Vol. 41 No. 3 情報処理学会論文誌 Mar. 2000

逃げログ: 削除まで考慮にいれたログ情報保護手法

高田哲司[†] 小池英樹[†]

不正侵入検知にはログ情報が必要不可欠である．不正侵入検知システムは，ログ情報に不正侵入の痕跡情報が記録されることを仮定し，ログ情報を分析することで不正侵入の有無を判断する．一方，不正侵入者は不正侵入が発覚するのを防ぐために侵入の痕跡を改ざんまたは削除する．したがってなんらかの方法を用いてログ情報を保護する必要がある．

そこで本研究では，新たなログ情報保護手法を提案する．本手法では，ログ情報のバックアップを作成するとともに，その改ざんを検知する．さらにログ情報が改ざんされた場合にはバックアップから自動的にログ情報を復元する．

これによりログ情報が削除されたとしても，ログ情報を保護することが可能になり，不正侵入者によるログ情報の改ざんを困難にする．また本手法の実装では，使用したライブラリを含めて標準規格に準拠した関数群を使用し，C++のクラスライブラリとして実装した．そのため本手法は，多くのプラットフォームで動作し，様々なアプリケーションに適用することが可能である．

NIGELOG: A method of protecting logging information from even their removal

Tetsuji Takada[†] and Hideki Koike[†]

Logging information is necessary for intrusion detection. Intrusion detection systems are investigating them in order to judge whether intrusion had occurred or not. It, therefore, must protect logging information, because there is much possibility that intruder compromise them. Though some systems had been proposed for protecting logging information until now. These systems detect their modification and make hard to modify them. If logging information had been modified or removed, original logging information are lost. This is undesirable for computer security. It is necessary to investigate them why intrusion had occurred. In this research, we propose logging information protection system that makes it possible to protect them even if they had removed.

When logging information was compromised by someone, our system make it possible to protect them which way is automatically restoring them from backups. Moreover, it is possible to recognize an occurrence of compromising them nearly in real-time. We implement our proposed method as a C++ Class. It is possible to integrate logging information protecting system into various applications.

1. はじめに

不正侵入検知は，ログ情報の収集とログ情報の分析の二大要素により成立する．つまり，内容の確かなログ情報なしに不正侵入を検知することは不可能である．一方で，ログ情報は不正侵入の痕跡情報を含んでいるため，不正侵入者によって改ざんされる危険性がある．また国内ではログ情報の保存義務化に関して様々な議論が行われたのは記憶に新しい．このようにログ情報は，不正侵入検知にとって必要不可欠なデータであり，その内容が正確である必要がある．ゆえに，不正侵入者による改ざんや削除からログ情報を保護する必要がある．

ログ情報保護手法として，これまでに様々なシステム[1)2)3)]が提案されている．これらのシステムの多くは暗号を使用することにより，ログ情報の改ざんを困難にしている．しかし，これらの手法ではログ情報が無差別改ざんを受けたり，削除された場合，改ざんを検知できたとしても，それまでのログ情報を復元することは不可能である．

システム設定ファイルやプログラムは，その更新頻度が極めて低いため，バックアップを行うことでこの問題に対処可能だが，ログファイルは随時更新されるため，定期的なバックアップによる方法では保護不可能である．

† 電気通信大学大学院 情報システム学研究科
Graduate School of Information Systems, University of Electro-communications

これらの問題をふまえ，我々はUNIX系OSが動作する計算機を対象とした新たなログ情報保護手法「逃げログ」を提案する．本手法を用いることにより，たとえログ情報が削除された場合でもそれを復元することが可能になり，不正侵入者によるログ情報の改ざんを困難にする．またログを出力するシステムの多様性を考慮し，我々は本手法を付加的なハードウェアを必要とせず，標準規格に準拠したライブラリを使用し，C++クラスライブラリとして実装した．この特徴により，本手法は多くのUNIX系OS上で動作可能であり，同時に種々のアプリケーションに容易に組み込むことが可能である．

以下では，第2章で従来の保護手法の問題点を整理し，第3章でこれらの問題を解決する本手法の構想を述べ，第4章では本手法の実装例について述べる．さらに第5章で考察を行う．

