In-Host Communication Pattern Observed for Suspicious HTTP-Based Auto-Ware Detection

Manh Cong Tran*, Yasuhiro Nakamura
Department of Computer Science, National Defense Academy, Yokosuka 239-0811, Japan.

* Corresponding author. Tel.: +818096598686; email: manhtc@gmail.com
Manuscript submitted November 1, 2014; accepted May 10, 2015.
doi: 10.17706/ijcce.2015.4.6.379-389

Abstract: In consequence of the growing cyber security threats, normal users and also system administrators are advised to closing inward ports and permitting outgoing communication only over selected protocols. In many decades, the flexibility and interoperability of HTTP make users progressively explore it in a much wider range of applications. Therefore, HTTP is always allowed on the network perimeter. HTTP-based applications could be classified into two types of Internet accesses: passive and active HTTP access applications. Passive type application (i.e. browsers) has just generated requests on users' demands, so users can clarify and control what content they will access and accomplish. On the contrary, active type is called automatic software (auto-ware), which allows completely or partly automatically access to its servers without users' intention. Auto-ware could be normal applications such as virus defining or operating system updating, but also are abnormal processes such as botnet, worms, virus, spywares, and advertising software (adware). Therefore, auto-ware, in a sense, consumes network bandwidth, and it might become internal security threats. Detection of suspicious auto-ware and its traffics are challenge work because the malicious traffic merges sufficiently with legitimate HTTP traffic. In this paper, based on the observation of communication pattern of HTTP auto-ware, it is proposed a detection method of HTTP-based Auto-ware. The experiment results show that the method is useful for host-based detection application.

Key words: HTTP-based malware, malware detection, network security management, periodic communication.

1. Introduction

Because of the growing of cyber security threats, normal users and also system administrators protect their networks by closing inward ports and permitting outgoing communication only over selected protocols such as HTTP. For that reason, in many decades, Internet application developers have trend to develop their web services for reaching users. However, in web environment, digital spies and thieves can cover their identities, conceal their physical locations, and create the malicious code from side to side. Therefore, cyber criminals or the Internet spiders also take advantage of web technology and using it as a medium for communication to lurk malicious software or variety of forbidden or illicit activities. An HTTP-based malware infected computer is controlled by either infrastructure can be instructed to perform malicious HTTP activity such as send Spam or download shellcode from secret servers [1].

HTTP-based applications can be classified into two categories: passive and active HTTP access application. Passive HTTP access application, i.e. a browser like Internet Explorer, has just generated

requests on demand of users to access websites/address which users want or need. So users can clarify and control what content they will access and accomplish. In the opposite way, active type is called automatic software (auto-ware), which allows completely or partly automatically request to their servers without users' intention.

Auto-ware can be classified into three types as illustrated in Fig. 1: **normal software** such as anti-virus updater, mail client, browser's toolbar; **greyware** encompasses adware, spyware, joke programs; **malicious software** acting as HTTP-based botnet, worm or trojan horses. For keeping the update from servers, HTTP-based auto-ware periodically generate legal traffic and requests to their servers as shown in Fig. 2. By this way, suspicious HTTP-based auto-ware might access to unknowing servers without user's intention. The distinction of normal and malicious activities from HTTP traffic is becoming tougher because the malicious requests merges adequately with legitimate HTTP traffic. Therefore, it is complicated to discriminate and impossible for network security devices to block malicious traffic. Besides that, malicious auto-ware can penetrate into users computer in dozen of ways such as plugged into free software, drive-by download attack, or spam link. Consequently, users might not control what or how many auto-wares are installed in their computers.

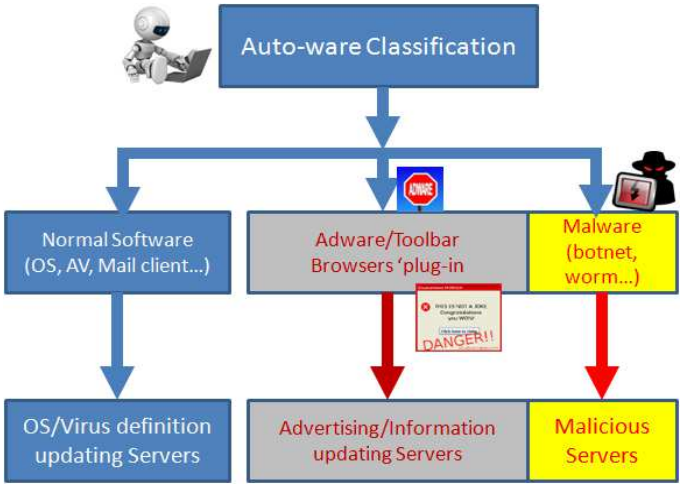

Fig. 1. Auto-ware classification.

