コンピュータセキュリティ 5－3
（1999．5．21）

# 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

電気通信大学大学院 情報システム学研究科

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，ログ情報を解析し，不正侵入が発生しているか否かを判断する．一方で，不正侵入者は侵入の痕跡を削除するために，ログ情報の改ざんを行なおうとする．したがってなんらかの方法でこれ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．本手法は，ログ情報のバックアップを複数作成し，それらをファイルシステム内に隠蔽することでログ情報を保護する．これによりログ情報の保護だけでなく，ログ情報改ざんの検知，さらには改ざん時のログ情報の自動復元も可能にする．また本手法をC++のクラスとして実装した．これにより様々なアプリケーションに本手法を適用することも可能である．

# NIGE Log - A method of protecting logging information from even their removal

TETSUJI TAKADA and HIDEKI KOIKE

The Graduate School of Information Systems
University of Electro-Communications

Logging information is essential to perform intrusion detection. Intrusion detection system inspects various logging information in order to detect intrusions. On the other hand, intruder would try to modify them in order to remove his intrusion's trails. We must protect them in some techniques.

In this paper, we propose new logging information protection method. It uses two processes to protect them against intruder. The one is to make more than one their backups. The other is to conceal these backups into file system. This method is not only enables to protect logging information from attacker's malicious modification, but it enables to detect their modification and recover modified information from backups automatically. We had implemented our method as C++ Class. It, therefore, is possible to integrate our method to 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，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と言える．一方で不正侵入者側の視点から考えると，不正侵入が発覚しないようにするためにはログ情報に記録される不正侵入の痕跡情報を削除することが必要不可欠となり，これを行うためのツールも存在する．また国内では不正侵入に対するログ情報の保存義務の立法化に対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が不正侵入検知にとって必要不可欠な情報であることは明確であり，これらの観点からも，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．

これに対し，ログ情報を保護するシステムは存在する[1)3)]．これらの保護システムは，主としてログ情報の改ざんを困難にすることに焦点がおかれている．これ自身は非常に重要な機能である．しかし，ひとたびログ情報が改ざん/削除された場合，ログ情報の一部は失われてしまい，それを復元することは不可能である．通常のファイルに関しては，バックアップや原本のメディアから復元すればよいが，ログ情報は随時更新されるため，ある時刻におけるバックアップが存在したとしても，その後に新たなログに情報が追加されている可能性があり，単純にバックアップから復元するだけでは不十分であることは明確である．

ログ情報の改ざん/削除に対する対策としてよく知られている方法には，以下の二つが挙げられる．

- Write-Onceのメディアにログ情報を保存する
- ネットワークを通じて，安全であるとされる計算

機にログ情報を移送する

しかし，これらの方法にも問題がある．Write-Once のメディアに保存するには何らかの契機が必要であり，その契機が適切でなければログ情報が欠落してしまう可能性がある．cron 等の使用による定期的なバックアップではログ情報が欠落する可能性があるのは明確である．

また安全であると想定される計算機にネットワークを通じてログ情報を移送する方法も，通信を行うという点でさまざまな問題をかかえることになる．

そこで我々は，不正侵入される傾向の高い UNIX 系 OS が動作するサーバ計算機のログ情報を対象にした新たなログ情報保護手法「逃げログ」を提案する．本手法では，ログ情報の改ざん/削除を困難にすることでログ情報を保護する．本手法の利点は大きく二つ挙げられる．一つは改ざん/削除されたログ情報を復元できることであり，もう一つは信用できる計算機も Write-Once メディアも不要であるためその導入が容易であることである．

以降では，第2章で本手法の構想を，第3章では本手法の実装例について述べ，第4章で本手法による利点と限界について考察する．

## 2. ログ情報保護手法 - 逃げログ

### 2.1 構　　想

本提案手法の構想は，ログ情報出力時に原本となるログファイルにログ情報を出力するのと同時にバックアップ用のログファイルにも同様のログ情報を出力することで内容に差分のないバックアップを生成することである．さらにログファイルの改ざん/削除に対する検知機能を持たせ，それらの事象を検知した際には，ログ情報を自動的に復元させる機能を付加したものである．

