Journal of Engineering and Applied Sciences 13 (Special Issue 2): 2992-3000, 2018
ISSN: 1816-949X

Design and Implementation of the Safety Application Execution Control System Using the White-Listing

[1]Sang-Ann Nam, [1]Jong-Chul Park, [2]Jae-Pyo Park and [1]Chang-Hong Kim
[1]Department of IT Policy Management,
[2]Department of Information Security, Soongsil University, 07027 Seoul, Republic of Korea

Abstract: The leakage of important files through cyber-attacks begins from the malicious codes installed in the personal computers and there is an increasing need of developing technology to block the activities of malicious codes by detecting and identifying process through integrated analyses. In this study, we suggest the endpoint application program controlling method that enables a more secure protection of PC not only from the malicious codes but also from the exploit attacks based on the vulnerabilities of the OS or the application program by applying various integration of protection technologies such as the white-list-based application program execution control method to verify integrity, media control, registry protection, important file manipulation prevention and the process access inverse connection IP/Port control. Also, we suggest a method of measuring and learning the riskiness of the agent application program before registering the user-proven safe application programs on the white-list. The system with the white-list based endpoint application program control method consists of the following three parts. The agent blocking the execution of unauthorized programs that is not on the white-list or the unidentified programs when running an application program, the event server that delivers the execution or stop orders of the application program and the monitoring console for the policy operation management, white-list registration and the application program management. The realized system blocks the execution of the unauthorized programs that are not registered on the white-list and the unidentified executive files through the execution control agent, when running an application program. For the performance evaluation of the realized system, after installing the verification agent and measuring the memory usage, booting speed, the execution program loading speed at the endpoint and security, the performance comparison with the conventional commercial system was conducted to verify the excellence of the proposed system.

Key words: Application execution control, white-list, endpoint, media control, registry protection, unidentified, performance

INTRODUCTION

Various security management solutions such as the patch management solutions, PC firewalls and media control management solutions are introduced to prevent the increasing risk of cyber-attacks. In consequence, the importance of security system policy management and operation is increasing and the costs for introducing security solutions accompanied by the introduction of individual solutions are continuously rising (Warren and Hutchinson, 2000). The maintenance expenses and operational manpower are increasing for the management and operation of such security solutions. Although, a number of enterprises are using various anti-virus products, security accidents are repeatedly occurring (Reaves and Morris, 2012). Therefore, it is necessary to have an alternative technology to overcome such issues. The malicious code distribution attacks are secretly committed targeting the endpoints such as terminal PCs to bypass the information protection system such as Anti-Virus Software using the traditional network security system and signature. Due to such attacks, the public offices, financial companies, telecommunication companies, large shopping malls are experiencing serious damages including a massive amount of personal information leakage and relevant system and network paralysis (Castro *et al*., 2006). As the attacks of malicious codes and cyber-attacks focusing on the endpoints are increasing, it takes a significant period of time to address such problems (Ransbotham and Mitra, 2013).

In this study, we suggest the white-list-based endpoint application program execution control system that integrated the techniques of execution control, media control, registry protection, blocking manipulations of

Corresponding Author: Jae-Pyo Park, Department of Information Security, Soongsil University, 07027 Seoul, Republic of Korea

important files of the OS and inverse access IP/Port control for the access to the process which can verify the integrity of the application program to address the previously mentioned limitations of the existing information protection and security systems. The proposed system can protect the terminal PC more safely against the attacks of malicious codes as well as the exploit attacks that make use of vulnerabilities in the OS and application programs. The proposed technique protects the system using a comprehensive method of data encryption, application use control and network access control while performing white-list-based application program control, digital signature validity verification, registry protection and inverse access detection and control. The proposed technique focuses on establishing a more realistic security model such as blocking the functions of the invaded malicious codes to make the invasion meaningless, instead of making unrealistic strategies such as trying to block 100% of the malicious codes that are using various invasive methods due to the advancement in hacking techniques.