2. 従来のログ情報保護手法とその問題点

本章では，これまでに提案されているログ情報保護手法についてふれ，その問題点を整理する．

- 暗号
  暗号を用いてログ情報を保護するシステムはいくつか提案されている[1)2)]．この手法により実現される保護機能は，ログ情報の改ざん検知と，不正侵入者による意図的な改ざんを困難にすることの二つである．
  しかしこれらの機能では，ログ情報が無差別に改ざんされたり，削除された場合，失われた情報を復元する事ができない．これでは仮にログ情報の改ざんを検知したとしても，改ざんされたログ情報は消失してしまうため，不正侵入に対する調査は不可能である．不正侵入対策では，不正侵入を検知するだけでは不十分であり，犯人追跡，被害状況や，セキュリティホールの特定などの調査も必要不可欠である．よって改ざんされる以前のログ情報が確実に保護される必要がある．したがって，本手法では改ざんによってログ情報を失ってしまうため，十分な保護を実現しているとはいえない．
- Write-Once メディア
  Write-Once メディアとは，一度記録した情報は決して改ざんできないメディアのことであり，紙への印刷やCD-R等がその例として挙げられる．この特徴を利用することで，比較的簡単にログ情報の改ざんを不可能にすることが可能である．
  しかし，この手法における問題点は，Write-Once メディアに記録する情報が正確であることを保証する必要があるということである．定期的にログ情報をWrite-Once メディアにバックアップする方法は，この要件を満たすことにならないため，ログ情報保護手法として適切な方法ではない．また仮になんらかの方法によって記録する情報の正確さが保証できたとしても以下の問題を回避することは不可能である．
  (1) 改ざん検知は不可能
  Write-Once メディアはあくまで改ざん不可能な記録メディアであり，改ざん検知を行うことはできない．
  (2) メディアのメンテナンスが必要
  Write-Once メディアは一般に大容量だが，限界はある．ゆえに人手によってメディアのメンテナンスを行う必要がある．
- 信頼できる計算機への移送
  本手法は，安全を保証された計算機に生成されたログ情報を移送することでログ情報を保護する手法である．この手法の安全度はログ情報を保存する計算機の安全度に依存するが，Write-Once メディアと同等の安全性を確立することも可能である．
  しかし，この手法における問題点は，ログ情報移送時におけるログ情報の安全を保証しなければならないことである．この安全性を保証する一方法として暗号が挙げられるが，暗号が実現する機能は移送情報の漏洩防止と改ざん防止であり，確実な通信を保証することではない．したがって通信を不可能にするようなDenial-of-Service等の攻撃を受けた場合，ログ情報が欠落する可能性がある．
- アクセス制御
  HP社のVirtualVault Operating System[3)]は，アクセス制御を用いてログ情報を保護している．この手法では，ログ情報を改ざん不可能であると仮定した場合，Write-Once メディアによる手法と同等の安全性が保証可能になる．しかし，アクセス制御を実現するためにはOSやハードウェアによる支援が必要になり，汎用的な方法ではないという問題がある．
  またログ情報が改ざん困難だと仮定するならば，上記の問題に加えて，以下に挙げる問題も発生する．
  (1) 改ざん検知が不可能
  (2) 改ざんにより失われたログ情報は復元不

可能

表 1 様々なログ情報保護手法とその特徴
Table 1 Various Log-file protection methods and its features

手法＼特徴	改ざん不可能	改ざん困難	改ざん検知	汎用性	情報復元
Write-Once メディア	○				
信頼できる計算機への移送	○				
アクセス制御	○				
暗号		○	○	○	

これらの考察をまとめると**表 1** になる．この考察から，我々はログ情報保護処理には以下のような機能が必要と考える．

- 改ざん困難
  ログ情報を保護するため，最低でもログ情報の改ざんが困難でなければならない．
- 改ざんされたログ情報の復元
  不正侵入調査作業を行うためには，改ざん以前のログ情報を確実に保護する必要がある．よって改ざんによってログ情報が消失したとしても，それを復元可能にする必要がある．
- 改ざん検知
  ログ情報の改ざんが検知可能でなければならない
- 汎用性
  ログ情報保護処理は不正侵入の増加に伴い，その必要性を増している．したがってその処理は汎用的であり，種々のアプリケーションに容易に組み込み可能であることが望ましい．