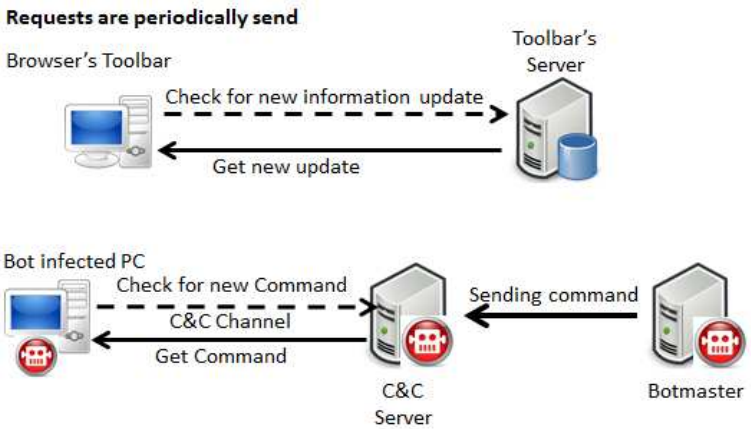

Fig. 2. Auto-ware periodically automatically communicates with their servers to keep up the renewal.

Despite the fact that all types of auto-ware periodically automatically communicate with their servers to keep up the maintenance, there still the difference in the way communication between normal HTTP-based software and suspicious HTTP-based software in some characteristics that are indicated as

periodic access and **access rate.** In this paper, by observing the variation of above two features of HTTP-based application communication, a simple and new host-based suspicious auto-ware detection method is proposed. Through which, two parameters, *Auto-ware Score* and *Suspicious Score* are defined. The method is examined on the real traffic and produces a good result in detection. By in-host type method, it is helpful for users and also administrators in control the auto-ware in their computers and networks.

The rest of the paper is structured as follows: the related work is reviewed in Section 2; Section 3 reviews the characteristics between many auto-wares and presents the proposed method; Section 4 describes experimental results, and Section 5 is conclusions and future work.

2. Related Work

A noticeable number of studies on malicious detection have adopted passive analysis by collecting the network traffic for a specific period. This section will review some of the recent related work in the field of malicious and abnormal HTTP activity detection.

Ashley has suggested a method for detecting potential HTTP C&C activity based on repeated HTTP connections to a C&C website [1]. According to this, an algorithm is proposed by for detecting HTTP polling activity. First, the "enough" time interval values of HTTP GET requests is determined. After that, k-means algorithm is applied to find a single cluster contains "most" data values. Finally, HTTP polling activity is pointed out by computing the standard deviation of the cluster. If it is "small enough", the HTTP GET requests are suspect of HTTP-based C&C traffic. The author has demonstrated that the method is able to detect automated polling activity to a website. However, the approach is just using the evaluation for periodic access. For that reason, the result is noticed with a caution of accuracy.

Using signature-based techniques, W. Lu *et al*. in [2] proposed a new hierarchical framework to automatically discover botnet on a large-scale Wi-Fi ISP network, in which the network traffic is classified into different application communities by using payload-signature. These signatures were used to separate known traffic from unknown traffic in order to decrease the false alarm rates. Like other signature-based techniques the proposed classifier is less effective as it is unable to identify new or encrypted patterns [3].

The investigating of suspicious HTTP-based software at the network scope is had proposed by most researchers [2], [4]-[6]. But the increase in sophistication and stealth-ness of auto-ware (e.g. botnet) traffic along with growth in volume of traffic at network edges makes this detection approach more arduous [7]. In addition, by any means, the examining an infected host at the local scope is also importance and potential and its advantages seem to be disregarded. Noticeably, network-based malicious HTTP-based software investigation forms based on communication protocol/request information acquired from malware infected machines. Therefore, host-based and network level investigation are in direct relevance [8].

At host level, Jae-Seo *et al*. in [9] introduced a parameter based on one of the pre-defined characteristics of HTTP-based botnet. They suggested a Degree of Periodic Repeatability (DPR) to show the pattern of regular connections (i.e. PULL style) of HTTP-based botnet to certain servers. Also at host level, Meisam Eslahi *et al.* in [3] proposed many filter parameters to reduce the false alarm in botnet detection. However, Jae-Seo *et al*. [9] and Meisam Eslahi *et al*. [3] do not give the details of formula or method to calculate and measure each parameter.