Literaure review

Security technique based on white-list: White-list refers to the list of verified or secured items it is the opposite concept of black-list which involves the ex-post prohibition of malicious items. The concepts of 'white-list' and 'black-list' are also called as 'positive' and 'negative' security methods, respectively (Lee *et al*., 2013). The black-list method that is currently implemented by the anti-virus solutions involves creating the database of signatures extracted from the malicious codes. In other words, the conventional technique blocks the programs or files that perform malicious activities without setting restrictions on program installation. Therefore, the black-list-based security solutions are convenient in terms of usability in the most universal environment but it has to involve ex-post management after the detection of malicious codes (Lee *et al*., 2010). In contrast, the white-list method only allows the execution of programs that are approved or permitted by the system manager and blocks any attempts to run unapproved or denied programs. Therefore, the advantage of white-list security method is that it is a stronger method that enables preemptive protection. Figure 1 shows the application program control process using the white-list method.

Anti-virus software is a typical solution of using black-list security measures. The anti-virus solutions create the signature database of the malicious codes that are identified dangerous and perform a comparison of these signatures with the files that are trying to enter the system or PC. When found identical, the Anti-Virus Software performs a treatment of the infected system or PC or blocks the execution of the file to protect the system or PC from the threats of malicious codes. Therefore, the black-list method safely manages the application programs and relevant files that may perform malicious activities, without setting preemptive restrictions on the installation of application programs. Thus, this method poses convenience in terms of usability under universal environment and conditions but once the malicious codes are detected, the action has to be made afterward and this method is vulnerable to new types or variant attacks (Jain and Gupta, 2016).

Also, if the system uses an internal network that cannot perform external or mutual communication such as in the case of PCs installed at manufacturing plants, it

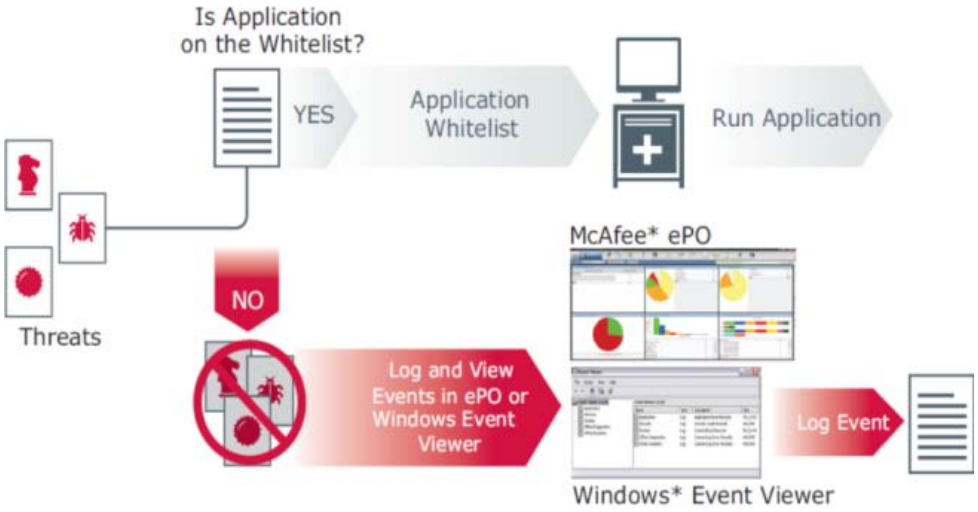

Fig. 1: The application program control process based on the white-list

takes significant efforts and time to download the security update engine. The black-list-based security products can deal with the new types of malicious codes only when such signature updates are made after the analysis of new types of malicious codes, so, the size of black-list-based vaccine engine constantly increases (Sheng *et al*., 2009). When there are frequent engine updates due to the new types of malicious codes, the engine size of the black-list-based vaccines continuously increases. Such accumulation causes the lack of storage drive capacity and influences the performance of the system, especially when the computer connected to the production facility is decrepit. Also, the Anti-Virus Software using the black-list-based security method regularly or irregularly performs signature updates in addition to virus inspection. Therefore, when the vaccine software runs on a system with lower specification, it uses up a large amount of system resource which leads to antivirus storm and negatively influences the performance due to the system overload (Prakash *et al*., 2010).