3. 逃げログ - 構想

ログ情報を改ざんや削除から保護するためには，その改ざんを不可能にするかバックアップを作成する必要がある．しかし付加的なハードウェアや OS の支援なしにログ情報の改ざんを不可能にするのは困難である．そこで本研究では，従来のログファイルを利用し，そのバックアップを複数作成することでログ情報を保護する．さらに原本も含めた全ログファイルを定期的に監視し，その改ざんを検知する．改ざんが検知された場合には，改ざんされていないバックアップから自動的にログ情報を復元する (**図 1**)．これらの機能によって不正侵入者がログ情報を改ざんし，侵入痕跡を削除することを困難にする．

以降の節では，これらの機能について述べる．

3.1 バックアップ作成

不正侵入に対し，それを検知するだけでは不十分であるのは前述したとおりである．不正侵入検知後はそれを様々な点から調査し，その結果を今後の不正侵入対策に反映する必要がある．したがってログ情報の改ざんにより，ログ情報が失なわれることは望ましくない．そこで本手法ではログ情報のバックアップを作成し，この問題に対処する．

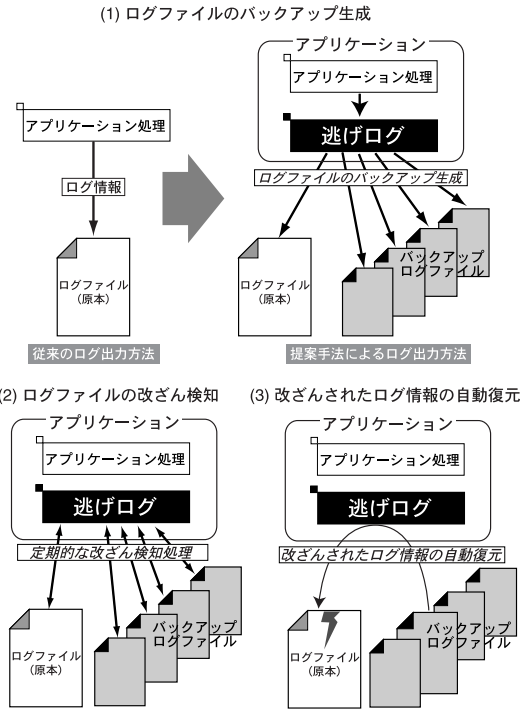


図 1 逃げログの構想
Fig. 1 Concept of NIGELOG

本手法におけるバックアップ作成の特徴は以下の通りである．

- 複数作成
  一つのバックアップでは，バックアップが既知となった場合，原本とバックアップを同時に改ざんまたは削除することで，ログ情報を削除可能である．そこでログ情報の安全性を高めるため，複数のバックアップを作成する．
- 原本と同一の内容を保持
  改ざんはいつ発生するか予想できない．したがって常にログ情報が復元可能である必要がある．定期的なバックアップによる方法では，この要件を満たすことは不可能である．
  そこで本手法では，常に原本のログ情報と同一の内容を保持するようにバックアップを作成する．

3.2 隠　　蔽

本手法では，ログ情報を復元可能にするためにバッ

クアップを作成する．しかし，仮に不正侵入者が全てのバックアップを知り得た場合，それらを同時に削除することでログ情報を復元不能にすることが可能であり，それはバックアップを複数作成したとしても同様である．

そこでバックアップの安全性を高めるため，本手法ではそれらをファイルシステム内に隠蔽する．これにより，不正侵入者による全バックアップの発見を困難にし，ログ情報の改ざんや削除を困難にする．

本手法で用いた隠蔽方法は以下の通りである．

(1) 任意の数のバックアップ
バックアップの数は任意とする．

(2) 任意のディレクトリ
書き込み可能な任意のディレクトリにバックアップを作成する．

(3) ランダムなファイル名
バックアップのファイル名は，システムが無作為に作成する．

(4) 定期的な再隠蔽
ファイルシステムに隠蔽されたバックアップは，半永久的に特定ディレクトリで保持されるのではない．ユーザが指定した間隔で定期的に異なるディレクトリに異なるファイル名で再び隠蔽される．