In this paper, based on the variation of periodic access and access rate, a measurement is proposed in detection suspicious HTTP-based auto-ware at host level.

3. Methodology

In this section, HTTP-based auto-ware communication characteristics have been reviewed and the detail of method is also interpreted.

For the purpose of reviewing the different of normal and suspicious auto-ware based on **periodic access** and **access rate** characteristics, four of HTTP-based auto-ware are selected, they are included:

- Two normal toolbars (Toolbar 1, 2): these toolbars are browser's software help people access to their favorite services.
- A cloud service backup tool (Cloud Drive Sync Tool): These kinds of tools will sync users' files from clients to cloud and vice versa. It helps users access the files or documents from anywhere based on their using cloud service, such as Dropbox or iCloud.
- A Malicious Zeus botnet with its C&C server: Zeus is HTTP-based bot. It has reportedly infected over 3.6 million computers in the United States. Zeus can be updated and directed by the botmaster through C&C channel [10].

This set of software is installed in a computer and their requests to server are captured and observed in two days. The data collection information is summarized in Table 1. As can be seen on Table 1, the communicated requests to server activities of malicious Zeus botnet are not as much as normal auto-ware.

Table 1. Data Collection Information

HTTP-Based Auto-Ware	Number of GET requests	Data collection length (2014)
Toolbar 1	1351	From 21:35:20 Sep 28 To 21:28:06 Sep 30 About 47 hours
Toolbar 2	1205	
Cloud Drive Sync Tool	4079	
Zeus Bot	220	

3.1. HTTP-Based Auto-Ware Communication Characteristics

In this section, by using real above data, two **periodic access** and **access rate** characteristics are compared between normal and malicious auto-ware.

- **Periodic access**: HTTP-based malicious software, such as botnet, which follow the PULL style where they periodically steadily connect to their server (i.e. command and control server) by GET requests with an interval in order to get the commands and updates [1], [3]-[5].

The observation data for this feature are shown in Fig. 3. In that X axis is the number of requests and Y axis represents the time different in second of two requests side by side. The graphs in Fig. 3 illustrate the GET requests interval of selected auto-ware in attempt to keep communication to their servers.

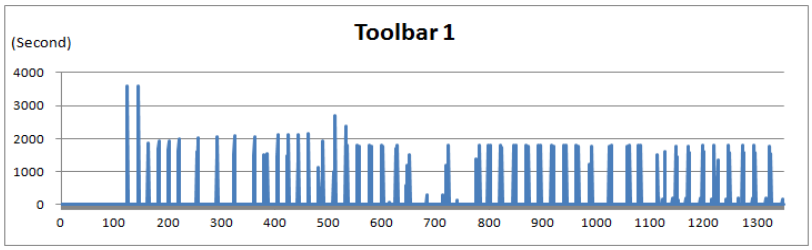

Fig. 3a. Toolbar 1's periodic access data observation.

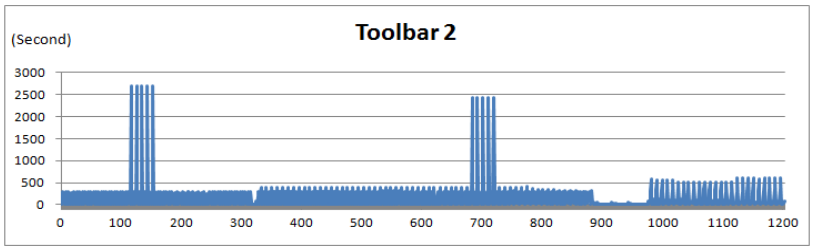

Fig. 3b. Toolbar 2's periodic access data observation.

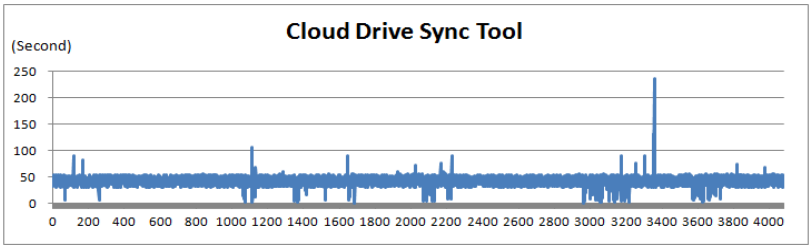

Fig. 3c. Cloud Drive Sync Tool's periodic access data observation.

