Conclusion

This paper develops a search engine on Hadoop platform to search potential C&C servers and victims for APT investigation. In the real APT investigation, we prove that the search engine can rank the known C&C servers in top 10 websites. The search engine also finds out the other C&C servers and potential victims. Meanwhile, the responding time of each query is less than 2 second.

The future work may include: 1) if the malware communicates with the C&C server through HTTPS, our approach may fail to find them. The statistical-based approaches may be a chance to improve the ranking mechanism; 2) the fast flux botnet changes the domain name frequently, so the websites may be ignored because of fewer HTTP requests. The ranking mechanism may be improved by introducing other supplemental attributes, such as TTL value or domain name location.

Acknowledgments. The authors would like to thank reviewers' helpful comments. This research is partially supported by the Information & Communication Security Lab, Telecommunication Laboratories, Chunghwa Telecom co., Ltd, the National Science Council of Taiwan, ROC under Grant No. NSC101-2218-E-008-004.

References

1. Daly, M.K.: The Advanced Persistent Threat. In: USENIX (ed.) 23rd Large Installation System Administration Conference. USENIX, Baltimore (2009)
2. http://www.damballa.com/knowledge/advanced-persistent-threats.php
3. HPGary, inc., http://www.issa-sac.org/info_resources/ISSA_20100219_HBGary_Advanced_Persistent_Threat.pdf
4. Juels, A., Yen, T.F.: Sherlock Holmes and The Case of the Advanced Persistent Threat. In: Proceedings of the 5th USENIX Conference on Large-Scale Exploits and Emergent Threats, p. 2. USENIX Association, San Jose (2012)
5. Winder, D.: Persistent and Evasive Attacks Uncovered. Infosecurity 8, 40–43 (2011)
6. McAfee, http://www.mcafee.com/us/resources/white-papers/wp-operation-shady-rat.pdf
7. Liu, S.-T., Chen, Y.-M., Hung, H.-C.: N-Victims: An Approach to Determine N-Victims for APT Investigations. In: Lee, D.H., Yung, M. (eds.) WISA 2012. LNCS, vol. 7690, pp. 226–240. Springer, Heidelberg (2012)
8. SANS Institute, http://www.sans.org/reading_room/whitepapers/malicious/detailed-analysis-advanced-persistent-threat-malware_33814

9. Li, F., Lai, A., Ddl, D.: Evidence of Advanced Persistent Threat: A case study of malware for political espionage. In: 2011 6th International Conference on Malicious and Unwanted Software, pp. 102–109. IEEE, Fajardo (2011)
10. Liu, S.T., Chen, Y.M.: Retrospective Detection of Malware Attacks by Cloud Computing. In: 2010 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, pp. 510–517. IEEE, Huangshan (2010)
11. Goebel, J., Holz, T.: Rishi: identify bot contaminated hosts by IRC nickname evaluation. In: Proceedings of the First Conference on First Workshop on Hot Topics in Understanding Botnets, p. 8. USENIX Association, Cambridge (2007)
12. Brustoloni, J., Farnan, N., Villamarin-Salomon, R., Kyle, D.: Efficient Detection of Bots in Subscribers' Computers. In: IEEE International Conference on Communications, pp. 1–6. IEEE, Dresden (2009)
13. Rieck, K., Schwenk, G., Limmer, T., Holz, T., Laskov, P.: Botzilla: detecting the "phoning home" of malicious software. In: Proceedings of the 2010 ACM Symposium on Applied Computing, pp. 1978–1984. ACM, Sierre (2010)
14. Warmer, M.: Detection of web based command & control channels. Mathematics and Computer Science. University of Twente (2011)
15. Perdisci, R., Lee, W., Feamster, N.: Behavioral clustering of HTTP-based malware and signature generation using malicious network traces. In: Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation, p. 26. USENIX Association, San Jose (2010)
16. Gu, G., Zhang, J., Lee, W.: BotSniffer: Detecting botnet command and control channels in network traffic. In: Proceedings of the 15th Annual Network and Distributed System Security Symposium, San Diego, CA (2008)
17. Larson, R.E.: CCSP: Cisco Certified Security Professional Certification All-in-One Exam Guide. McGraw Hill, New York (2003)
18. Thonnard, O., Bilge, L., O'Gorman, G., Kiernan, S., Lee, M.: Industrial Espionage and Targeted Attacks: Understanding the Characteristics of an Escalating Threat. In: Balzarotti, D., Stolfo, S.J., Cova, M. (eds.) RAID 2012. LNCS, vol. 7462, pp. 64–85. Springer, Heidelberg (2012)
19. Sood, A., Enbody, R.: Targeted Cyber Attacks - A Superset of Advanced Persistent Threats. IEEE Security & Privacy 99, 1–3 (2012)
20. Sood, A., Enbody, R., Bansal, R.: Cybercrime: Dissecting the State of Underground Enterprise. IEEE Internet Computing 99, 1 (2012)
21. Baize, E.: Developing Secure Products in the Age of Advanced Persistent Threats. IEEE Security & Privacy 10, 88–92 (2012)
22. Tankard, C.: Advanced Persistent threats and how to monitor and deter them. Network Security, 16–19 (2011)
23. Gordon, T.: APTs: a poorly understood challenge. Network Security, 9–11 (2011)
24. Dempsey, K., Chawla, N.S., Johnson, A., Johnston, R., Jones, A.C., Orebaugh, A., Scholl, M., Stine, K.: Information Security Continuous Monitoring (ISCM) for Federal Information Systems and Organizations. National Institute of Standards and Technology U.S. Department of Commerce, U.S.A. (2011)
25. Jost, L.: Entropy and diversity. Oikos 113, 363–375 (2006)
26. Dean, J., Ghemawat, S.: MapReduce: Simplified data processing on large clusters. Communications of the ACM 51, 107–113 (2008)
27. http://hadoop.apache.org/
28. http://lucene.apache.org
29. SANS Technology Institute, https://www.sans.edu/student-files/projects/JWP-Binde-McRee-OConnor.pdf