Malicious code attacks on endpoints: The IDG (International Data Group) an American IT research group, cited the 2013 report based on collected data of worldwide security appliance's by FireEye to analyze the attacking types and methods to bypass the traditional means of network information protection system in a short time using the evolved techniques to inject malicious codes as described (FireEye, 2014; Chen *et al*., 2014):

- Attacks on companies are occurring every 3 min on average
- Attacks are relatively more concentrated on tech companies with high intellectual property rights
- The number of attacks using the DLL (Dynamic Linking Library) files, a more advanced attacking method is increasing
- The number of malicious code attacks using the spear phishing method is increasing on social engineering

Due to the limitations of the signature detection technique, vaccine programs are implementing the cloud reputation management technique to supplement the shortcomings but it is still difficult to deal with the new types or variant malicious codes that have been evolved or not widely known. Accordingly, the need for signatureless-based security products are highlighted around the American security companies. The endpoint terminal computers are vulnerable to illegal hacking or malicious codes and the aspects of cyber-attacks are evolving as they are targeting the endpoint terminal computers to hack the server through the method of granting authority (Liu *et al*., 2013). It takes significantly more efforts and time to analyze the causes and address the problems related to the attacks on endpoint terminal computers, than the network invasion attacks. Therefore, there is an increasing need for management including program execution control and integrated control in terms of the endpoint terminal PCs.

MATERIALS AND METHODS

The design of white-list-based application program execution control system

System architecture: The proposed system is designed to make preemptive blocks to prevent unapproved application programs and relevant unidentified execution files that are not on the white-list by applying the functions of application program execution, media control, registry protection, protection against malicious manipulation of important files and inverse access IP/Port control through the execution control agent. Figure 2 shows the overall conceptual diagram of the proposed white-list-based application program execution control system.

The proposed system performs integrity verification based on the white-list in terms of all the execution files that are dependent on the application program and after the execution of the application program, it blocks the unknown malicious codes through the execution flow control. The proposed system can cut off all the malicious activities based on the integrity and execution flow of the application program without depending on the known signatures. Also, the subject of the log data analysis is the terminal PC, instead of the server connection log, so, it is designed to focus the control on the PC and the user, instead of the network and server which involves the large-scale statistical log analysis and big data analysis of the flow of executed files and application program. Figure 3 shows the overall structure of the proposed system.

The proposed system consists of three parts: the agent, event server and monitoring console. The agent performs preemptive blocks against the unapproved application programs and relevant unidentified execution files that are not on the white-list at the beginning stage of the application program execution. The event server transmits the execution order to run the normal application programs and the stop order to block the abnormally executed programs. The monitoring console provides the functions of policy management white-list registration and application program management.