現在，一般に利用されている UNIX 系 OS は，インストール直後の状態でも 1000 以上のディレクトリを持つ．またファイルサーバであれば，その数は数倍にもなりうる．これらの方法によりバックアップをファイルシステム内に隠蔽し，その安全性を確保する．

3.3 改ざん検知

本手法では，バックアップを含めた全ログファイルを定期的に監視し，その改ざんを検知する．これにより迅速に改ざんを認識することが可能になる．

ログ情報の改ざん検知には，ファイルの属性情報をもつ **stat** 構造体を使用する．**stat** 構造体とはファイルの状態情報を保持した構造体であり，ファイルの大きさや許可権，更新時刻などの情報を保持している．しかし **stat** 構造体だけでは改ざんを検知できない場合があり得るため，ログファイルの全内容を入力として作成される “指紋情報” を導入し，これと **stat** 構造体を合わせたものを拡張 **stat** 構造体 (**図 2**) として定義した．これを改ざん検知に用いることでその機能をより確実にしている．現在，指紋情報の作成には MD5 Message Digest[8] を使用している．

この “ログ情報の改ざん発生” という事象は，本手法において非常に重要な情報である．なぜならばこの事象はログ情報保護処理を起動する契機となるからである．

図 2 拡張 stat 構造体
Fig. 2 Extended stat structure

本手法では改ざんを検知した場合，改ざんされていないバックアップからログ情報を自動的に復元する．これにより原本のログファイルは常に正確な情報を保持することになり，システム管理者や不正侵入検知システムに対して常に正確な情報を提供することが可能になる．

さらにこの事象を利用して付加的な保護処理を実行することにより，様々な応用も可能になる．その例を以下に挙げる．

- 簡易不正侵入検知システム
ログ情報の改ざんが発生したことを電子メイル等によりシステム管理者へ通知する
- より確実な保護のための処理
Write-Once メディアへの記録や，不正侵入の調査作業支援のための追加情報収集処理の実行

4. 実　　装

我々は，前述の構想をもとに本手法を実装した．実装においては，ログ情報を出力するアプリケーションの多様性を考慮し，種々のアプリケーションで使用できるよう C++のクラスライブラリとして実装した．また，複数のログファイルに対する処理の実時間性を実現するためには個々のバックアップに対する処理を並行処理することが望ましいと考えた．そこで今回の実装では，プラットフォームに依存せず，複数のプロセスを並行実行させるためのライブラリである Pthread[6] を利用し，並行処理を実現した．

以下では逃げログの実装について述べる．

4.1 三種の処理

本実装では，本手法における処理を三種類に分類し，これらを thread を用いて並行処理させる．この三種の処理について以下に説明を行う．

(1) ログ情報受付処理
本処理は，ログを出力するアプリケーションへのインタフェースを提供するとともに，内部キャッシュに関する管理処理を行う．その詳細は以下

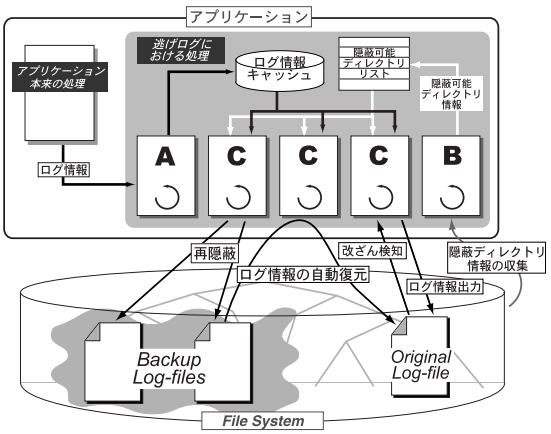


図 3 逃げログの処理概要
Fig. 3 Overview of NIGELOG modules

の通りである (**図 3** 処理 **A** に該当).