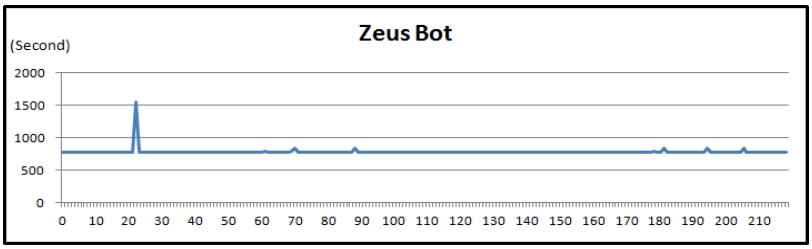

Fig. 3d. Zeus bot's periodic access data observation.

Toolbar 1 and Toolbar 2's graphs (respectively in Fig. 3a, Fig. 3b) are looked similar with each other. They have many long inactive durations which shows as high peaks in diagrams. The Cloud Drive Sync Tool's graph (in Fig. 3c) express high intensity activities by big number of requests. The frequency of requests are also high and interval fluctuates around 50 seconds. Like Toolbar 1 and Toolbar 2, Cloud Drive Sync Tool shows the long inactive durations. In summary, these three normal auto-ware illustrates that they do not keep communication to their servers with steady periodic or interval of time. In contrary, the diagram of malicious Zeus bot (in Fig. 3d) shows that it owns the most steady interval with only some peaks and the frequency of requests is at normal level with about 800 seconds between two requests. The requests are equally distributed during the running time.

- **Access rate:** Normal automatic software (e.g. updater and downloader) transmits a similar periodic pattern of traffic that has been generated within a short period of time. A suspicious software does not generate bulk data transfer [3], [6].

The observation data of the feature are shown in Fig. 4. In that X axis is the index of 1 hour time slot (about 47 slots, see Table 1), Y axis represents the number of GET requests in a slot of time. The graphs in Fig. 4 illustrates the fluctuation of number of GET requests in each time slot for each selected auto-ware in attempt to keep communication to their servers.

Once again, Toolbar 1 and Toolbar 2's graphs (respectively in Fig. 4a, Fig. 4b) are looked similar with each other with the high fluctuation in the graphs. They send many requests in a short of time which shown as peaks. Normal fluctuation in diagram is represented in The Cloud Drive Sync Tool's graph (Fig. 4c), however the number of requests in each slot always keep at high level around 80 requests. Can be conducted that three normal auto-ware keep the high variation access rate in communication with their servers. In contrast that, the graph of Zeus bot, in Fig. 4d, illuminates that in Zeus bot's requests, there is almost no variation in requests number in each time segment just around 4-5 requests and active time is equally distribution in running time.

In both observations of features, the negligible variation of suspicious auto-ware characteristics can be seen. If the total data collection is divided into N equivalent segments from t_1 to t_N with time duration Δ, the distribution of normal and suspicious auto-ware activities through two properties mentioned above can be illustrated as in Fig. 5.

Based on this observation, a method to detect suspicious HTTP-based auto-ware is proposed and expressed in next section.

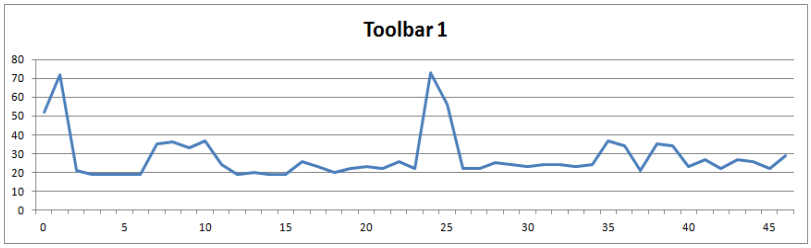

Fig. 4a. Toolbar 1's access rate data observation.

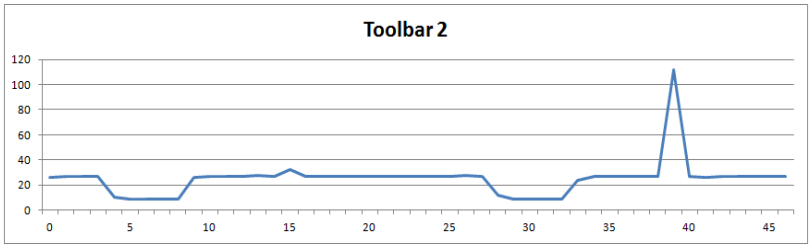

Fig. 4b. Toolbar 2's access rate data observation.