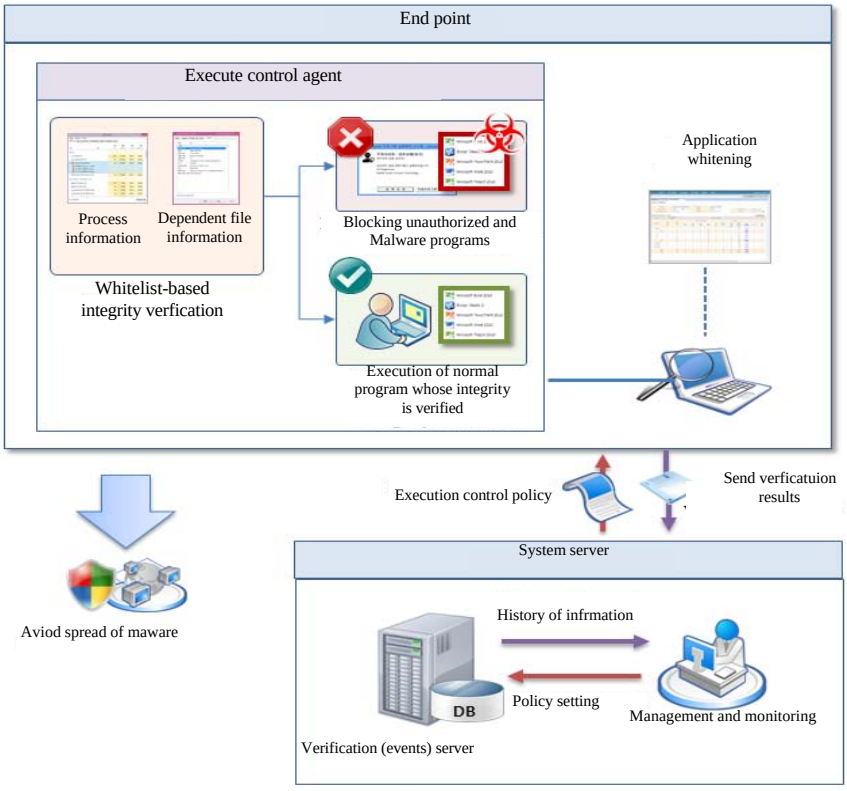

Fig. 2: Concept diagram for proposed system

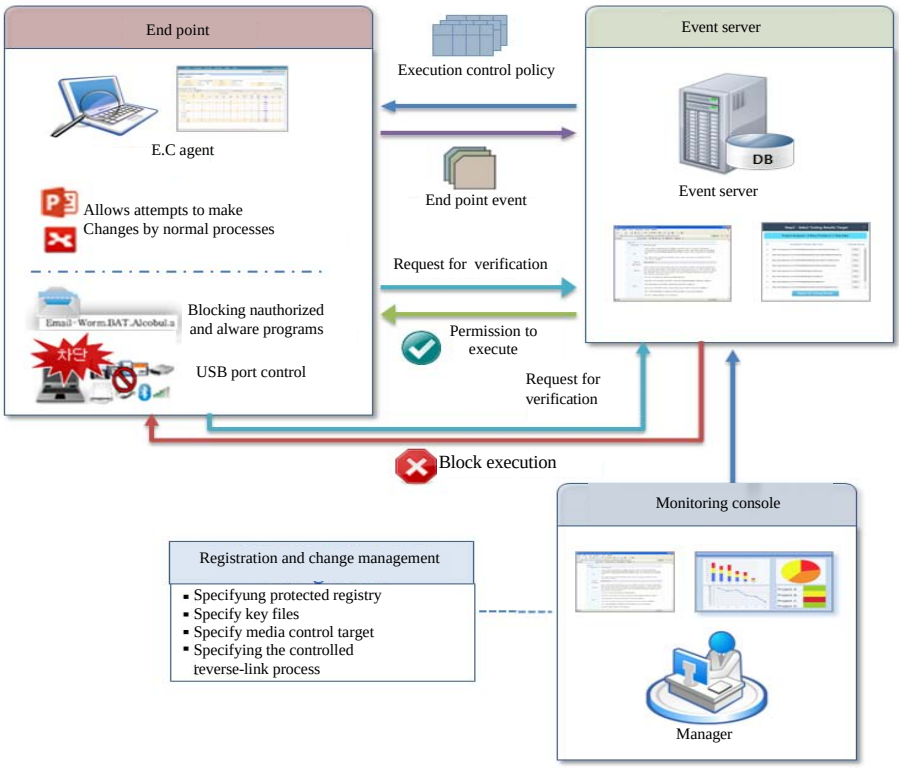

Fig. 3: Architecture of proposed system

The agent is installed in the terminal PC of the endpoint and it operates integrity verification and static information analysis based on the white-list, before the execution of the application program. When an event occurs in the operating system to run the application program, the agent normally runs the execution files that are registered on the white-list by the approved media in accordance with the execution control policy and preemptively blocks the execution of unapproved application programs, unidentified programs and relevant execution files that are not registered on the white-list. At this point, the endpoint agent transmits the history information of the execution of the application program to the event server and the event server performs a detailed analysis to manage the white-list and execution control policy.

