Attacker Investigation System Triggered by Information Leakage

Yuta Ikegami, Toshihiro Yamauchi
Graduate School of Natural Science and Technology
Okayama University
Okayama, Japan 700-8530
Email: yamauchi@cs.okayama-u.ac.jp

***Abstract*—While a considerable amount of research has been devoted to preventing leakage of classified information, little attention has been paid to identifying attackers who steal information. If attackers can be identified, more precise countermeasures can be taken. In this paper, we propose an attacker investigation system that focuses on information leakage. The system traces classified information in a computer and substitutes it with dummy data, which is then sent to the outside. Moreover, a program embedded in the dummy data transmits information back from the attacker's computer to a pre-specified system for investigation. Information about the attacker can be obtained by an attacker executing the program.**

***Keywords*—Attacker investigation, Information leakage, Operating system.**

I. Introduction

Cyber attacks aiming to steal classified information have emerged as a major challenge to online security in recent years [1]. It is difficult to completely prevent attacks on a computer because methods of cyber attacks are becoming increasingly sophisticated. Thus, research has been conducted on preventing the leakage of classified information [2]–[5]. However, few studies have investigated the possibility of identifying the attacker attempting to steal information. Identifying the attacker allows for offensive control measures against a new attack and punitive legal action against the attacker. CrowdStrike [6] and Junos WebApp Secure [7] have been proposed as services that can help identify attackers. However, both require malware analysis or a large database a huge quantity of attackers. One proposed method of attacker identification involves obtaining an attacker's information by transmitting spyware to his/her computer [8]. However, spyware is not send into an attacker because forbidden low.

In order to handle the above problems, we propose in this paper a system that can identify an attacker attempting to access classified information. We also implement and test this system on a Linux operating system (OS). In our system, the file containing classified information is registered as classified information into the computer is traced. When the file is transmitted to the outside of a computer, it is substituted with dummy data, thereby preventing classified information from leaking. Moreover, the file contains an embedded program (exploration program) that collects and sends information regarding the computer to which the attacker has transferred the file. Using this information, our system can obtain the attacker's information. Because the proposed system can replace classified information with what kind of dummy data, it can obtain information of attacker in response to purpose.

The contributions of this paper are as follows:

- We propose a system that can procure the information of attackers attempting to steal valuable classified information.
- Our system prevents leaking of confidential information.

II. Related Work

A. Identifying and investigating attackers

Among other security-related services, CrowdStrike [6] can specify an attacker service which analysis of malware, and explorer damaging computer. In analyzing malware, we can investigate the programming language used, the server that communicates with the malware, the malware ' s communication information, etc.

Junos WebApp Secure [7] is a product that can detect an attack on a website or a web application. It transmits a token to the attacker when insertion of unjust code to input form is detected. This token is permanently saved in the attacker's computer, and subsequent attempted attacks from the computer can be detected by checking the token. Moreover, it obtains approximately 200 items of information regarding the attacker's computer, such as the version of his/her browser, add-ons, the attacker's Internet Protocol (IP) address, time zone, etc.

By Honeyfiles [9], if dummy data (passwords.txt etc.) which pulls an attacker's interest is installed on a file server, it will be detected as being attacked.

B. Problems with existing methods

One or more of the following problems exist in the research described so far:

1) Leaks of classified information cannot be prevented.
2) The malware used for attacks is needed for analysis.
3) Identifying attackers takes time.
4) Attacker identification is impossible.

Because Cloudstrike [6] specifies an attacker following an attack, it permits the theft of classified information. Moreover, it is difficult to identify attackers when the malware used

cannot be obtained, and the identification is time consuming. Similarly, Junos WebApp Secure [7] cannot specify an attacker when attacked from a third-party computer.

Honyefiles [9] only detects dummy data, and cannot prevent leakage of classified information. Moreover, it remains only in offensive detection and has not up to specified attacker.

III. System Design

A. System Requirement

In this section, we describe our proposed system to solve the aforementioned problems. To address problem 1 mentioned above, classified information in a computer is supervised. It is effective not to leak and to detect transmission of the classified information to the outside of a computer to supervise. To solve problems 2, 3, and 4, we devise an exploration program for the attacker's computer. However, the exploration program cannot legally be transmitted pre-emptively to the attacker's computer. For this reason, it is necessary to execute the attacker itself by making an exploration program.