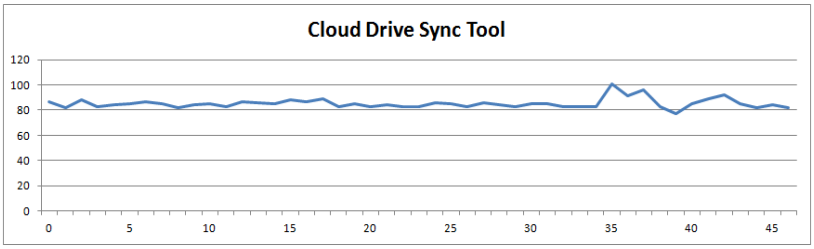

Fig. 4c. Cloud Drive Sync Tool's access rate data observation.

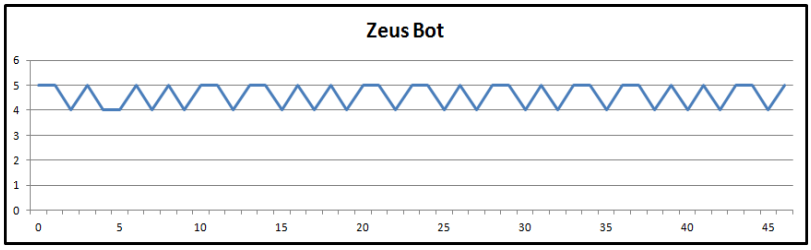

Fig. 4d. Zeus bot's access rate data observation.

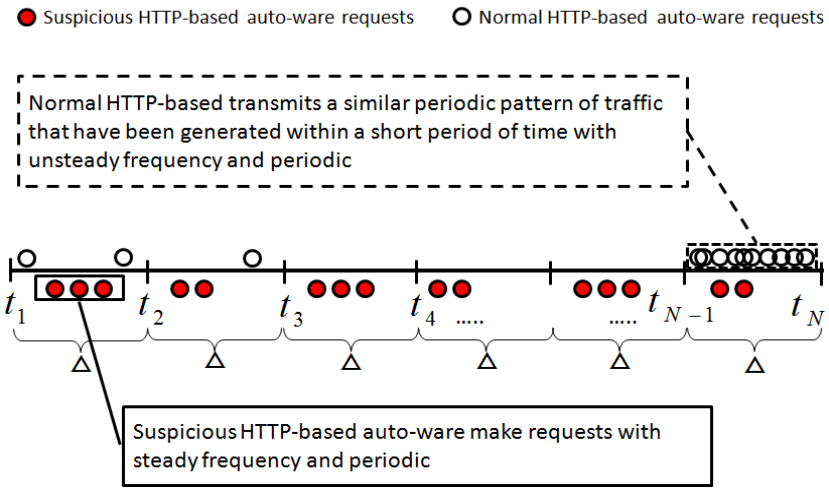

Fig. 5. The distribution of HTTP-based normal and suspicious auto-ware.

3.2. Proposed Method in Suspicious HTTP-Based Auto-Ware Detection

The method is proposed based on examining the variation of two features which are reviewed in Section 3.1. Correspondingly, two parameters are introduced, *Auto-ware Score* and *Suspicious Score*, they are respectively shows the variation periodic access and access rate requests.

Standard deviation is used usefully in measuring the amount of variation or dispersion from the average of sequences. For that meaning, the calculation approach of each parameter is using standard deviation σ formula [11]. σ of a vector $X=(x_1, x_2,..., x_n)$ is illustrated as in (1):

$$\sigma(X) = \sqrt{\frac{\sum_{j=1}^{n}(\bar{x} - x_j)^2}{n-1}} \tag{1}$$

General imagination of method is show in Fig. 6. In the remainder of the section, all parts of flow will be gone throw. For that purpose, assume that the total data collection is divided into N equivalent segments from t_1 to t_N with time duration Δ.

