NIGELOG: Protecting Logging Information by Hiding Multiple Backups in Directories

Tetsuji Takada Hideki Koike

Graduate School of Information Systems
University of Electro-Communications
Chofu, Tokyo 182-8585 JAPAN
{*zetaka,koike*}*@vogue.is.uec.ac.jp*

Abstract

This paper proposed a novel method to protect logging information more securely. The method, named NIGELOG, produces multiple backups of the logging information, hides them in arbitrary directories, and periodically moves these backups to other directories. Also, NIGELOG can automatically restore the original logging information by using the backups even if the original file is altered or deleted by intruders. NIGELOG is implemented as a C++ class library with some standard software libraries and does not require additional hardwares such as write-once media. It is easier to apply NIGELOG to UNIX applications which produces log-files.

1 Introduction

Intrusion detection consists of the collection of logging information and its analysis. In other words, it is impossible to detect any intrusion without the logging information(hereafter cited as *log-info*). In order to find the intrusion, system administrators inspect the log-info manually and intrusion detection systems analyzes the log-info automatically. On the other hand, intruders try to alter or delete the log-info to hide the evidence of their intrusion. Therefore, the log-info must be protected from malicious alteration even if the intruder successes to invade the system.

Some methods protecting the log-info have been proposed to date. It is, however, difficult to restore valuable information which is lost by the alteration even if these methods are used. Unlike the system configuration files, the log-info are frequently updated and therefore it is difficult to protect them by periodical backup.

This paper proposes a novel method to protect the logging information. The method, named "NIGELOG[1]," makes multiple backups of the log-info, hides them in arbitrary directories, and periodically moves the backups to other directories. Moreover, NIGELOG automatically restores the original log-info by using the backups even if the original is altered or deleted.

The remainder of this paper is organized as follows. In section 2, existing log-info protection methods are summarized. The concept and implementation of NIGELOG are described in section 3 and 4, respectively. Section 5 discusses advantages and limitations of NIGELOG. Section 6 concludes the paper.

2 Existing Log-info protection methods

Various log-info protection methods have been proposed to date. In this section, we categorize these methods in four and discuss their problems.

Write-once media The first method is to use write-once media. Since the information recorded in the media is impossible to alter, the log-info cannot be altered. It, however, requires to put additional hardware and system administrators have to exchange the media manually.

Trusted host The second method is to transfer the log-info to another reliable host computer. It, however, requires another computer to keep the log-info. Also, we must

[1] "NIGE" in Japanese means "run away."

Table 1. Various logging information protection methods and its functions

methods \ features	Impossible to Alter.	Difficult to Alter.	Detect Alteration.	Applicable to application.	Possible to Restore.
Write-Once media	●				
Trusted host	●				
Access Control	●				
Encryption		●	●	●	

ensure that the log-info is transfered correctly without any loss.

Access control The third method is to use access control which prohibits normal users to access the log-info[5]. However, since it requires special functions provided by operating systems (OS), it cannot be used in general.

Encryption The last method is to use encryption[1][3]. It makes it harder for the intruder to alter the log-info. And it enables to detect alteration. However, this method is useless when the intruder deletes the entire log-files. The lost log-info cannot be restored.

From these observations, there are two major problems in existing methods. One is the applicability of the methods. The methods 1, 2, and 3 require additional hardwares or special functions provided by operating systems.

The other is the difficulty of restoring the lost log-info. To detect intrusions, the log-info must be correct. And when the intrusion is detected, system administrators have to investigate when the system is invaded, how to invade the system and what the intruder did using the log-info. These investigation are necessary to recover the system correctly or to study about how to cope with such attacks. For this reason, the log-info must not be deleted.

3 NIGELOG: Basic Idea

To address the problems discussed in the previous section, we propose a new log-info protection method, named NIGELOG. In the followings, we describe some features of NIGELOG.

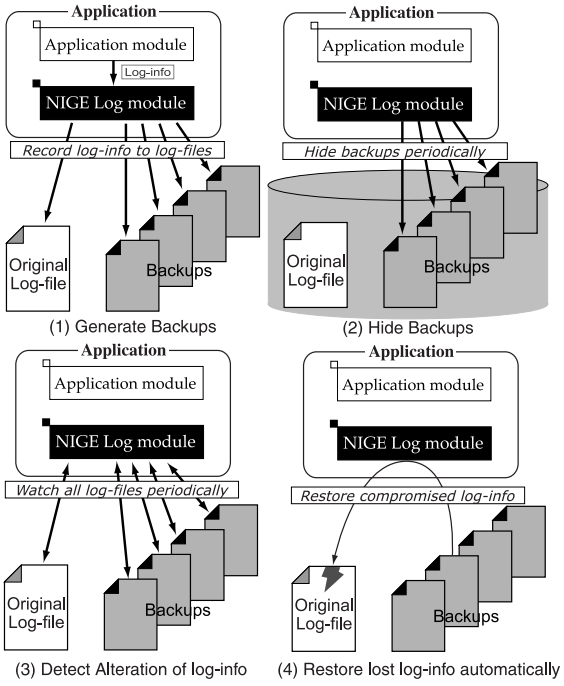

Figure 1. Features of NIGELOG

3.1 Multiple Backups

To keep the log-info more securely, NIGELOG makes multiple backups of the original log-file. However, a periodical backup is not appropriate for our purpose because there is a time delay until the backups have the same contents with the original. NIGELOG was designed to always keep the same contents of the original.