(a) アプリケーションからの出力されたログ情報を受け取り，逃げログの内部キャッシュ内に格納する

(b) 全ログファイルに出力されたログ情報を内部キャッシュから削除する

(2) 隠蔽可能ディレクトリ情報収集処理
ファイルシステム内を探索し，隠蔽可能なディレクトリ情報を収集する (図 3 処理 **B** に該当). なお，隠蔽可能ディレクトリ情報が存在しない場合，バックアップの隠蔽処理を行なえず，その安全性を保証できない．したがって一定数の隠蔽可能ディレクトリ情報が得られるまでは，本処理を優先的に行う．
なお隠蔽可能ディレクトリ情報が一定数収集された後は，定期的に処理を行う．

(3) ログ情報出力/改ざん検知処理
本処理は，ログ情報の出力，改ざん検知，自動復元，定期的な再隠蔽処理といったログファイルに対する処理を行う (図 3 処理 **C** に該当). この処理は，各ログファイルごとに thread が生成され，並行に処理が行われる．

次節では，ログファイルの保護処理を行う“ログ情報出力/改ざん検知処理”について詳細な説明を行う．

4.2 ログ情報保護処理

ログ情報保護処理は，四つの処理から成り立っている (**図 4** 参照).

- ログ情報出力
  ログ情報受付処理によって内部キャッシュに保存されたログ情報をログファイルに出力する．出力後直ちに改ざん検知のための基礎情報となる拡張 **stat** 構造体情報を取得する
- 再隠蔽
  バックアップを現在と異なるディレクトリに任意のファイル名で再隠蔽する．
  隠蔽時に必要となる情報は隠蔽ディレクトリとバックアップ用ファイル名の二つである．現在の実装では，隠蔽ディレクトリは，隠蔽可能ディレクトリ情報収集処理によって得られた情報からランダムに選択し，ファイル名は，mktemp 関数を用いて無作為に生成している．
  なお本処理はバックアップのみを対象としており，改ざん検知とは関係なく定期的に行われるが，改ざん検知時にもログ情報保護のため強制的に行う
- 改ざん検知
  ログファイルから拡張 **stat** 構造体情報を取得し，前回取得した情報と比較することで改ざんを検知する．取得した拡張 **stat** 構造体情報は，改ざんがなければ次回の改ざん検知処理のため保持されるが，改ざんが検知された場合は，ログ情報の復元後再び取得する
- 自動復元
  本処理は改ざんを検知した時に行われる．改ざんされたログファイルを削除し，改ざんされていないバックアップからログ情報を復元する．この処理は，原本のログファイルだけでなく，バックアップも含めた全ログファイルを対象としている

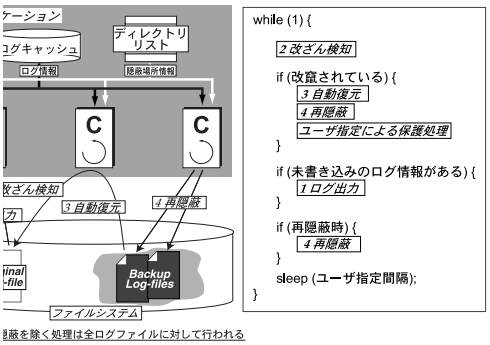


図 4 ログ出力/改ざん検知処理
Fig. 4 Log-file protection module

これらの処理を用いたログ情報保護処理の流れは図 4 右の通りである．まずはじめに現在のログ情報に対する改ざん検知処理を行う．

この処理によって改ざんが検知されなければ，内部キャッシュを調べ，未書き込みのログ情報があればそれらをファイルに記録する．さらに定期再隠蔽の必要があればログファイルの再隠蔽も行う．

改ざんが検知された場合，前述の処理を行う前にロ

グ情報の自動復元処理を行い，復元されたログ情報の安全性を確保するため，再隠蔽処理を行う．この一連の処理により，ログ情報は改ざん前の状態に復元される．またユーザの指定により，ログ情報の復元後に付加的な保護処理も行う．現在の実装では，電子メイルを使用した改ざん通知処理を実現している．