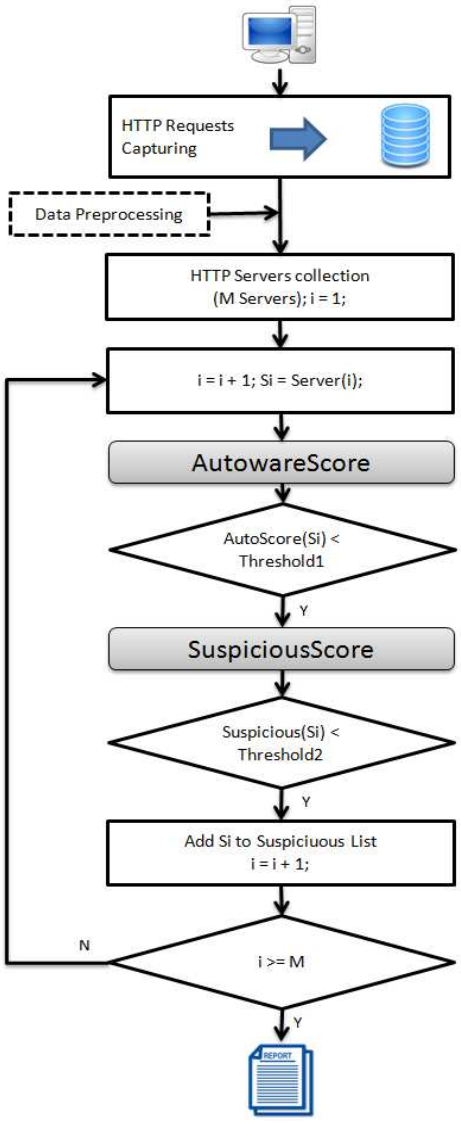

Fig. 6. Overall flow of proposed method.

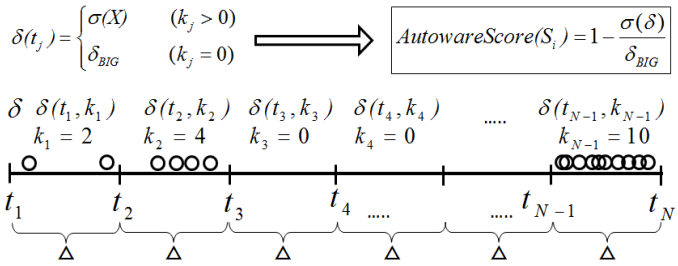

Fig. 7. Imagination of *auto-ware score* calculation for a server S_i.

Preprocessing step helps to reduce or filter out unnecessary information. There is many ways to do this, for example, if the number of URIs connects to a server is too small in a long period of time, that server is not a candidate for a suspicious communication (e.g. entire data collecting period) because botnet/malicious software are designed to perform bigger tasks and much faster than humans, hence they do not generate brief traffic [3], [6]. Another simple way is using a good white-list.

Auto-ware Score is determined by observing the periodic access of client to server with some following steps, a calculation sample of *Auto-ware Score* for a server S_i is illustrated in Fig. 7.

- For a server S_i, with each segment from t_j to t_{j+1}, define k_j is number of requests in that segment, $X=(x_1,x_2,...,x_{k_j-1})$ is vector of time interval between every two adjacent requests, $\bar{x}$ is mean of vector *X*, the $\delta(t_j)$ is calculated as in (2).

$$\delta(t_j)=\begin{cases}\sigma(X) & (k_j>0)\\ \delta_{BIG} & (k_j=0)\end{cases} \tag{2}$$

In the case of $k_j>0$, $\delta(t_j)$ is standard deviation of vector *X* by using formula in (1). The constant δ_{BIG} is a big number.

- A vector $\delta=(\delta(t_1),\delta(t_2),...,\delta(t_{N-1}))$ is accomplished after last step, $\sigma(\delta)$ is calculated based on (1) and reform as in (3) below:

$$\sigma(\delta)=\sqrt{\frac{\sum_{j=1}^{N-1}(\bar{\delta}(\bar{t})-\delta(t_j))^2}{N-2}} \tag{3}$$

- *Auto-ware Score* of server S_i is determined as in (4), one more time (1) is used.

$$Auto\text{-}ware\ \ Score\ \ (S_i)=1-\frac{\sigma(\delta)}{\delta_{BIG}} \tag{4}$$

δ_{BIG} is chosen big enough which $\delta_{BIG}\geq\max(\sigma(\delta))$ for standardization $0\leq$ *Auto-ware Score* $(S_i)\leq 1$.

Suspicious Score is determined by observing the access rate of client to server. As is defined above,

$k = (k_1, k_2, ..., k_{N-1})$ is a vector which presents the number of requests in each time segment. The *Suspicious Score* of server S_i is standard deviation of *k* which is formulated as in (5).

$$Suspicious\ \ Score\ \ (S_i) = \sigma(k) \tag{5}$$

Threshold 1 and Threshold 2 are can be set as configurable parameters, depending on typical traffic on a network.