3.2 Hiding Backups

Although NIGELOG makes multiple backups, if the intruder already knows names of all backups and directories where the backups are saved, he/she can alter the log-info easily. Thus, NIGELOG hides the backups into the file systems in order to keep them more securely. The ways of hiding backups are listed as follows.

- Arbitrary number of backups are created.
- Arbitrary names are given to the backups.
- The backups are hidden into randomly selected directories.
- The backups are periodically moved to other directories with different names.

The intruder can detect all backups theoretically. For example, the intruder might search files which have the same contents to the original. However, even if the intruder succeeded to find all backups, NIGELOG had already moved some of the backups to other directories with different names. It is, therefore, extremely difficult to alter or delete all backups.

3.3 Alteration detection and automatic restoration

NIGELOG keeps watching not only the original log-files but also the backups in a short period. This makes it possible to detect alteration of the log-files quickly.

When alteration of the original or the backups is detected, the log-info is automatically restored from other correct backups. Because of this, it is possible to always provide correct log-info for intrusion detection systems or system administrators.

By using this automatic detection mechanism, it is also possible to develop more secure protection system, such as a system which automatically notifies to the system administrator by E-mail or a system which automatically records the log-info to write-once media.

4 Implementation

NIGELOG is implemented as a C++ class library and is composed of three modules and each module runs concurrently using Pthread library[6].

1. Receiving the logging information
   This module (marked "**A**" in figure 2) receives the log-info from the application and stores it in internal cache. It also removes the log-info which was already recorded to all backups from the cache. This module has two functions: one is an interface between NIGELOG and the application, the other is a management process of the internal cache.
2. Collecting the directory information
   This module (marked "**B**" in figure 2) scans an entire file system and finds directories where the backups can be written. When NIGELOG does not have the directory information, it is impossible to hide backups into the file system. NIGELOG, therefore, gives higher priority to this module, until it collects a certain number of directory information.
3. Output the log-info and detecting alteration
   This module does the following work.
   - output log-info
     If there exists a log-info which has not yet recorded in the log-files in the internal cache, it writes it in the log-file.
   - alteration detection
     This watches the log-files periodically and detects its alteration. It makes new "extended stat structure" information from the log-file and compares it with the past one.
     "Extended stat structure" is data structure which integrates stat structure which has status information about file with "Finger print" made by one-way hash function. We currently use MD5 Message-Digest Algorithm[7] for making finger print.
   - automatic restoration
     If any unexpected alteration is detected, this removes the altered log-file and restores the original from other correct backups.
   - periodical hiding
     This periodically moves the log-files to other directories and changes their

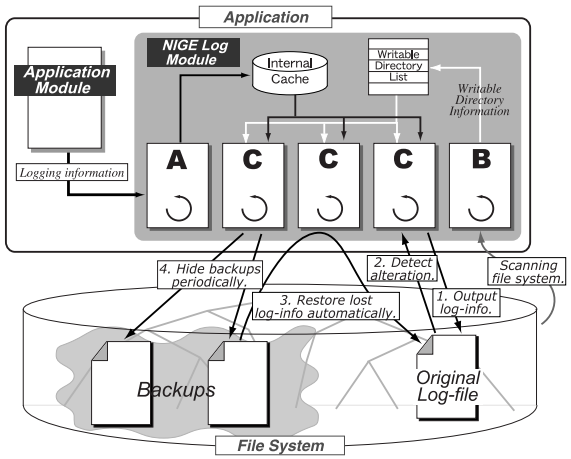

Figure 2. Overview of NIGELOG

names. The file names are randomly generated using `mktemp` function and the directories to hide are selected randomly from the directory information which was collected previously.

4.1 Generality of NIGELOG

There are various applications which generates the log-info. It is, therefore, desirable that logging information protection method is applicable to various applications. NIGELOG was carefully designed and implemented by considering this point.

- standard libraries NIGELOG was implemented with standard software libraries. It follows the POSIX standard as much as it can including Pthread library.
- C++ class library Also, NIGELOG was implemented as a C++ class library. Thus, it is easy to be used in the applications and to extend its function.
- minimum interface NIGELOG provides only two interface functions: an initialization function and an output function. It minimizes programmers load to use this library.
- contents independence NIGELOG does not depend on the contents of the log-info. Therefore, it is possible to use NIGELOG even if the log-info is compressed or encrypted.

These features make it easier to apply NIGELOG to various applications running on UNIX-based environment. Currently, NIGELOG runs on SGI IRIX6.5 and Sun Microsystems Solaris2.6.