またこれまでの説明からわかるように，ログ情報保護処理において原本のログファイルとバックアップに対する保護処理上の違いは，隠蔽処理の有無だけである．

4.3 汎 用 性

ログ情報を出力するシステムは多数存在する．したがって種々のアプリケーションに本手法による保護処理を組み込み可能にするためには，その手法が汎用的である必要である．

そこで今回の実装では，以下の特徴により本手法の汎用性，および組み込みの容易さの実現を目指した．その特徴を以下に挙げる．

(1) 標準的なライブラリの使用
今回の実装で使用したライブラリとしてPthread[6] と Standard Template Library[7] (STL) が挙げられる．Pthread は並行処理を実現するライブラリで POSIX 準拠であり，STL は ANSI/ISO C++標準案に採用された C++のクラスライブラリである．どちらも標準規格のライブラリであり，様々なプラットフォームで使用可能である．また実装で使用している他のシステムコールについても可能な限り POSIX 互換の関数群を使用した

(2) クラスライブラリとして構築
本手法を C++のクラスライブラリとして実装した．したがって C++という制約はあるものの，種々のアプリケーションに組み込む際には単なる型として使用可能であると同時に，その拡張も容易である

(3) 最小限のインタフェース
現在の実装では，アプリケーションに対し二種類のインタフェースを提供している．それらは逃げログの初期化関数とログ情報出力関数である．これゆえプログラマは，最小限の負担で本手法をアプリケーションに組み込むことが可能である

(4) ログ情報の内容に非依存
本手法は，ログ情報の内容に依存しない．したがってログ情報が文字情報以外であっても保護可能である．この特徴を利用し，アプリケーション側でログ情報を暗号化または圧縮等の処理を行い，ログ情報の安全性をより一層高めることも可能である

これらの特徴により，UNIX 系 OS で動作する種々のアプリケーションのログ出力処理に本手法を組み込むことが可能である．現時点では，SGI 社の IRIX6.5 と Sun Microsystems 社の Solaris2.6 上での動作を確認している．

5. 考 察

本章では，本手法の利点と問題点について考察を行う．

5.1 利 点

本システムの利点を以下にまとめる．

- ログ情報の復元が可能
  本手法では，原本のログ情報と同一の内容をバックアップが保持しているため，ログ情報改ざん時はもちろん，ログ情報が削除されたとしても，改ざん前のログ情報を復元することが可能である
- ログ情報の改ざん検知が可能
  本手法では定期的にログ情報を監視しているため，ログ情報の改ざんを迅速に認識することが可能である．この事象は，改ざんされたログ情報の自動復元処理や，付加的なログ情報保護処理を行う契機として必要不可欠である
- 付加的なハードウェアは不要
  本手法では，ログ情報を保護するために付加的なハードウェアを必要としない．したがって，本手法によるログ情報保護処理の導入は容易である
- 汎用性
  4.3 節で述べたように，標準規格によるライブラリを使用し，C++のクラスライブラリとして実装したことにより，種々のアプリケーションに組み込むことでログ情報の保護処理を容易に実現可能である

これらの利点により,「不正侵入された場合，不正侵入によって権限取得が成功する以前のログ情報は，改ざん不可能であること.」というログ情報の保護において必要とされる要件[1] の一つを満たすことが可能になる．

5.2 問 題 点

本手法における問題点および限界について考察する．

- 保護不能になる可能性の存在
  何らかの理由により本手法を適用したシステムが終了した場合には，ログ情報の保護は不可能である．またバックアップを含めた全ログファイルが

不正侵入者によって既知となった場合は，その改ざんや削除が可能となりうる．しかし本手法において後者の実現が困難であるのはこれまでに述べてきた通りである．

- ログファイルの安全性は隠蔽箇所数に依存
  バックアップの隠蔽場所は隠蔽可能ディレクトリ情報からランダムに選択される．よって隠蔽可能なディレクトリ数が少ない場合，バックアップの隠蔽箇所も限定され，バックアップの安全性を低下させる可能性がある．またログを生成するアプリケーションが一般ユーザ権限で実行された場合，隠蔽可能ディレクトリの条件に User ID や Group ID による制約が加わり，問題をさらに悪化させてしまう．
- アプリケーションへの適用制約
  本手法は，処理を継続して行う必要があるため，コマンドなど短時間で処理を終了するアプリケーションのログ出力に適用することは困難である．しかしこの問題は，syslog デーモンに本手法を適用することで解決可能である．
- 負荷の増大
  本手法では，複数のログファイルに対して定期的に処理を行なうため，計算機の負荷は増大する．定期的に行う処理の間隔はユーザが指定可能なため，負荷調整は可能だが，ログ情報の安全性と反比例するため，負荷を大幅に下げることは困難である