The event server receives the history of the execution of normal attempts and the blocking of abnormal attempts of the application program from the endpoint agent and creates the DB to manage the records. Upon the registration requests in the white-list, the event server determines whether the application program is normally approved. In that way, it makes changes in and manages the white-list to deliver the execution control policy to the agent. In particular, the event server creates and stores the execution control policy events in accordance with the version checking results or the white-list registration requests from endpoint agents. Similar to the endpoint agent, the event server also verifies the execution history of the transmitted application program from the endpoint agent. According to the verification result, it transfers the execution or stop orders to the endpoint agent.

The monitoring console supports announcement tasks and manages the application programs registered in the white-list as well as the policy operation. It also performs the service condition analysis and system operation functions. It provides various statuses and trend information in the form of charts and statistical reports through the identification and tracking of the monitored information and performs creation and registration of policies. Also, it registers the endpoint agent update module and provides the information about the execution frequency of the application program as well as the reports about various log information. It also performs user history checking function. Through the monitoring console, the manager controls the previously designated media, verifies the integrity of the white-list and designates the registry and important files that are subject to protection.

The execution control process of endpoint application program: As shown in Fig. 4, the endpoint application program execution control process simultaneously monitors the application program execution events from the operating system at the white-list agent installed in the terminal PC and preferentially identifies if the execution process of the detected execution event occurs in the approved media by determining if the media is previously designated or not. Then, upon execution in the approved media, the integrity verification is performed in terms of the application program, through the white-list-based execution control of application program based on the execution file hash value and file properties and path information.

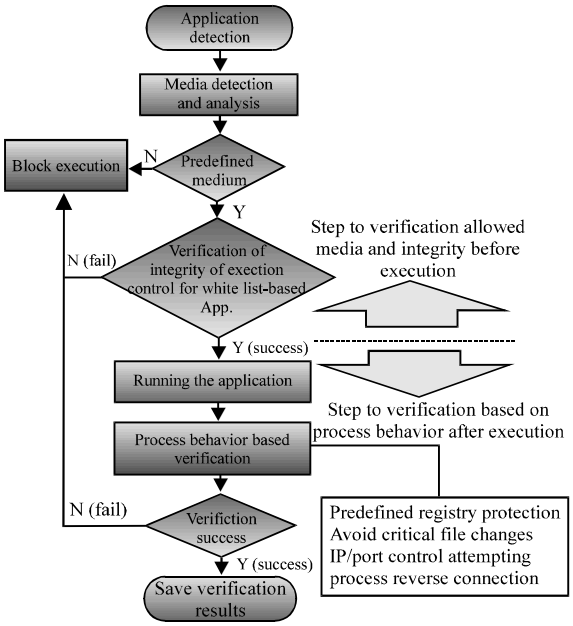

Fig. 4: Execution control process of endpoint application program

The program is executed when the integrity of the application program is verified and the execution is blocked when the verification fails. After the execution of the application program, the functions including the registry protection function predefined in the execution control policy, the protection function against malicious manipulation of important files, the process inverse access IP/Port control function based on execution control policy and the verification function based on process activities are performed. The execution of the application program is blocked if the process activity-based verification fails and the verification results are stored in the DB when the verification is successful. As shown in Fig. 4, the preceding steps before the application program execution are for the verification of approved media and integrity and the steps after the application program execution are for process activity-based verification processes.

RESULTS AND DISCUSSION

Implementation and tests

System implementation: The white-list application program execution control system consists of three modules: the verification agent module that is installed in the user's terminal device (the endpoint), the verification server module that is installed on the server and the management and monitoring server module. They are realized using C++, JAVA and JSP. As in Fig. 5, the information extraction function in the verification agent module is performed from the execution files that are subject to the white-list. The operation time and inspection pathway can be checked during the inspection, as well as the number of cases of registration and non-registration. The inspection is performed targeting the execution process and when the inspection is finished, the application program's file name, product name, product version, producer, file installation path, the number of files, the number of registrations and the number of non-registrations can be checked in the group that is subject to the registration.

