Development of White List Based Autonomous Evolution of Defense System for RAT Malware

Tomohiro Shigemoto*†, Shota Fujii*, Ichiro Kuriama*, Tetsuro Kito*,
Hirofumi Nakakoji*, Yasuhiro Fujii*, Hiroaki Kikuchi†

*Research & Development Group
Hitachi, Ltd.
292 Yoshida-cho, Totsuka-ku Yokohamashi
Kanagawa, Japan

†Graduate School of Advanced Mathematical Sciences
Meiji University
4-21-1 Nakano, Nakano-ku
Tokyo, Japan

***Abstract*—In order to minimize the damage caused by Remote Access Tool (RAT) malware used in targeted attacks, various countermeasures such as the black list and white list approaches have been developed. In the black list approach, we manage servers where the malware tried to communicate and block the malware communications. However, recently malware has been frequently changing C&C servers, and it is difficult to catch up the change of servers. In the white list approach, we permit only servers already known to be safe. However, all communications not on the white list are blocked, and this can have a disruptive effect on business. In this work, we propose a new Autonomous Evolution of Defense System based on white list. When unknown communication occurs, the proposed system requires an additional authentication, which a malware cannot pass through. Our system can be useful countermeasures against malware without disruption of business activities.**

Keywords—malware, white list, CAPTCHA

I. Introduction

As the malware used in targeted attacks has more sophisticated in recent years, the existing inbound measures have failed to detect attacks and allowed incursions into an organization. An attacker often use RAT malware for controlling the hosts on a target company. RAT is a program that provide the capability to allow covert surveillance or the ability to gain unauthorized access to a host. After RAT infection, he or she orders the infected host to intrude into another host on the internal network, and then executes a RAT malware, hacking tool, command or other program on the host, and finally gains confidential information from the targeted company. For example, in June 2015, the Japan Pension Service was compromised using the RAT malware EMDIVI [1] to exploit confidential information, and more than 44 organizations were involved in the same type of attack [2].

In response to such sophisticated attacks, various countermeasures are proposed. Blocking connect-back traffic with proxy authentication is recommended [3], but recently some malware have the ability to pass authentication proxy [4] (Problem 1). Some appliances use the Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA) to block malware communications. CAPTCHA is a type of challenge-response test used in computing as an attempt to ensure that the response is generated by person, and it has been widely used as a security measure to restrict access from automated program. However, these appliances always request CAPTCHA, so it has a disruptive effect on daily work (Problem 2). A method of protection against malicious social networking sites using proxies was proposed by Tsai et al. [5], where the social networking site is scanned before a user accesses it and if the site is suspect, the proxy disconnects access and sends a warning message to the user. However, this method cannot block communications when a malware uses an unknown site (Problem 3).

To solve these problems, we proposed a system called the Autonomous Evolution of Defense (AED) [6]. The AED requires users an additional authentication before the proxy server when a user tries to access suspicious URLs provided from the dynamic malware analysis system. Therefore, even if the AED gives incorrect information concerning the URL of a benign site, it allows Internet connection for users who pass authentication without causing any disruption to business. The AED also blocks any Internet connection that has been accessed by a machine program, e.g., malware. The AED addressed the problem 1 and 2, however, had the problem 3. Recently malware has been using the Domain Generation Algorithm (DGA) [7] and changes C&C servers frequently, and it is difficult to follow the change of servers.

In this work, in order to address the problem 3, we improve an Autonomous Evolution of Defense System based on white list (WAED) that blocks communications of malware without a suspicious list. The WAED places safe sites on a white list and requires users an additional authentication to the proxy server when a user tries to access an Internet site that is not in the white list. In this way, the WAED takes countermeasures against attacks that utilize unknown sites and address problem 3. The advantage of the proposed system is that it blocks malware communications even if they use an unknown site and does so without disruption of business activities. A comparison with existing countermeasures is given in Table 1.

In Section II of this paper, we propose the WAED. In Section III, we explain the experiments performed to evaluate the proposed system, and in Section IV, we discuss the optimum user size for the proposed system. We conclude in Section V with a brief summary and mention of future works.