4. Experimental Results

The experiment has been done by using the data which are described in Table 1 of Section 3.1. (PC1) and web access data of 3 other clients (PC2, PC3, PC4), the information of experimental data is represented in Table 2, these data is after data preprocessing step, as in Fig. 6.

Exception the applications which are installed in PC1 is already known as can be seen in Table 1, all the information about the kind of applications using in PC2, PC3 and PC4 are unknown before applying the proposed method.

An host-based Windows application is developed to apply the proposed approach with above data. All the threshold values has been chosen as description in Fig. 6 and Section 3.2, the time duration Δ will be 1 hour, δ_{BIG} has been automatically chosen as 30 after calculate $\sigma(\delta)$ by using (3). Threshold 1 and Threshold 2 are can be set as configurable parameters, depending on typical traffic on a network. In this implementation, Threshold 1 = 0.9 and Threshold 2 = 0.5 are chosen for experimentation purpose.

The experimental results have been summarized in Table 3. For evaluation the data of PC2, PC3, PC4 the application servers (column servers in Table 3) are referenced with users of these PCs.

The results show that all auto-wares are well detected by the parameter *Auto-ware Score* of method combined with Threshold 1, and Zeus bot is detected as a suspicious auto-ware by *Suspicious Score* in combined with Threshold 2.

This paper has proposed an approach to detect suspicious HTTP-based auto-ware merely through observing communication pattern features of natural HTTP traffic at host level. Compare to traditional using signature method, this method can detect new type of suspicious which having periodic communication to their server.

Table 2. Experimental Data Collection Information

PCs	Number requests	Data Length	Descriptions
PC1	60,439	2 days	Zeus Bot, 2 toolbars, 1 cloud drive sync tool are installed.
PC2	18,889	1 day	No information
PC3	10,094	1 day	No information
PC4	7036	1 day	No information

The proposed methods are evaluated based on the detection of real data and its efficiency in the detection of a HTTP-based botnet and other normal auto-wares.

The experimental results show that HTTP-based auto-ware is detected well by the proposed method. The method can be developed in the future to be a real time detection application with the working flow as in Fig. 8. With this flow, users can proactively to control the auto-ware activities in their computers and also promptly prevent suspicious auto-ware.

Proposed method also will be needed improvement for detection in various working models of auto-ware such as decentralization auto-ware, encrypted auto-ware traffic, or the malicious auto-ware communicate to server with randomized interval.

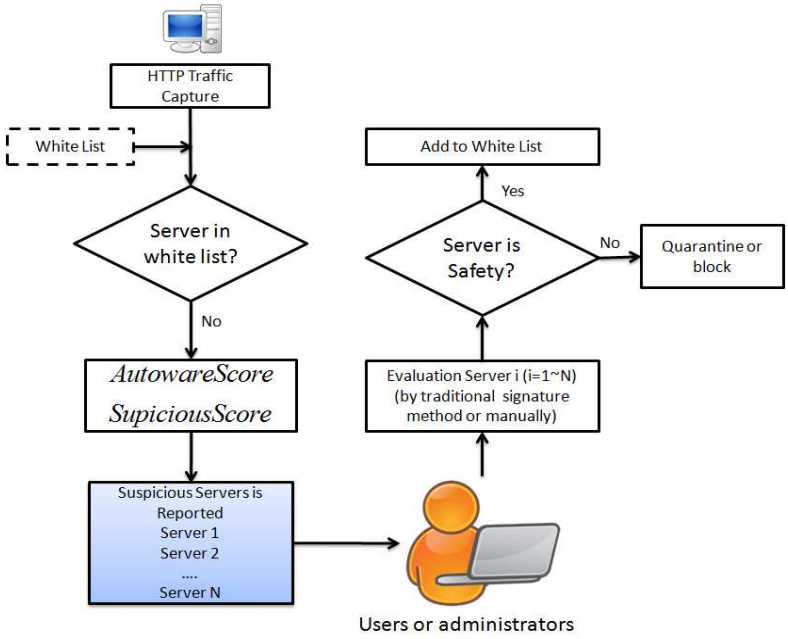

Fig. 8. Proposal flow for real time HTTP-based suspicious auto-ware detection.