Figure 6 shows the resulting screen after the integrity verification of execution files. Although, a specific

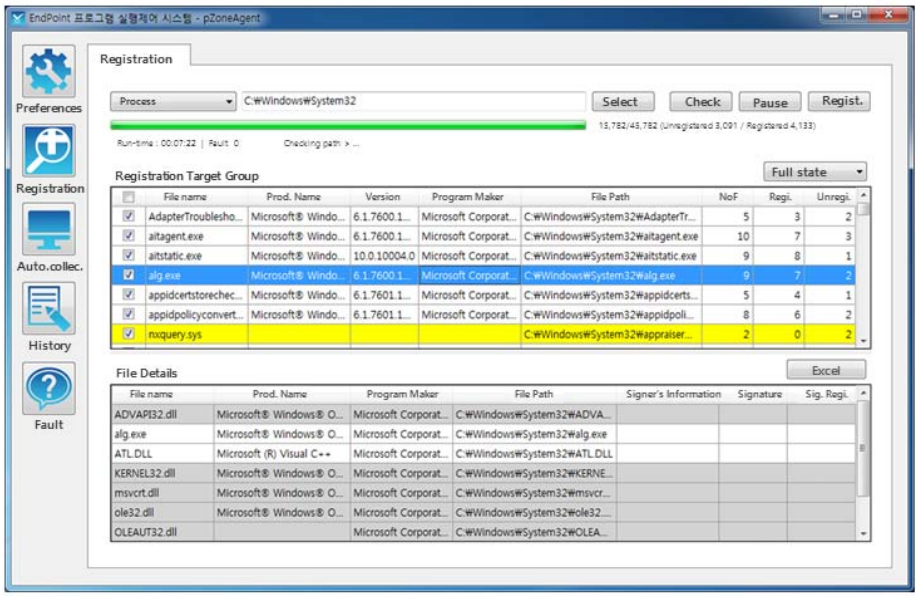

Fig. 5: Verification agent module-extraction and registration of execution file information

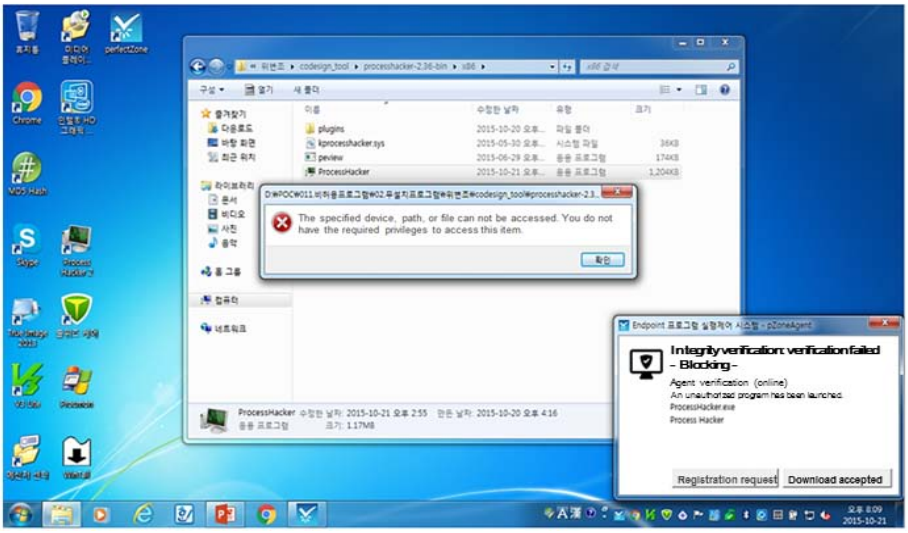

Fig. 6: Integrity verification for progress and dependent files

program is registered in the white-list, when the subordinate DLL is blocked due to the tampering or manipulation, the execution of the application program is blocked and a message of prohibition is displayed as in Fig. 7.

When the device subject of control is designated in the manager module, the execution of the network drive (Fig. 7a) and USB media (Fig. 7b) is controlled in the user environment as in Fig. 7.

Test and evaluations: As in Table 1, the performance of making preemptive prevention and prohibition against major security threats was tested to confirm the security of the realized system in terms of 7 methods of execution prohibition function and 6 methods of access control function. Table 1 shows the test results in terms of 200 times of experimental comparison between the conventional commercial system and the proposed system.