具体的には，以下の通りになる (図1参照)．

(1) 本手法を利用したシステムは，起動時に原本のログファイル以外に複数のバックアップ用ログファイルを作成する．その後システムから出力されるログ情報は，原本およびバックアップの全ログファイルにほぼ同時に記録される．つまり実時間でログ情報のバックアップが生成される．

(2) 本手法では，ログ情報の改ざんを検知するために，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てのログファイルを定期的に監視する．これによってシステム管理者はログ情報の改ざんや削除をほぼ実時間で認識することが可能になる．

(3) ログ情報の改ざんを検知した場合には，直ちにバックアップ用のログファイルからログ情報を復元する．

しかしこのままではバックアップも含めた全てのログファイルを改ざんまたは削除することで不正侵入の痕跡が削除可能である．そこで本手法では，原本のログファイルはその所在が既知でよいとする一方で☆，全てのバックアップ用ログファイルは，ファイルシステム内に**隠蔽**する．これによってバックアップ用ログファイルの所在がわからなくなり，結果として不正侵入者は，ログ情報を改ざんすることによって不正侵入の痕跡情報を削除することが不可能になる．

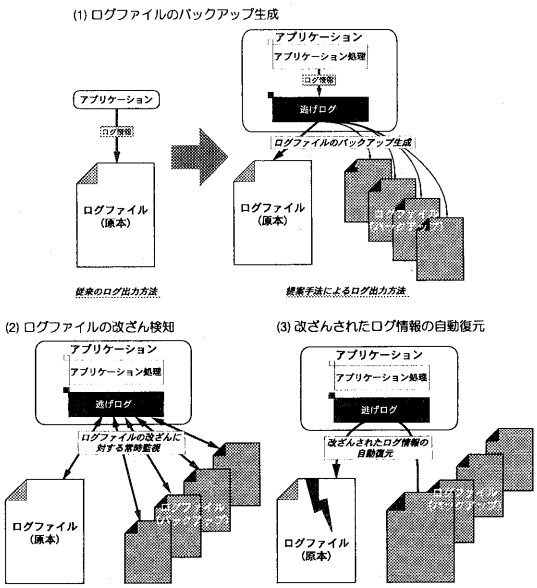


図1　逃げログの基本構想

また本手法を用いることでログファイルの改ざんに対する監視が定期的に行なわれる．そのため実時間でログファイルに対する改ざん/削除を認識することが可能になる．これにより，システム管理者は不正を認識し，さらなる調査を行うための契機を得ることが可能になる．

### 2.2 バックアップ用ログファイルの隠蔽方法

前節で述べたように，本手法ではバックアップ用のログファイルを，計算機のファイルシステム内に**隠蔽**する．全てのログファイルが同時に削除された場合には本手法であっても，ログ情報の復元は不可能である．したがってバックアップ用ログファイルは，可能な限り発見困難であるように隠蔽する必要がある．