978-1-5386-7380-5/18/$31.00 ©2018 IEEE
DOI 10.1109/AsiaJCIS.2018.00024

TABLE I. COMPARISON WITH EXISTING COUNTERMEASURES

Countermeasure	*Pass auth. (problem 1)*	*Disruption (problem 2)*	*Unknown site (problem 3)*
Proposed system			
Authentication proxy [3]	✓		
CAPTCHA [4]		✓	
Tsai [5]			✓

II. WAED

In this section, we propose the WAED, which adds additional authentication to allow human connection and uses the authentication result to improve the precision of the white list.

A. Cyber Attack

An overview of the process flow of a cyber attack using RAT malware is given in Fig. 1. First, the attacker sends an e-mail that contains RAT malware to a user. If the user executes the malware by mistake, the user is infected by it. The malware has a connection to the attacker and the attacker then controls the user's PC, which it uses to gather and steal data.

The proposed system focuses on the RAT malware's connection to the attacker ("Connect" in Fig. 1). When an HTTP connection occurs, the system evaluates the HTTP request and if the connecting server is not on the white list, it adds an additional authentication, e.g. CAPTCHA, which malware cannot pass. As a result, even if the user is infected by RAT malware, the proposed system blocks connection to the attacker and prevent the leakage of information.

B. Overview of the Proposed System

An overview of the proposed system is shown in Fig. 2. The system consists of three components: an additional authentication controller, an additional authenticator, and a white list updater.

Additional Authentication Controller, which makes decision whether to add authentication or not on the basis of white list or previous authentication history.

Additional Authenticator, which generates a challenge-response test for the user.

White list Updater, which updates the white list based on the result of additional authentication.

The details of each component are described in Section II-C.

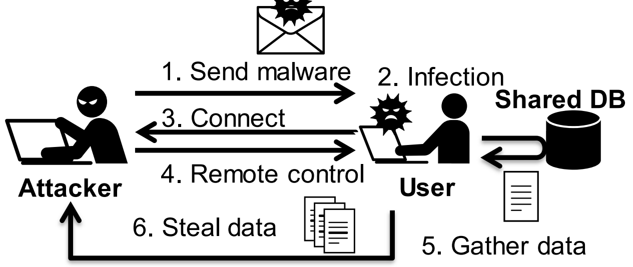


Fig. 1. Process flow of a cyber attack.

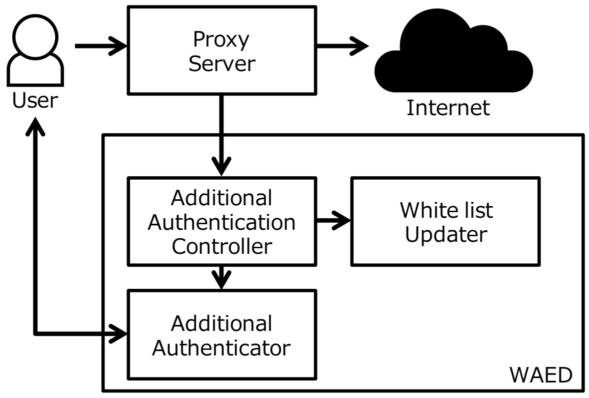


Fig. 2. Overview of the WAED.

When the system receives an HTTP request from a user, it checks if the connecting server is in the white list or the user's authentication history and determines whether to add authentication or not. In cases where additional authentication is required, the system generates additional authentication (CAPTCHA). If the user passes CAPTCHA, the system permits the HTTP request to the Internet as it has been verified that the HTTP request comes from a human. When several users have passed the additional authentication on a particular server, that server is added to the white list because it is deemed as being used for business. Thus, the proposed system takes countermeasures against malware such as RAT without any disruption of business activities.

C. Details of the Proposed System

The three components of the proposed system are described in detail below.

Additional Authentication Controller. Figure 3 shows the flow of the additional authentication. First, the additional authentication controller receives an HTTP request from a user and checks if the connecting domain is in the white list. If the connecting domain is registered in the white list, it permits the request. If it is not, it checks if the connecting domain is in the authentication history of the user. If the user passed the previous authentication of that domain, it permits the request. If the user have never passed, the additional authentication controller forwards the HTTP request to the additional authenticator.