The requirements of our proposed system are as follows:

1) It keeps track of and prevents leakage of classified information in a computer.
2) The attacker itself runs an exploration program on an attacker's computer.

B. Design Goals and Assumptions

The purpose of a proposed system is to prevent leakage of the classified information and execute an exploration program for the attacker's computer to obtain the attacker's information. The process, which is a run state of a program, accesses classified information, and leakage of information takes place by transmitting classified information to the outside of a computer. The flow that a process delivers information has generation of a file operation, inter-process communication, and a child process. In these processes, since the OS is always involved, supervising the system calls of these processes can prevent leakage of valuable information. In order to execute an exploration program on an attacker's computer, it is necessary to install the program on his/her computer.

In order to increase the likelihood of such installation, our system replaces classified information with the exploration program prior to transmitting it. The exploration program can thus be send to an unsuspecting attacker. The OS can replace the classified information with the exploration program by supervising the system calls of the process.

In order to obtain the attacker's information, it is necessary for him/her to execute the exploration program. However, if an exploration program is installed on the attacker's machine as is, it will be easily detected. Hence, in order make it difficult for the attacker to detect the exploration program, it is effective to embed it into the dummy data that substitutes the actual classified information when the attacker tries to access it. Thus, the attacker misunderstands his/her theft of the dummy data as having yielded classified information, opens the file containing the dummy data, and hence enables the exploration program to run on his/her machine.

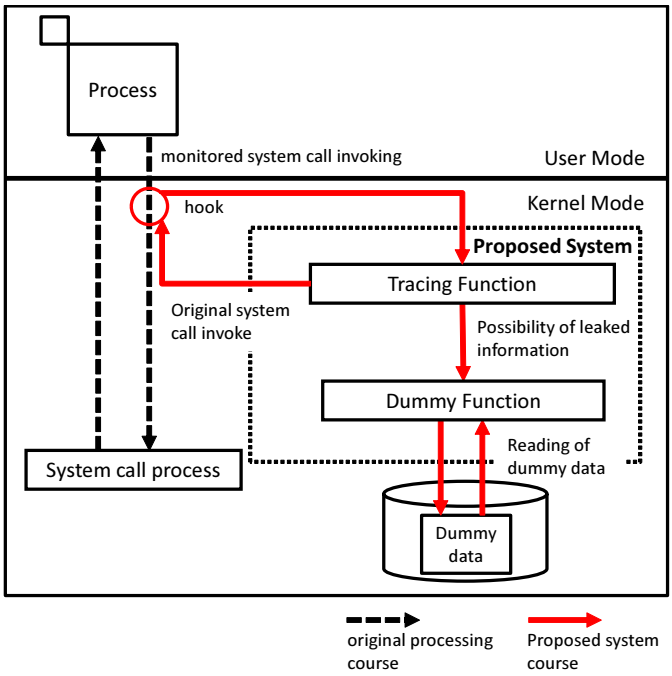

Fig. 1. Overview of proposed system

Our proposed system is designed to prevent remote attacks on systems through malware as well as illegal access to sensitive information.

C. System Architecture

An overview of the proposed system is shown in Figure 1. It implements the following functions and mechanisms:

- A function to trace classified information diffusion (Tracing Function)
- A dummy data exchange function (Dummy Function)

The Tracing Function uses a system proposed elsewhere [2]. Since the Tracing Function may tell the contents to other processes, files, etc., when the system call which the process published is hooked and a process reads classified information, the relevant processes and files are registered as classified information, and the system traces them.

The Dummy Function replaces classified information with dummy data, and is called from the Tracing Function prior to sending classified information from the computer.

D. Tracing Function

The Tracing Function tracks the diffusion of classified information. When the relevant data is accessed and transferred from the computer, the Tracing Function detects this. Classified information diffusion is carried out by reading classified information and sharing its contents with other processes, files, etc.

As a file operation which classified information diffuses, there is reading about the contents of a file and writing about contents of a file. In the case of a file containing classified

information, reading origin needs to make a process applicable to surveillance. Moreover, since the classified information may have been written out when the surveillance process wrote the information to a file, it is necessary to consider the file of the beginning point as the file for surveillance.