本手法では以下のような方法でバックアップ用ログファイルを隠蔽する．

(1) 任意のディレクトリに隠蔽
書き込み可能なディレクトリのうちの任意のディレクトリにログファイルを作成する

(2) 任意のファイル名で隠蔽

☆ 既知であることを限定するわけではない

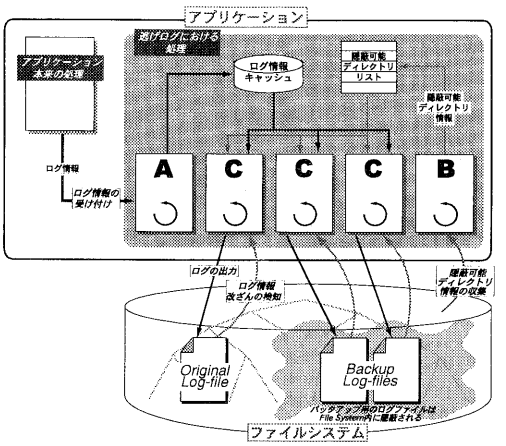


図2　逃げログ処理概要

バックアップ用ログファイルのファイル名は任意とする

(3)　任意の数のバックアップ用ログファイルを作成
バックアップ用ファイルの数は任意とする

(4)　定期的なログファイルの再隠蔽
隠蔽されたログファイルは，半永久的に一つのディレクトリに存在するのではなく，定期的にファイルシステム内を移動する☆

今日，UNIX系OSが稼働しているサーバ計算機は，インストール直後の状態でも1000以上のディレクトリが存在する(表2.2参照)．また本手法では，一度でも

表1　UNIX系OSがインストールされた計算機のディレクトリ数

| ディレクトリ数 | 計算機環境 |
|---|---|
| およそ1600 | RedHat Linux release 5.2 標準インストール ユーザ未使用 |
| およそ8000 | Sun Solaris 2.6 20ユーザが使用するファイルサーバ |

ログ情報の改ざんに失敗した場合，システム管理者へ通報といった，ログ情報に対するより安全な保護処理を自動的に実行することが可能である．これらの特徴から，本手法により実用上十分であり，かつ簡単にログ情報の保護が実現可能になる．

### 2.3　ログ情報の改ざん検知

ログ情報の改ざん検知にはファイルの属性情報を保持しているstat構造体を使用する．しかしこの情報は比較的簡単に改ざんできるため，これだけでは逃げログに改ざんを検知されずにログ情報を改ざんできる可能性がある[2]．

そこで本研究では，stat構造体にログファイルの内容をもとに生成される指紋情報を付与した拡張stat構造体を作成し，これを改ざん検知情報として用いる．ここでいう指紋情報とは，ログファイルの全内容を入力とし，一方向ハッシュ関数で生成される情報であり，現在の実装ではMD5 Message-Digest[6]を使用している．これによって時刻情報が改ざんされ，stat構造体の情報がまったく変化していなくても，ログ情報の改ざんを検知することが可能になる．

## 3.　実　　装

我々は，逃げログをpthreads[4]を用いたC++のクラスとして試作実装した(図2参照)．pthreadとは複数のプロセスを並行に実行させるための，ポータブルでプラットフォームに依存しないライブラリであり，POSIX規格で標準化されたものである．これを用いて2章で述べてきた逃げログの機能を個々に並行に実行させる．

threadが実行する処理は以下の三種類にわけられる．

- ログ情報受け付け処理
本処理は，アプリケーションから出力されたログ情報を受け取り，逃げログ処理内の内部キャッシュに保存する(図2Aに該当)．
- 隠蔽ディレクトリ情報収集処理
ファイルシステム内を走査し，隠蔽可能なディレクトリ情報を収集する処理である．この情報も逃げログ処理内の変数に保持され，ログファイル生成および移動時に使用される(図2Bに該当)．
- 改ざん検知/ログ情報書き込み処理
各ログファイル毎に一つのthreadが生成され，処理が行われる．本処理においてログファイルの改ざん検知，改ざん検知時の自動復元処理，定期的な再隠蔽を行っている(図2Cに該当)．その詳細について以下に説明する(図3参照)．

(1)　ログ情報出力処理
ログ情報受け付け処理によって内部キャッシュ内に保存されたログ情報は，本処理によって即座に各ログファイルに記録される．なお内部キャッシュ内のログ情報は，全ログファイルに記録された時点で削除される．

(2)　ログ情報の改ざん検知
本処理では，定期的にログファイルから拡張stat構造体情報を取得し，前回の処理において得た情報と比較することで改ざんを検知する．この処理により最悪でも改ざん検知監視間隔後には改ざんを検知することが可能になる．
またこの改ざん検知を契機として，付加的な処理を自動的に行うことも可能である．今回の実装では，セキュリティ上の問題で限定した処理しか実装しなかった．しかしユーザは実装を改良することで，必要な処