Some web pages contain cascading style sheet (CSS) files that are hosted by another server. This server is often not in the white list, so the proposed system causes collapse of web layout (Fig. 4, left side). To solve this problem, the additional authentication controller permits the HTTP request if contains a HTTP Referer. The additional authentication controller stores an access log to manage the previous connection status. This access log contains the access timestamp, user id[1], IP address, URL, and HTTP Referer.

[1] In order to protect the user's privacy, we store hashed user id.

Additional Authenticator. The additional authenticator generates the CAPTCHA form and tests user input (Fig. 5). We used SimpleCaptch [8] to generate it.

White list Updater. The white list updater updates the white list based on the result of additional authentication. Servers where a certain number of users pass the additional authentication are added to the white list, as these servers are classified as being used for business.

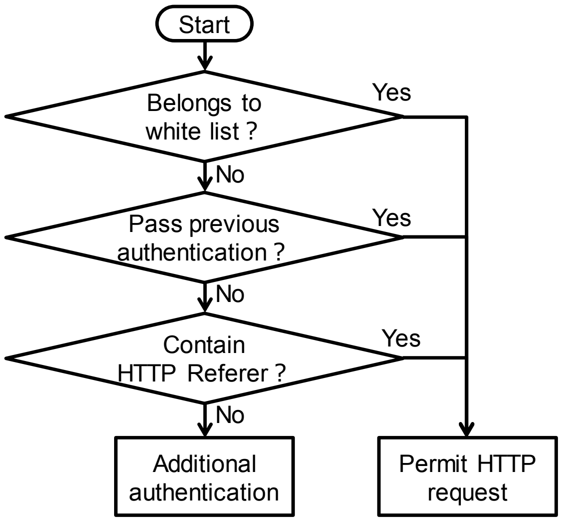
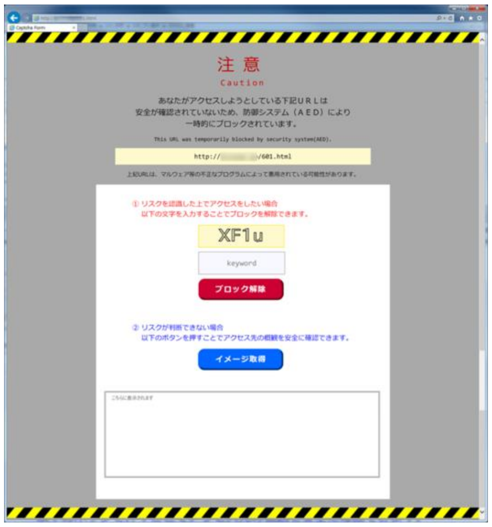


Fig. 3. Algorithm flow of additional authentication.

Fig. 4. Example of problem (Left: before, Right: after).



注 意

Caution

あなたがアクセスしようとしている下記ＵＲＬは
安全が確認されていないため、防御システム（ＡＥＤ）により
一時的にブロックされています。

This URL was temporarily blocked by security system(AED).

http://[redacted]/601.html

上記URLは、マルウェア等の不正なプログラムによって悪用されている可能性があります。

① リスクを認識した上でアクセスをしたい場合
以下の文字を入力することでブロックを解除できます。

XF1u

keyword

ブロック解除

② リスクが判断できない場合
以下のボタンを押すことでアクセス先の概観を安全に確認できます。

イメージ取得

こちらに表示されます

Fig. 5. Screen capture of CAPTCHA form.

D. *Implementation of the Proposed System*

An overview of the proposed system is shown in Fig. 6. We implemented the entire system on a single real machine using the software listed in Table II. The Internet Content Adaption Protocol (ICAP) [9] was used for additional authentication judgment. The white list is initialized to empty.