The surveillance process can be transmitted to other processes through inter-process communication. The Trace Function supervises inter-process communication using a socket, shared memory, a pipe, and a message queue. When the sending process is a surveillance process, it is necessary to make the surveillance target for communication and the receiving process of mediating transmission.

When there is a function where a child process inherits the resources of a parent process at the time of its generation (e.g., fork processing in UNIX), information is shared with processes by using this function. Therefore, when a parent process is a surveillance process, it is necessary to turn the child process into a surveillance process as well. These processes are supervised, and the diffusion of classified information is thus traced.

E. Dummy Function

The Dummy Function is called by the Trace Function when classified information is being sent from the computer, and replaces this information with dummy data, thereby preventing a leak. The procedure of the Dummy Function is as follows:

1) The size of the file containing the classified information is acquired.
2) The file size of the dummy data is acquired.
3) The file size in 1) is compared with that in 2).
4) If the size of the file containing classified information is larger, fit the file size of the dummy data to the file size of the classified information.
5) When the size of the file containing classified information is smaller, fit the file size of the dummy data to it.

When the sizes of the classified information and the dummy data differ, attackers can detect the information switch. For this reason, the Dummy Function fits the size of the dummy data to that of the classified information. When size of the file containing classified information is larger than that containing the dummy data, the contents of the latter are written into the buffer containing the classified information, which pads the dummy data to the size of the classified information. The buffer can be acquired by the supervise argument of the system call to supervise.

Furthermore, when the size of the classified information is smaller than that of the dummy data, the Dummy Function fits the latter to the former. For this reason, the file size of dummy data has a problem that it cannot fit in the file size of classified information.

F. Exploration Program

The dummy data is embedded into an exploration program that obtains the attacker's information. Since any program can be embedded, it can change into dummy data according to the information desired by the user.

IV. Implementation and Evaluation

A. Contents of implementation

We implemented the Dummy Function prototype on Linux 3.4.9. The Dummy Function was realized as a Loadable Kernel Module (LKM) and can cooperate with the Tracing Function without the need to alter the OS. However, Dummy Function is not realized cooperation with Tracing Function. We intend to tackle this issue in future research. Hence, we implemented file operation tracing function into Dummy Function. Dummy Function is hooked system call, which read(), write(), and sendto(). This evaluation used above mentioned Dummy Function.

B. Evaluation purpose

Within the computer imported proposed system, the program that send classified information to the external of computer for upload, and the following two points are checked.

1) Prevention of leakage classified information
2) Acquisition of the information on attacker's computer

Our proposed system prevents information theft by replacing classified information with dummy data. Moreover, it checks to see if the exploration program embedded into the dummy data is functioning, and can obtain an attacker's information through his/her computer. We tested our system on an Intel Core i7-3770 processor with 3.4 GHz and 4 GB of random-access memory (RAM) running Debian with Kernel 3.4.9. Since we assumed that the attacker was using a Windows OS, we designed the exploration program to execute on Windows. However, the exploration program is not recognized as an executable file, and the file name of dummy data has been a file name of classified information, because it is an executable file that operates on Windows (.exe file). For this reason, the extension of the file containing the dummy data was changed to .exe format on the attacker's computer. Handling the problem of file extensions is subject for our future research.

An overview of the evaluation is shown in Figure 2, and the experimental procedure was as follows:

1) On the computer equipped with the proposed system, execute the program that sends classified information to the computer to upload.
2) The computer is accessed from an attacker's computer and classified information is transferred to the latter.
3) An attacker opens classified information.

In this experiment, the server receiving the attacker's information from the exploration program was implemented in our proposed system. The size of the classified information was 102,400 bytes and that of the exploration program was 53,003 bytes.

C. Results

The size and name of the file acquired from the attacker's computer were the same as the classified information in Section IV-B. When the extension of the file obtained was changed to .exe and run, the attacker's information was received by the