☆ ファイルシステム内をログファイルが不正侵入者による改ざんの魔の手から逃げまわっている．これが逃げログの由来である．

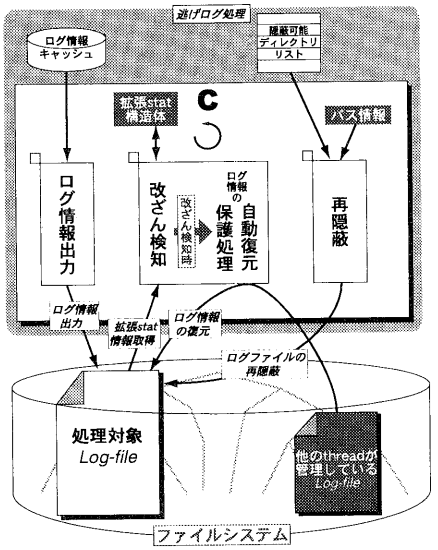


図3　ログファイルに対する処理概要

理を追加することは可能である．

(3)　改ざんされたログ情報の自動復元
改ざんが検知された時に実行される処理で，改ざんされたファイルを削除し，改ざんされていないログファイルからログ情報を自動的に復元する．この処理は原本のログファイルだけでなく，バックアップ用のログファイルにもまったく同様に処理が行われる．
つまりバックアップ用のログファイルと原本用ログファイルの処理上の違いは，ログファイルが隠蔽されていて，定期的に再隠蔽されるか否かの違いである．

(4)　ログファイルの再隠蔽
これは改ざん検知とは関係なく，隠蔽されているバックアップ用ログファイルを，定期的に現在とは異なるディレクトリに再隠蔽する．またこの処理により隠蔽ディレクトリの変更だけでなく，ログファイルのファイル名も変更される．なおこの処理は，ログファイルの改ざんが検知された時にも強制的に実行される．

試作したシステムは，現在SGI社のIRIX 6.5とSun Microsystems社のSolaris 2.6上で動作を確認している．今後，Linuxへの移植も行う予定である．

## 4. 考　　察

本章では，本システムの利点と欠点について考察する．また従来のログ情報保護手法との比較も行う．

### 4.1 利　　点

本システムにおける利点は以下の通りである．

- ログ情報の改ざんを検知可能
本システムでは，改ざん検知処理がログファイルを監視しているため，ログ情報の改ざんを検知可能である．
これにより，システム管理者は秒単位の間隔でログファイルに対する不正行為を知ることが可能になり，不正行為に対する迅速な対応を行うことが可能になる．またこの事象の発生を契機として，ログ情報保護のためのいくつかの定型処理を自動的に処理することも可能である．その一例としては，改ざんの検知を契機として現在の全ログ情報を紙へ印刷するということが考えられる．
- 改ざんされたログ情報を復元可能
本システムはログ情報を実時間でバックアップしている．このためログ情報が改ざんされた際には，改ざん前のログ情報に復元することが可能になる．これにより，たとえログファイル自身が，不正侵入者に削除されたとしても，削除前のログ情報を復元することが可能になる．
- 付加的なハードウェアは必要ない
本手法は，安全性を保証した計算機やWrite-Onceメディア等の付加的なハードウェアを必要としない．そのため通信上の安全性を考慮する必要はなく，またメディアのメンテナンスの手間やそれに対する人的誤りも発生しない．
- 任意のアプリケーションに組み込み可能
本システムはC++のクラスとして試作された．これによりC，C++で書かれている種々のアプリケーションに対して容易に組み込み可能である．現在の実装においてログ情報を出力するアプリケーションに対するインタフェース関数は，初期化関数と，ログ情報出力関数の二種類のみである．また，本手法の処理はログ情報の内容には依存しない．そのため，ログ情報の暗号化や圧縮等といった追加機能の実現も，アプリケーション側で実現するか，本クラスを継承したクラスを作成し拡張することで容易に実装可能である．
- 多くのUNIXプラットフォームで動作可能
本システムは，種々のUNIX系OSで動作することを目指している．現在の実装における動作条件は，C++の処理系とStandard Template library(STL)[5]，それにpthread libraryが使用可能であることである．pthread libraryはPOSIX標準であり，STLもC++の標準的なクラスライブラリとなりつつある☆ので，より多くのUNIX系OSの計算機で稼働可能である．

上記の利点から，本手法を用いることで比較的簡単

☆ 詳細はhttp://www.stlport.org/doc/platforms.html参照

にログ情報の改ざんや削除を困難にすることが可能である．不正侵入検知においてログ情報に必要とされる要件の一つとして，不正侵入された際，不正侵入により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ということが挙げられる[1]．この条件が満たされてはじめて，不正侵入検知作業においてログ情報を監視/分析することによる不正侵入検知が可能になるからである．本手法では，改ざん困難であること，また，たとえログ情報が削除されたとしてもその復元が可能であることにより上記条件を満たすことが可能になる．