Table 3. Experimental Results

PC	Servers	Auto-ware Score	Suspicious Score	Detection Results
PC1	Zeus Bot C & C server	0.994	0.49	Suspicious Auto-ware
	Toolbar1 server	0.989	3.84	Auto-ware
	Toolbar2 server	0.940	12.35	Auto-ware
	Cloud Drive Sync Tool server	0.997	14.89	Auto-ware
PC2	Anti-virus software Server	0.947	13.52	Auto-ware
PC3	Cloud Drive Sync Tool server	0.965	23.9	Auto-ware
	Anti-virus software Server	0.967	176.18	Auto-ware
PC4	All servers	≤0.79	≤35.03	No Auto-ware

References

[1] Ashley, D. (2011). An algorithm for http bot detection. University of Texas at Austin, Information Security Office.

[2] Lu, W., Tavallaee, M., & Ghorbani, A. A. (2009). Automatic discovery of botnet communities on large-scale communication networks. *Proceedings of the 4th International Symposium on Information, Computer, and Communications Security* (pp. 1-10). Sydney: Australia.

[3] Eslahi, M., Hashim, H., & Tahir, N. M. (2013). An efficient false alarm reduction approach in HTTP-based

botnet detection. *Proceedings of 2013 IEEE Symposium on Computers & Informatics* (pp. 201–205). Langkawi: Malaysia.

[4] Gu, G., Perdisci, R., Zhang, J., & Lee, W. (2008). BotMiner: Clustering analysis of network traffic for protocol and structure independent botnet detection. *Proceedings of the 17th Conference on Security Symposium* (pp. 139-154). San Jose, USA.

[5] Koo, T. M., Chang, H. C., & Wei, G. Q. (2011). Construction P2P firewall HTTP-botnet defense mechanism. *Proceedings of the IEEE International Conference on Computer Science and Automation Engineering (CSAE)* (pp. 33-39). Zhangjiajie, China.

[6] Strayer, W. T., Walsh, R., Livadas, C., & Lapsley, D. (2006). Detecting botnets with tight command and control. *Proceedings of the 31st IEEE Conference on Local Computer Networks* (pp. 195-202). Tampa, Florida, USA.

[7] Soniya, B., & Wilscy, M. (2013). Using entropy of traffic features to identify bot infected hosts. *Proceedings of 2013 IEEE Recent Advances in Intelligent Computational Systems (RAICS)* (pp. 13-18). Trivandrum, India.

[8] Farhood, F. E., & Payam, V. (2012). Real-time botnet command and control characterization at the host level. *Proceedings of 6th International Symposium on Telecommunications (IST'2012)* (pp. 1005–1009), Manchester, UK.

[9] Lee, J.-S., Jeong, H. C., Park, J.-H., Kim, M., & Noh, B.-N. (2008). The activity analysis of malicious HTTP-based botnets using degree of periodic repeatability. *Proceedings of the International Conference on SecurityTechnology (SECTECH)* (pp. 83-86). Hainan Island, China.

[10] Binsalleeh, H., Ormerod, T., Boukhtouta, A., Sinha, P., Youssef, A., Debbabi, M., & Wang, L. (2010). On the analysis of the Zeus botnet crime-ware toolkit. *Proceedings of 2010 Eighth Annual International Conference on Privacy Security and Trust (PST)* (pp. 31-38). Ottawa, ON, Canada.

[11] Erwin, K. (2011). *Advanced Engineering Mathematics* (10th ed. pp. 1011-1015), ch. 24. Wiley & Sons, Inc.

Manh Cong Tran was born in Namdinh, Vietnam in 1981. He graduated and got his master degree in computer science from Le Quy Don Technical University of Vietnam in 2007.

He is currently a PhD candidate in the Department of Computer Science, National Defense Academy, Yokosuka, Kanagawa, Japan. His current research interests include network traffic classification/analysis and anomaly/malicious detection. He also was employed in Le Quy Don Technical University, Hanoi, Vietnam.

Yasuhiro Nakamura was born in Tokyo, Japan in 1959. He graduated from NDA (National Defense Academy), Japan in 1982, and received the Ph.D. engineering degree from Keio University, Tokyo, Japan in 1990.

He was a major of Japan ground self-defense force until 1995, and has been a professor in the Department of Computer Science, NDA since 2011. He is currently the chairperson of the Department of Computer Science from 2013. His current research interests are in computer network security and network system management.

Dr. Nakamura is a member of IEICE (The Institute of Electronics, Information and Communication Engineers), IPSJ (Information Processing Society of Japan) and IIEEJ (The Institute of Image Electronics Engineers of Japan).