As shown in the test result, the realized system showed similar or better performance than the conventional system in terms of 9 criteria (rates of unapproved software blockage, unit-application program control, unit-file verification block, blockage of malicious codes pretending document files, blockage of downloads from unreliable websites, access control of registry manipulation, control of the execution of portable storages including USB drives, control of network access

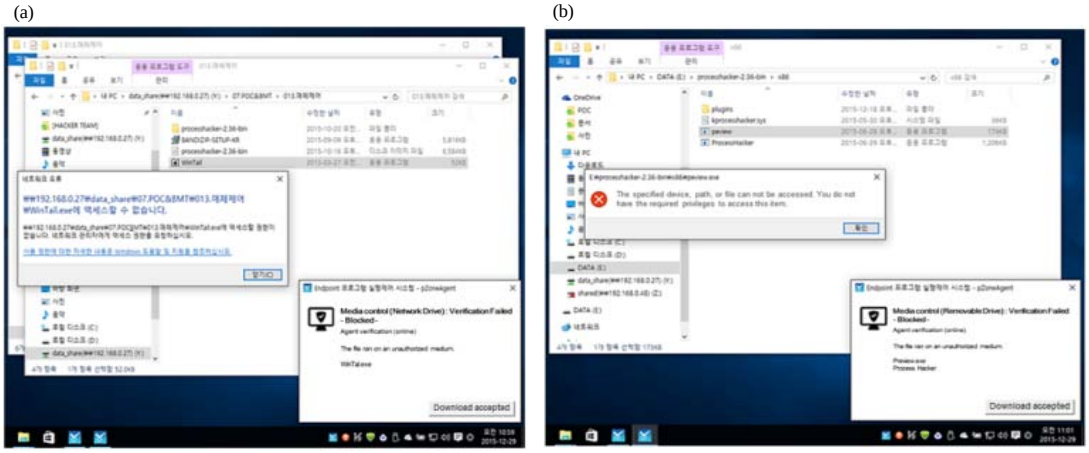

Fig. 7: a, b) The execution control for network drive and USB media

Table 1: Security test results and the comparison between the conventional system

Items	Details	Embedded security	Trust line	End point	BIT9	Proposed system
Execution prohibition (protection)	Rate of successful prohibition of unapproved software	1.000	1.000	1.000	1.000	1.000
	Rate of successful unit application program control (blockage, approval,verification of authentication, etc.)	0.895	-	0.905	0.920	0.945
	Rate of successful blockage of APT attack/zero-day backdoor execution malicious code	0.985	0.990	0.995	0.990	0.990
	Rate of successful verification blockage of unit file	0.675	0.985	-	0.975	0.995
	Rate of successful blockage of malicious codes (extension manipulation attack) that are disguised as document files	1.000	1.000	-	1.000	1.000
	Rate of successful blockage of downloads from unreliable websites	1.000	1.000	-	-	1.000
	Rate of successful detection and blockage of malicious activities including manipulation of important system information	1.000	0.955	0.945	0.970	0.995
Access control	Rate of successful access control of registry manipulation	0.895	-	-	0.955	0.995
	Rate of successful control of process access	0.995	0.985	1.000	0.995	0.995
	Rate of successful control of system file manipulation	0.995	0.995	-	1.000	0.995
	Rate of successful control of portable storages, including USB drive	-	1.000	1.000	1.000	1.000
	Rate of successful control of network access (outbound remote addr control)	-	0.980	1.000	-	1.000
	Rate of successful control of application program	1.000	-	-	1.000	1.000

and the control of application program) of a total of 13 criteria of security measures which proves the excellence of the performance of the realized system.

CONCLUSION

In this study, an integrative technology of covering the white-list-based application program execution control and media control for the verification of integrity, the protection against malicious tampering and manipulation, the prevention of manipulation of important files and the inverse access IP/Port control of the process was applied to overcome the limitations of the conventional black-list security measures to suggest a white-list-based endpoint application program execution control method that enables the protection of the endpoint terminal PC not only against the cyber-attacks due to the invasion of new types and variant malicious codes but also against the exploit attacks that make use of vulnerabilities in the OS or the application program.