  表 2 は，逃げログを適用した簡易版 syslog デーモンを作成し，間接的に逃げログの性能を測定した結果である．測定機器は Silicon Graphics 社の **O₂** ワークステーションで CPU R5000 180MHz, Memory 128MB, HardDisk 6GB である．また測定計算機をネットワークから隔離し，通常動作させている種々のデーモン処理は動作させた条件下で試験用 syslog デーモンおよびログ出力用プログラムを一時間動作させ，その間の負荷の平均の値（ゲシ・反転部）を基準とし，計算機の負荷値を性能として測定した．表 2 は，上から二段目を変動させると思われる要因をそれぞれ個別に変更して，それぞれの負荷を計測した結果である．この結果から，定期的にログ情報の保護を行う処理が主に計算資源を消費していることが明白であり，特にログファイルの大きさによる影響が大きいことが明らかである．
- 保護可能なログファイルの適用条件
  大きく二つの条件が存在する．一つは，逃げログが保護可能なログファイルは，ログ情報を追記するログファイルのみに制限される．つまり UNIX 系 OS においてユーザの計算機利用状況を記録している wtmpx ファイルは，その形式がデータベース形式であるため，NIGELOG にて保護することは不可能である．

  もう一つは，ログファイルの大きさが運用時の計算機の負荷に大きな影響を与えることである．したがって計算機の負荷を増大させないためには，ログファイルの大きさを一定の大きさ以下に保ち続ける必要がある．

表 2 逃げログの性能評価
Table 2 Average CPU Loads of simple syslog daemon using NIGELOG

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

これらの問題点から，ログ情報の安全性を高めるためにはバックアップファイルを確実に保持する必要があり，そのためには一定数以上の書き込み可能なディレクトリの存在および CPU 性能等の計算機資源が豊富にあることが望ましい．

5.3 より確実な保護に向けて - 今後の課題

本手法を用いても，ログ情報を保護不能にする脅威は依然として存在する．これらに対抗する手法として，以下に挙げる方法を提案する．しかし，これらの手法の導入は，汎用性や導入の容易さを損なう可能性がある．

- メモリへのバックアップ
  本手法では，バックアップをファイルシステムに生成しているため，付加的なハードウェアを必要とせず，種々のアプリケーションにおけるログ情報保護処理の導入を容易にしている．しかし，この特徴はファイルシステムが使用不能になるとログ情報を消失するという危険をかかえることにもなる．

  この対策法としてメモリへのバックアップがあげられる．その容量はファイルシステムと比較すると大幅に制限されるが，最新のログ情報を固定量保持することで，ファイルシステムに対する依存を補完をすることは可能である．

  なおログ情報の復元先は，付加的な保護処理の実装により Write-Once メディアへの記録が可能なため，ファイルシステムへの依存はない

- バックアップのアクセス困難化
  本手法を用いても，不正侵入者がバックアップにアクセスすることは可能であり，これがログ情報の改ざんを可能にする一因になっている．よってより確実な保護方法として，バックアップファイルへのアクセスを不可能にすることがあげられる．その一手法として，一時的なデータファイルを作成する方法を応用する方法がある．この方法は，プログラムがファイルを作成した後，すぐにそのファイルをディレクトリエントリから削除する．これにより該当ファイルはファイル名が存在しなくなるため，不正侵入者は該当ファイルにアクセスすることが不可能になるが，プログラムからは通常通り使用でき，バックアップのためのログ情報を保持することが可能である
- 疑似バックアップによる錯乱
  全バックアップを発見困難にする方法として，保護処理を行わないが，ある時刻までの正確なログ情報を持つ疑似バックアップをファイルシステム内に生成し，保護処理を行っているバックアップファイルの発見を錯乱し，困難にする方法がある．この方法は，全バックアップの発見を錯乱するだけでなく，計算機の負荷を高めることなくバックアップの数を増やすことを可能にする．これらのバックアップは保護処理が行われないため，原本のログ情報と差分ができてしまうが，この方法とメモリへのバックアップを併用すると以下に挙げる方法で保護処理を行っているバックアップと同様に原本と同一のログ情報を保持することが可能になる．
  保護処理を行わないバックアップが作成された時刻を **T** とすると，ログ情報は以下のように保持される
  (1) 時刻 **T** 以前のログ情報は保護処理なしのバックアップファイルに保持
  (2) 時刻 **T** 以後のログ情報はメモリ上に保持
  これらの情報をたしあわせることで，完全なログ情報として利用可能であり，改ざん検知処理と組み合わせることで，ログ情報自動復元時の復元情報として使用することが可能である．