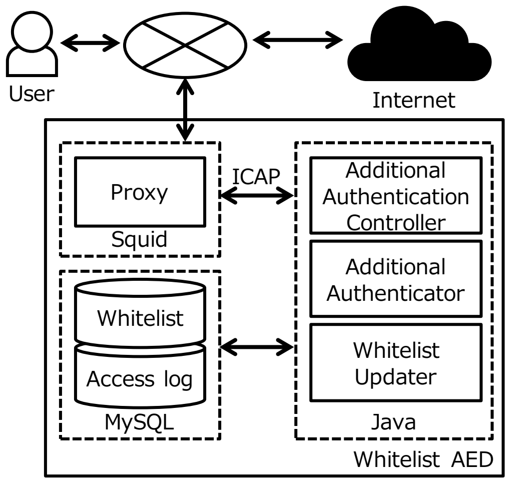


Fig. 6. Implementation of the proposed system.

TABLE II. SOFTWARE USED IN THE IMPLEMENTATION

Components	*Software*
Additional Authentication Controller, White list Updater	Java 1.8.0
Additional Authenticator	Java Servlet 3.0
Database (White list, Access log)	MySQL 14.14
Proxy	Squid 3.3.8

III. EVALUATION

We performed three experiments to evaluate the proposed system's performance in terms of security risk mitigation.

A. Focus of Evaluation

Evaluation 1: Blocking rate of RAT malware. We evaluated the accuracy of the prototype system.

Evaluation 2: Requesting rate of authentication. We evaluated the requesting rate of additional authentication in daily business activities using the prototype system.

Evaluation 3: Relation between user size and requesting rate. The greater the number of the WAED users, the higher the white list accuracy, so we evaluated the relation between user size (*n*) and requesting rate.

Evaluation 4: Threshold of updating white list. The lower the selected threshold the number of "passed" users to be included on the white list, the lighter the effect on business, so we evaluated the relation between threshold (*k*) and requesting rate.

Evaluation 5: Business user study. We evaluated the effect on business by administering a questionnaire to users of the prototype system.

B. Evaluation Methods

The three experiments are described in detail below.

Experiment 1: We used 50 RAT malware applications in the wild to evaluate blocking accuracy. The example of RAT malware used in the experiment 1 is shown Table III.

Experiment 2: We conducted a trial run of the WAED with 50 users for 17 days, February 8, 2016 to February 24, 2016. We added the domain to the white list when five users passed the additional authentication because 10% of users in an organization would have low security knowledge.

We evaluated the relation between the number of users, n = 10, 50, 100, 500, 1000, who used the prototype system and the requesting rate. In the experiment, we used 15 days' worth of data from the proxy access log to calculate the requesting rate. Specifically, we selected n user's proxy access log and determined whether to add additional authentication or not. If it was determined to add additional authentication, we regard that the user had passed additional authentication. If five users passed the additional authentication (k = 5), we insert the domain into the white list. We selected n users at random ten times and took the average requesting rate.

We evaluated the relation between the threshold, k = 1, 5, 10, 25, 50, and the requesting rate. In the experiment, 15 days' worth of data from the proxy access log was used to calculate the requesting rate. Specifically, we selected the 100 users proxy access log (n = 100) and determined whether to add additional authentication or not. If determined to add additional authentication, we regard that the user passed additional authentication. If k users passed the additional authentication, we inserted the domain into the white list. Thus, we could evaluate the requesting rate. We selected 100 users at random 10 times and calculated the average requesting rate.

Experiment 3: We administered an anonymous questionnaire to evaluate the effect on business.

C. Evaluation Results

Evaluation 1: Blocking rate of RAT malware. Table IV lists the blocking rate for the RAT malware. As shown, the proposed system blocked all RAT malware used in the experiment 1, while the authentication proxy blocked only 82%.

Evaluation 2: Requesting rate of authentication. Table V lists the total records of connections and additional authentication. Figure 7 shows transitions of the size of the white list. Excluding noise, we selected 24 users who constantly used the prototype system. As shown in the table, the requesting rate of authentication was 1.93%. In the experiment 2 period, there were zero security incidents, so 1.93% indicates false positive. The mechanism using of the HTTP Referer is efficient to reduce the requesting rate.