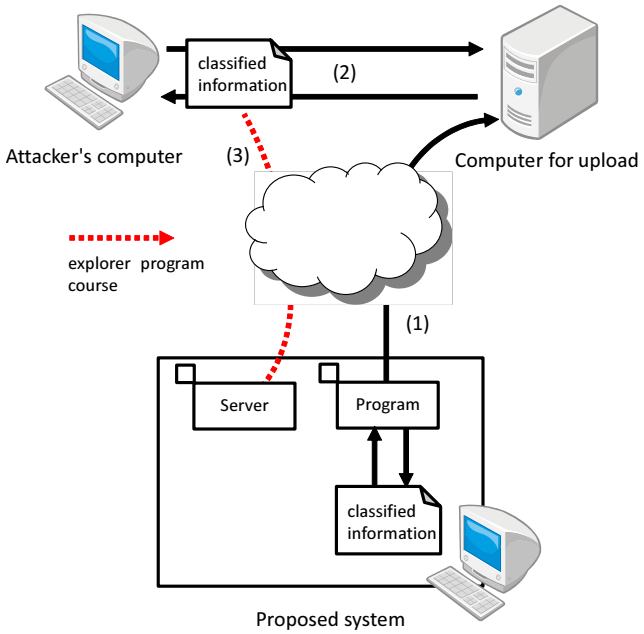

Fig. 2. Evaluation Overview

server. Thus, our system successfully prevented the leakage of classified information by replacing it with dummy data, and obtained the attacker's information.

V. Discussion

Proposed system left-behind subject is shown below. (1) Efficient execution method of exploration program. Our proposed system cannot currently execute the exploration program absolutely. Therefore, it is necessary to propose efficient execute method of exploration program. Similarity, exploration program is not executed absolutely when attacker use a character user interface (CUI). (2) Execution platform of the exploration program. Our system cannot currently execute the exploration program on multiple platforms. Then, proposed system can correspond to multiplatform by proposed system cooperate with p0f [10]. p0f is a tool that analyzes the Transmission Control Protocol (TCP) headers to specify the OS.

VI. Conclusion

We proposed an attacker investigation system focusing on classified information leakage. It is hooking the system call related to operation of classified information in a computer. Our proposed system tracks classified information as it is sent from the computer and replaces this information with dummy data. Thus, it can prevent classified information leakage. The dummy data contains an embedded exploration program that collects and transmits the attacker's information from his/her computer.

In experiments, our system successfully prevented leakage of classified information and obtained the attacker's information. The subject described with Section V is solved from now on, and the overhead of the evaluation or evaluation using actually operated malware.

References

[1] (2013, Apr.) 2013 DATA BREACH INVESTIGATION SREPORT. [Online]. Available: http://www.verizonenterprise.com/resources/reports/rp_data-breach-investigations-report-2013_en_xg.pdf

[2] Tabata, T., Hakomori, S., Ohashi, K., Uemura, S., Yokoyama, K., Taniguchi, H, " Tracing Classified Information Diffusion for Protecting Information Leakage, "IPSJ Journal (in Japanese), vol.50, no.9, pp.2088–2012, 2009.

[3] N. Otsubo, S. Uemura, T. Yamauchi, H. Taniguchi, " Design and Evaluation of a Diffusion Tracing Function for Classified Information Among Multiple Computers, " in Proc. 7th FTRA International Conference on Multimedia and Ubiquitous Engineering (MUE 2013), Lecture Notes in Electrical Engineering (LNEE), vol.240, pp.235–242, 2013.

[4] D. Y. Zhu, J. Jung, D. Song, T. Kohno, D. Wetherall, " TaintEraser: Protecting Sensitive Data Leaks Using Application-Level Taint Tracking, " ACM SIGOPS Operating Systems Review, vol.45, no.1, pp.142–154, 2011.

[5] S. Liu, R. Kuhn, " Data Loss Prevention, " IT Professional, vol.12, no.2, pp.10–13, 2010.

[6] Crowdstrike. [Online]. Available: http://www.crowdstrike.com/index.html

[7] (2013, Nov.) Junos WebApp Secure. [Online]. Available: http://www.juniper.net/us/en/local/pdf/datasheets/1000401-en.pdf

[8] (2012, Oct.) CYBER ESPIONAGE Against Georgian Government (Georbot Botnet). [Online]. Available: http://dea.gov.ge/uploads/CERT%20DOCS/Cyber%20Espionage.pdf

[9] J. Yuill, J. Zappe, D. Denning, F. Feer, " Honeyfiles: deceptive files for intrusion detection, " in Proc. 15th Annual IEEE Information Assurance Workshop, pp.116–122, 2004.

[10] p0f. [Online]. Available: http://lcamtuf.coredump.cx/p0f3/