### 4.2 本システムの限界

本節では，本システムのおける問題点/限界について述べる．

- ログ情報が復元不可能になる可能性がある
  本手法においても，ログ情報が復元不可能になる可能性はある．それは全てのログファイルがほぼ同時☆に削除された場合である．この状況は，ログ情報が復元不可能になると考えられる場面の一つである．
  本手法では，ファイルシステム内にバックアップ用ログファイルを作成し，それらを隠蔽することでログファイルに対する保護を行っている．しかしながら，全てのバックアップ用ログファイルを発見することは可能である．しかし，それを遂行するためにはファイルシステム内の全ファイルを原本のログファイルと同一か確認し，かつ全ファイルシステムを確認中に，発見したログファイルが他のディレクトリに再隠蔽されないという二つの条件が満たされなければならない．よってその実現は容易ではないと考える．.
- ログファイルの安全性はファイルシステムの容量に依存
  本手法を用いてもログ情報が復元不可能になる可能性が存在し，それを遂行するためには，ファイルシステム内の全ファイルを調査する必要がある．つまり，本手法のログファイルに対する安全性はファイルシステムの容量に依存する．
  したがってファイルシステムの容量が比較的少ない場合，必然的に隠蔽可能なディレクトリ数は少なくなり，結果として全ログファイルが発見される可能性は高くなる．またアプリケーション実行時のUser ID，Group IDによっては書き込み許可のあるディレクトリが更に少なくなり，問題を悪化させる可能性がある．
- 適用可能なアプリケーションは限定される
  本手法は，UNIXにおけるdaemon処理など，永続的に動作するアプリケーションのロギング機能にしか適用できず．コマンドのように限られた時間のみ動作し，処理を終了するアプリケーションのロギングには適用できない．その理由は，ログ情報が改ざんされているかを定期的に監視する必要があるからである．
  しかしこの問題は，syslogに本手法を適用することによって解決可能である．syslogに本手法を適用し，稼働させることにより，syslogが出力するログ情報は保護される．種々のコマンドから出力されるログ情報は，syslogを利用してロギングすることにより，それらのログ情報は保護可能になる．
- 計算機への負荷は増大する
  定期的にログファイルに対する改ざん検知処理を行うため，またログ情報を複数のログファイルに出力するため計算機への負荷は増大する．しかし，改ざん検知処理を行う間隔はユーザによって設定可能であるため，計算機に対する負荷調整は可能である．またこの問題は，ログファイルに対する重要度やログ情報の出力量にも依存する．
- 隠蔽時のファイル名やディレクトリ選択の方法
  本手法は，バックアップ用ログファイルを任意のディレクトリに任意のファイル名にて隠蔽する．しかしどのように多くのディレクトリからディレクトリを選択し，そしてどのようなファイル名を作成すれば発見されにくいか，その方針は明確でない．またUNIX系OSのファイルシステムに対する知識を利用することにより，より発見されにくい隠蔽方法が存在する可能性もある．また原本のログ情報が存在するディレクトリが削除された場合，ログ情報を復元する場所がなくなるという問題もある．

これらの理由から，本手法がサーバ計算機のログファイルに対象をおいた理由が明らかになる．提案する手法自身には動作環境をサーバ計算機に限定する理由はない．もちろんサーバ計算機でなくても本手法は機能する．しかし，計算機に負荷がかかること，ログ情報に対する安全性はファイルシステム内の隠蔽可能なディレクトリ数に依存することなどの理由により，本手法はCPUやファイルシステムの資源が一般的に豊富である考えられるサーバ計算機に適した方法であるといえる．

### 4.3 従来のログ情報保護手法との比較

ログ保護手法に関する関連研究や従来の方法と本手法との差異について比較考察する．

- 暗号による手法
  暗号によるログ情報を保護するシステムはいくつか存在する[1)3)]．しかし，これらのシステムはログ情報の改ざんを困難にするだけであり，無差別改ざんやログ情報の削除に関しては保護不可能である．また，ログ情報の改ざん/削除を検知する