Although the number of additional authentications is 12,115, the number of responses is 1,620, daily average response: 3.97. Digital certificates distribution sites, such as CRL and OCSP, and software update sites were included as a class of no response sites. These sites were connected by browser or OS in the background, so the user could not view the CAPTCHA form. To solve this problem, we have to insert these sites into the white list in advance.

TABLE III. EXAMPLE OF RAT MALWARE USED IN EXPERIMENT 1

Type	*Hash (MD5)*
Emdivi [1]	e5653a4bca1239b095509438a3040244
PlugX [10]	5a22e5aee4da2fe363b77f1351265a00
ChChes [11]	8a93859e5f7079d6746832a3a22ff65c

TABLE IV. RESULTS OF BLOCKING RATE

	Blocked malware	*Blocking rate*
Proposed system	50	100%
Authentication proxy [3]	41	82%

TABLE V. RESULTS OF EXPERIMENT 2

	Records	*Ratio*
Additional authentication	12,115	1.93%
- with response	1,620	0.26%
- no response	10,495	1.67%
No additional authentication	616,718	98.07%
- connect to authenticated domain	122,467	19.48%
- connect with HTTP Referer	473,677	75.33%
- connect to white list	20,574	3.27%
Total	628,833	100.00%

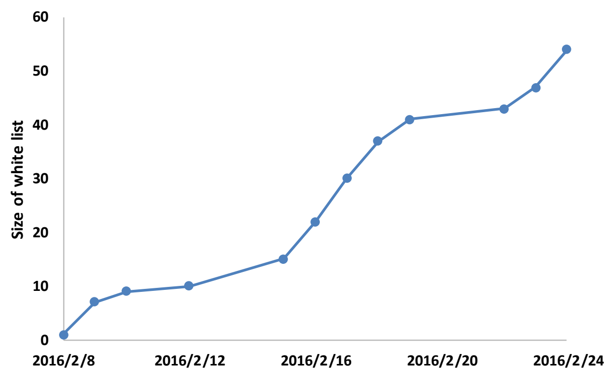


Fig. 7. Transitions of size of white list.

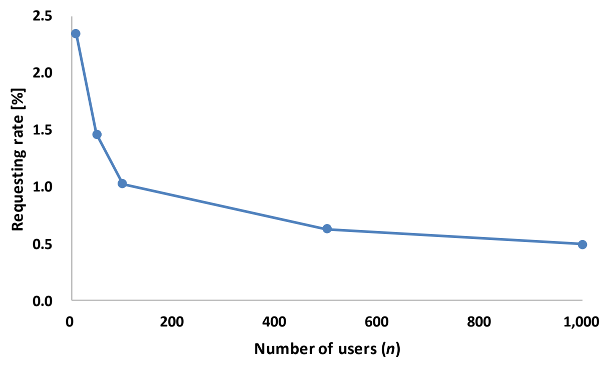


Fig. 8. Requesting rate with regard to number of users (*n*).

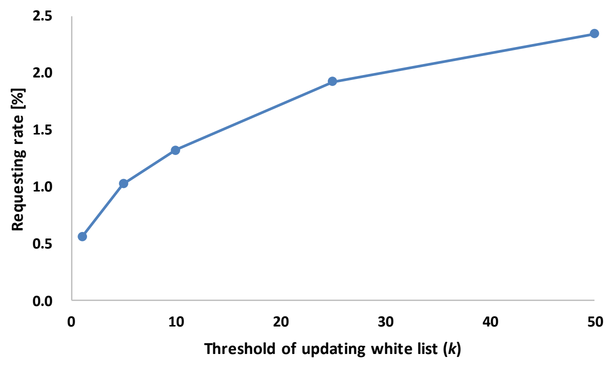


Fig. 9. Requesting rate with regard to threshold of updating white list (k).

Evaluation 3: Relation between user size and requesting rate. Figure 8 shows the relation between the number of users and the requesting rate. As shown in the figure, the requesting rate decreased rapidly when the user size was less than 200, and decreased more gradually after that.

Evaluation 4: Threshold of updating white list. Figure 9 shows the relation between the threshold and the requesting rate. As shown in the figure, the requesting rate gradually sped up when the threshold was increased. We found that when *k* was changed from 5 to 1, the requesting rate dropped by half.