The proposed system of implementing the white-list-based endpoint application program execution control method was designed to consist of three parts: the endpoint agent that blocks the execution of unapproved application programs that are not registered in the white-list or the relevant unidentified execution programs, the event server that transfers the execution or stop orders to the agent and the monitoring console that provides policy operation management, white-list registration and application program management.

After the installation of verification agent, the application program was registered on the white-list and measurements were made in terms of each item including the endpoint's memory occupancy, boot speed, application program loading speed and security to evaluate the performance of the realized system. Then, a comparative analysis was conducted to find the differences between the conventional systems and the proposed system and the excellence of the proposed system was investigated.

SUGGESTIONS

It is suggested that the proposed method as well as the system designed and realized in this study will contribute to the research and development of the white-list-based security solutions.

REFERENCES

Castro, M., M. Costa and T. Harris, 2006. Securing software by enforcing data-flow integrity. Proceedings of the 7th Symposium on Operating Systems Design and Implementation, November 06-08, 2006, USENIX Association, Seattle, Washington, ISBN:1-931971-47-1, pp: 147-160.

Chen, P., L. Desmet and C. Huygens, 2014. A Study on Advanced Persistent Threats. In: Communications and Multimedia Security, Decker, D.B. and A. Zuquete (Eds.). Springer, Berlin, Germany, ISBN:978-3-662-44884-7, pp: 63-72.

FireEye, 2014. Fireeye advanced threat report 2013. FireEye, Milpitas, California, USA.

Jain, A.K. and B.B. Gupta, 2016. A novel approach to protect against phishing attacks at client side using auto-updated white-list. EURASIP. J. Inf. Secur., 2016: 1-11.

Lee, J., Y. Lee and S.C. Kim, 2013. A White-List based Security Architecture (WLSA) for the Safe Mobile Office in the BYOD Era. In: Grid and Pervasive Computing, Park, J.J.H., H.R. Arabnia, C. Kim, W. Shi and J.M. Gil (Eds.). Springer, Berlin, Germany, ISBN:978-3-642-38026-6, pp: 860-865.

Lee, K., R.S. Tolentino, G.C. Park and Y.T. Kim, 2010. A Study on Architecture of Malicious Code Blocking Scheme with White List in Smartphone Environment. In: Communication and Networking, Kim, T., A.C.C. Chang, M. Li, C. Rong and C.Z. Patrikakis *et al.* (Eds.). Springer, Berlin, Germany, ISBN:978-3-642-17586-2, pp: 155-163.

Liu, S.T., Y.M. Chen and S.J. Lin, 2013. A Novel Search Engine to Uncover Potential Victims for Apt Investigations. In: Network and Parallel Computing, Hsu, C., X. Li, X. Shi and R. Zheng (Eds.). Springer, Berlin, Germany, ISBN:978-3-642-40819-9, pp: 405-416.

Prakash, P., M. Kumar, R.R. Kompella and M. Gupta, 2010. PhishNet: Predictive blacklisting to detect phishing attacks. Proceedings of the 29th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, March 14-19, 2010, San Diego, CA., USA., pp: 1-5.

Ransbotham, S. and S. Mitra, 2013. The Impact of Immediate Disclosure on Attack Diffusion and Volume. In: Economics of Information Security and Privacy, Schneier, B. (Ed.). Springer, New York, USA., ISBN:978-1-4614-1980-8, pp: 1-12.

Reaves, B. and T. Morris, 2012. An open virtual testbed for industrial control system security research. Intl. J. Inf. Secur., 11: 215-229.

Sheng, S., B. Wardman, G. Warner, L. Cranor, J. Hong and C. Zhang, 2009. An empirical analysis of phishing blacklists. Proceedings of the 6th Conference on Email and Anti-Spam, July 16-17, 2009, Mountain View, California, USA.

Warren, M. and W. Hutchinson, 2000. Cyber attacks against supply chain management systems: A short note. Intl. J. Phys. Distribution Logistics Manage., 30: 710-716.