6. おわりに

本論文では，ログ情報の削除まで考慮にいれた新たなログ情報保護手法「逃げログ」を提案した．本手法では以下の方法によりログ情報を保護する．

- バックアップの作成
- バックアップの隠蔽
- 改ざん検知
- ログ情報の自動復元

これらの特徴により，不正侵入者が不正侵入の発覚を防ぐためにログ情報すべてを削除したとしても，ログ情報を保護することが可能になる．したがってシステム管理者が手作業でログ情報を調査したり，不正侵入検知システムを使用することで確実に不正侵入を検知することが可能になる．

またログ情報の改ざん検知機能を利用し，ログ情報の自動復元のほか，システム管理者への改ざん通知による簡易不正侵入検知機能，Write-Once メディアへのログ情報記録等の付加的な保護処理も実現可能である．

さらに本手法は，付加的なハードウェアを必要とせず，標準規格によるライブラリを用いて C++のクラスライブラリとして実装されているため，ログ情報保護処理の導入は比較的容易であると同時に，種々のアプリケーションへの組み込みも容易である．

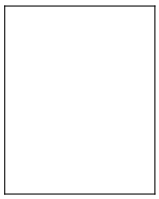
今後，不正侵入の増加とともにログ情報の重要性は増し，ログ情報保護の必要性も高まりつつある．我々はこのような状況において，従来の手法よりもより確実にログ情報を保護する新たな一手法を提案した．

参考文献

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) Core SDI S.A.: secure syslog, http://www.core-sdi.com/Core-SDI/english/slogging/ssyslog.html
3) Hewlett-Packard Company: HP Praesidium/VirtualVault White Paper, (1998), http://www.hp.com/security/products/
4) 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).
5) syslog-ng, The free software company BalaBit, (1999), http://www.balabit.hu/products/syslog-ng/
6) B. Nichols, D. Buttlar, J.P. Farrell: Pthread Programming A POSIX Standard for Better Multiprocessing, p. 284, O'Reilly, (1996).
7) D.R. Musser, A. Saini, A. Stepanov: STL Tutorial and Reference Guide - C++ programming with the standard template library, p.400,

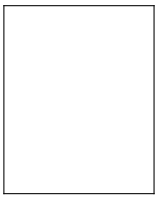
Addison-Wesley Publishing, (1996).
8) R.L. Rivest: RFC1321: The MD5 Message-Digest Algorithm, MIT Laboratory for Coumputer Science and RSA Data Security, Inc., (1992).

(平成？年？月？日受付)
(平成？年？月？日採録)



高田　哲司（学生会員）

1995 年電気通信大学大学院電気通信学研究科電子情報学専攻修士課程修了．1997 年電気通信大学大学院情報システム学研究科情報システム運用学専攻博士課程入学，現在同博士課程在学中．情報視覚化の研究に従事．情報視覚化、情報セキュリティに関心がある．



小池　英樹（正会員）

1991 年 東京大学大学院工学系研究科情報工学専攻博士課程修了．工学博士．同年電気通信大学電子情報学科助手．1994 年同大学大学院情報システム学研究科助教授．現在に至る．1994 - 1996,1997 年 U.C. Berkeley 客員研究員．情報視覚化の研究に従事．特に視覚化へのフラクタルの応用，情報検索システム，パーセプチュアルユーザインタフェース，情報セキュリティに興味を持つ．1991 年 日本ソフトウェア科学会高橋奨励賞受賞．IEEE/CS，ACM，日本ソフトウェア科学会各会員．