TABLE VI. QUESTIONNAIRE RESULTS

Question	*Choices*	*Users*
How often did the additional authentication occur?	No additional authentication.	0
	Once per day.	3
	Several times per day.	13
	More than 10 times per day.	3
How did you feel about the frequency of additional authentication?	Inconvenient.	9
	No problem like this.	10
	No problem if increased.	0
Did you notice any change of the frequency of additional authentication?	Increased.	0
	No change.	9
	Decreased.	10

TABLE VII. RESULTS OF FISHER'S EXACT TEST

	No problem	*Inconvenient*	*p value*
Once per day.	3	0	0.3731
Several times per day.	6	7	
More than 10 times per day.	1	2	
Total	10	9	–

Evaluation 5: Business effects by questionnaire. The results of the questionnaire are summarized in Table VI. Responses were received from 19 participants in our department. We conducted Fisher's exact testing to determine if there are associations between the frequency of additional authentications and negative feeling. The results of this testing are listed in Table VII.

The number of users who felt there was no problem is greater than the number of users who felt inconvenience, and the Fisher's exact test results indicate there were no significant differences between the frequency of additional authentications and inconvenience of users. Although about half of users felt inconvenience, this number should decrease as they use the system more or if we set an initial white list.

IV. DISCUSSION

A. *Optimum User Size*

Here, we discuss the optimum user size for the proposed system. The relation counts between answering and requesting, shown in Fig. 10, show that answering counts are a linear to that of requesting. Figure 11 shows the relation between the number of users and the requesting counts. We already know efrom the questionnaire (Table VII) that no users felt any inconvenience when additional authentication occurred once per day. In Fig. 10 (approximation formula), we see that answers occurred once per day when the requesting counts were 5.8. In Fig. 11, the requesting counts of 5.8 corresponds to 1,000 users, which means that if 1,000 users use the proposed system, the average answering counts are once per day. This demonstrates that the proposed system can take countermeasures against malware without disruption of business activities.

B. *Throughput of the Proposed System*

We determined that the optimum user size for the WAED is 1,000. In this subsection, we discuss how this system is able to manage the load of 1,000 users. We performed a performance evaluation experiment using Apache Jmeter [12] with the evaluation environment specifications listed in Table VIII. The size of the white list is determined by simulation using the proxy logs of 1,000 users. Figure 12 shows the throughput and average response time of the proposed system. The proposed system's throughput was about 3,000 requests/sec, and the maximum requests of 1,000 users was 748 request/sec (average: 22.9 requests/sec), thus indicating that the system can stand a load of 1,000 users.

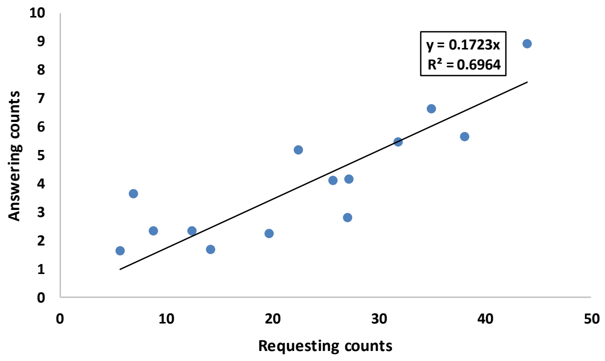


Fig. 10. Answering counts vs. requesting counts.

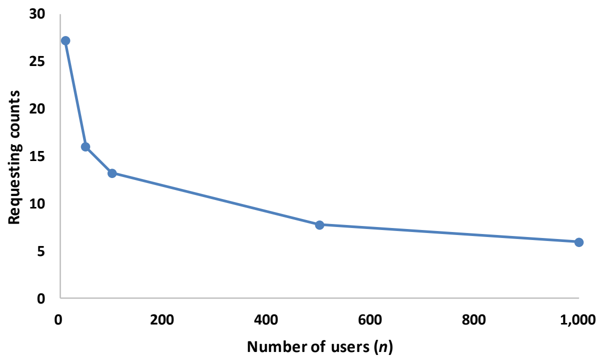


Fig. 11. Requesting counts with regard to number of users (n).