5 Discussion

5.1 Advantages

As is described in the previous sections, NIGELOG produces multiple backups of an original log-file, hides them into directories, and periodically moves them to other directories. And NIGELOG watches all log-files periodically, NIGELOG can detect unexpected alteration of the log-files. When the alteration are detected, NIGELOG can restore the lost log-info.

Also, it is easy to introduce NIGELOG into applications because NIGELOG does not require any additional hardware and special operating system support. Moreover, NIGELOG was implemented just by using standard software libraries. As was mentioned in section 4.1, NIGELOG is applicable to various applications running on UNIX environments.

5.2 Limitations

Unfortunately there are some cases that NIGELOG fails to protect the log-info. One is that the hidden directories and names of all backups are already known to the intruder. It is, however, extremely difficult to know them. Another case is that the application using NIGELOG is terminated for some reason. In such case, NIGELOG cannot perform alteration detection or automatic restoration continuously. However, any daemon processes cannot work when they are terminated.

Second, the security level of the log-files depends on the number of directories the backups can be written. For example, if the application runs with normal user's authority, NIGELOG can write the backups only in the directories where the user's write permission is on. Therefore, NIGELOG should be used in the applications which are executed by the super user. In this case, the number of directories are huge and the security level becomes much higher.

Finally, the application using NIGELOG consumes more CPU resource than that without NIGELOG because NIGELOG has to do additional processes continuously. To reduce the load, users can adjust the interval that NIGELOG watches and moves the backups. However, the interval time is inversely proportion to the secure level. It is difficult to reduce the load dramatically.

The average load of NIGELOG was measured as follows. We executed NIGELOG for an hour and measured the CPU load in every 5 seconds. The load average was calculated from these data (Table 2).

5.3 Future Work

In order to keep log-info more securely, we need to implement more secure mechanism for backup.

Table 2. Average CPU load of NIGELOG

Interval of alteration detection	Interval of hiding backup	Log-file size at the beginning	Number of backups	The status of outputting log-info	Average load in one hour
2 sec	2 sec	***2KB***	5	No Output	**0.06**
2 sec	2 sec	2MB	***1***	No Output	**1.03**
10 sec	***10 sec***	2MB	5	No Output	**1.38**
2 sec	2 sec	2MB	5	No Output	**1.88**
2 sec	2 sec	2MB	5	***40byte/sec***	**2.09**

- access restriction If a file is removed while it is opened, the program can use the file normally but users cannot access to the file any more. This can be applied to NIGELOG in order to hide backups from intruders.
- dummy backups The one way to ensure the safety of log-info is to increase the number of backups. It is, however, undesirable because it increase the system's load. To overcome this problem, we are thinking to use a dummy backup which does not continuously perform protection process.
  The idea is described as follows. Suppose that a dummy backup was made at time MT and current time is CT. Then, the log-info generated until MT was kept in the dummy backup and the log-info generated between MT and CT is kept in memory. If NIGELOG verifies the contents of the dummy backup, it can be used as the source information for restoration on the occasion of log-info restoration process.

6 Conclusions

This paper proposed NIGELOG: a novel method to protect log-info more securely. The main features of NIGELOG are summarized as follows:

- making multiple backups
- hiding them into arbitrary directories
- moving them periodically

NIGELOG can automatically restore the original logging information by using the backups even if the original file is altered or deleted by intruders. Also, NIGELOG is designed and implemented by considering its generality.

NIGELOG protects the logging information more securely and therefore the intrusion detection by system administrators or intrusion detection systems becomes more reliable.

Programmers can extend NIGELOG easily. For example, it could be used with write-once media or it could be extended as a simple intrusion detection system.

Intrusions increase more and more. NIGELOG itself cannot protect all such intrusions. However, since NIGELOG can protect the log-info more securely, it will help to make more secure systems.

References

[1] Bruce Schneier, John Kelsey: Cryptographic Support for Secure Logs on Untrusted Machines, *The Seventh USENIX Security Symposium Proceedings*, USENIX Press, pp. 53-62, (1998).

[2] G.H. Kim, E.H. Spafford: The Design and Implementation of Tripwire: A File System Integrity Checker *Purdue Technical Report CSD-TR-93-071*, Purdue University, (1993).

[3] Core SDI S.A.: secure syslog, http://www.core-sdi.com/Core-SDI/english/slogging/ssyslog.html

[4] syslog-ng, The free software company BalaBit, (1999), http://www.balabit.hu/products/syslog-ng.html

[5] Hewlett-Packard Company: HP Praesidium/VirtualVault White Paper, (1998), http://www.hp.com/security/products/

[6] B. Nichols, D. Buttlar, J.P. Farrell: Pthread Programming A POSIX Standard for Better Multiprocessing, p. 284, O'Reilly, (1996).

[7] R.L. Rivest: RFC1321: The MD5 Message-Digest Algorithm, MIT Laboratory for Computer Science and RSA Data Security, Inc., (1992).