☆ ログファイルの削除に数秒程度の時差がある場合，削除されなかったログファイルからログ情報を復元可能である．ただし，これはバックアップログファイル数や改ざん監視間隔に依存する

によって不正侵入は検知できるので十分という考えも存在するが，不正侵入が成功した原因やそれによる影響範囲の調査，犯人の特定を行うための情報が失われてしまうため，それは決して望ましいことではない．

- Write-Once メディアへの保存
  Write-Once メディアを使用したログ情報保護手法は，一度メディアに書き込まれた情報は決して改ざん/削除できないという意味で最も安全な手法である．しかしこの手法にも問題がある．その一つは，どのような契機でログ情報をメディアに保存するかという問題である．考えられる方法としては，一定期間内に出力されたログ情報を一括して定期的に保存する方法と，ログ情報が出力される度にそれを保存する方法である．
  前者の方法は，ログ情報の改ざんとは無関係に保存されるため，ログ情報が改ざんされたとしても，改ざんされたログ情報を記録してしまい，ログ情報を保護できないことは明確である．一方後者は，ログ情報を完全に保護可能だが，その機能は保護のみであり，改ざん検知については別の仕組みを導入しなくてはならない．またどちらの方法においてもメディアのメンテナンスを行わなければならず，手間がかかると同時に人的誤りを誘発する可能性もある．
  我々は，ログ情報の改ざんを検知したというイベントを Write-Once メディアへログ情報を保存する契機に反映することが有効であると考える．しかし通常のログ情報では，なんらかの方法で改ざんを検知したとしても，その時にはすでにログ情報が失なわれているため，Write-Once メディアに保存してもログ情報は保護不可能である．
  しかし，本手法を利用することで改ざんを検知でき，かつ，改ざん後も失われたログ情報を復元することができるため，Write-Once メディアを有効に使用することが可能になる．
- 安全性が保証された計算機へのログ情報の移送
  本方法は，生成されたログ情報を安全性の保証された計算機へ移送することでログ情報を保護する手法である．しかしこの手法は，ログ生成計算機とログ保護計算機間の通信の安全性を保証しなければならない．
  この安全性を保証する一方法として転送するログ情報を暗号化する方法がある．しかし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は，たちまちログ情報が欠落してしまう恐れがある．

## 5. おわりに

本論文では，不正侵入検知においてログ情報は重要な情報源であり，保護すべきものであることを述べた上で，ログ情報の削除まで考慮にいれたログ情報保護手法「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．

- ログ情報の複数バックアップ
- ログ情報の改ざん検知
- バックアップ用ログファイルの隠蔽
- 改ざん時のログ情報の自動復元

なお本手法では，ログ情報の改ざん検知を契機として付加的な処理を自動的に行うことが可能である．またログ情報を保護するために付加的なメディアやログ情報を生成する計算機と別の計算機を必要としないため，比較的容易に導入可能である．

これによりログ情報は保護され，ログ情報を改ざん/削除することにより不正侵入者が自身の痕跡を消すことは困難になる．したがって定期的にログ情報を調査/解析することでより確実に不正侵入を検知することが可能になる．

しかし本手法であってもログ情報を完全に保護可能なわけではない．けれども今後，不正侵入の増加とともにログ情報はますますその重要性を増すと想定され，ログ情報保護の必要性も高まると考える．我々は，そのような状況を想定した上で，ログ情報保護として有効だと考える一方法を提案した．

## 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, 1998, pp. 53-62
2) Gene H. Kim, Eugene 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) Bradford Nichols, Dick Buttlar and Jacqueline Proulx Farrell: Pthread Programming A POSIX Standard for Better Multiprocessing, O'Reilly, 1996
5) D.R. Musser, Atul Saini, Alexander Stepanov; STL Tutorial and Reference Guide - C++ programming with the standard template library, Addison-Wesley Publishing, 1996
6) R.L. Rivest, RFC1321: The MD5 Message-Digest Algorithm, MIT Laboratory for Computer Science and RSA Data Security, Inc., 1992