TABLE VIII. Specifications of evaluation environment

Item	*Value*
CPU	Intel Core i5 3.2GHz
Memory	4GB
Size of white list	21,000

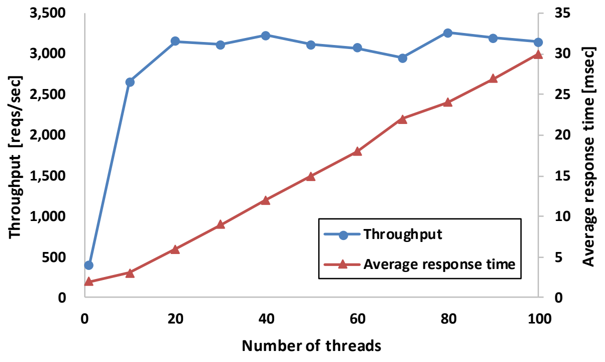


Fig. 12. Throughput and average response time of proposed system.

C. *Limitations*

For convenience, this system allows access with a HTTP Referer, so it cannot block Drive by Download attacks, which use a landing site with a HTTP Referer. Moreover, this system cannot block a compromised site if the site was previously included on the white list. To solve this problem, it is necessary to monitor security vendor sites and remove them if they become compromised.

V. Conclusion

We proposed an Autonomous Evolution of Defense System based on white list that blocks communications of RAT malware without a suspicious list. We implemented the proposed system and found that it succeeded in blocking malicious connections. The optimum number of users in the proposed system is 1,000. As future work, we will conduct a large-scale experiment for verification. We will also try to find HTTP Referer features used by landing sites to block Drive by Download attacks.

The system, products, and service names used in this paper are the trademark or registered trademarks of each organization.

References

[1] Symantec Corporation. (2014) Backdoor.Emdivi. [Online] https://www.symantec.com/security_response/writeup.jsp?docid=2014-101715-1341-99

[2] Japan Computer Emergency Response Team coordination Center. (2015) Emdivi and the Rise of Targeted Attacks in Japan. [Online] http://blog.jpcert.or.jp/2015/11/emdivi-and-the-rise-of-targeted-attacks-in-japan.html

[3] Information-techonogoly Promotion Agency. (2013) System Design Guide for Thwarting Targeted Email Attacks. [Online] https://www.ipa.go.jp/files/000035723.pdf

[4] Japan Computer Emergency Response Team coordination Center. (2015) PoisonIvy adapts to communicate through Authentication Proxies. [Online] http://blog.jpcert.or.jp/2015/07/poisonivy-adapts-to-communicate-through-authentication-proxies.html

[5] Dwen-Ren Tsai et al., "A Proxy-based Real-time Protection Mechanism for Social Networking Sites," Security Technology (ICCST), pp. 30-34, 2010.

[6] H. Nakakoji et al., “Proposal and Evaluation of Cyber Defense System Using Blacklist Refined Based on Authentication Results,” 2016 19th International Conference on Network-Based Information Systems (NBiS), Ostrava, 2016, pp. 135-139.

[7] D. Plohmann et al., “A Comprehensive Measurement Study of Domain Generating Malware,” 25th USENIX Security Symposium (USENIX Security 16), Austin, 2016, pp. 263-278.

[8] James Childers. (2008) SimpleCaptcha. [Online] http://simplecaptcha.sourceforge.net/

[9] The Internet Society. (2003) Internet Content Adaptation Protocol (ICAP). [Online] http://www.ietf.org/rfc/rfc3507.txt

[10] TELUS Security Labs. (2015) Backdoor.Win32.Plugx.L [Online] http://telussecuritylabs.com/threats/show/TSL20150720-08

[11] LAC Co., Ltd. (2017) The Relationship Between the Attack Group menuPass and Malwares "Poison Ivy, PlugX, ChChes". [Online] https://www.lac.co.jp/english/report/2017/08/30_alert_01.html

[12] Apache Software Foundation. (2018) Apache JMeter. [Online] https://jmeter.apache.org/