ログファイル分散管理方式の検討

江藤　文治†　高橋　健一†　堀　良彰‡　櫻井　幸一†‡

†（財）九州先端科学技術研究所 〒814-0001 福岡市早良区百道浜2-1-22

‡九州大学大学院システム情報科学研究院 〒819-0395 福岡市西区本岡744番地

E-mail: †{f-etoh, takahashi, sakurai}@isit.or.jp, ‡hori@csce.kyushu-u.ac.jp

あらまし 近年、デジタル・フォレンジックへの関心の高まりから、デジタルデータに対する破壊、改竄及び訴訟時の証跡情報となるログの重要性も高まっている。攻撃者にとって不利な証跡となるログ情報は改竄や削除、ログファイル自体の削除等の危険性がある。よって、攻撃者による改竄や削除を防ぎ、発生時のままのログ情報やログファイルを保証するログ保全の仕組みが必要となる。本稿では、ログ情報の内容と冗長性を保証する為、ログファイル格納前にログ情報を複数ログファイルに分散保存する方式を提案する。分散時に該ログ情報のハッシュ値と共に保存することにより、発生順序性を含んでログ内容を保全し、改竄や削除への耐性を提供する。

キーワード デジタル・フォレンジック，ログファイル，ファイル分散，ハッシュ値

A study of the log file dispersion management method

Fumiharu Etoh†　Kenichi Takahashi†　Yoshiaki Hori‡　Kouichi Sakurai†‡

†Institute of Systems, Information Technologies & Nanotechnologies

2-1-22 Momochihama, Sawara-ku, Fukuoka, 814-0001, Japan.

‡Department of Computer Science and Communication Engineering, Kyushu University, 744 Motooka Nishi-ku Fukuoka 819-0395, Japan.

Abstract Digtal Forensics has been paid attention recently, so that, the importance of the log, which are evidence information for damage, manipulation and lawsuit, about digital data has been rising. Because log information are disadvantageous for attackers, log information has dangerousness such as manipulation and deletion or the deletion of log file in itself. Therefore, structure of the log maintenance to prevent manipulation and deletion by the assailant and to guarantee log information and log file s at the time of the outbreak is necessary.

In this paper, we suggest a method that disperse and store to log information plural log files before log file storage to guarantee redundancy of the log information. This method includes the order characteristics by saving it with a hash value of the log information when dispersing, guarantees log contents, and make it possible to prevent manipulation and deletion.

Keyword Digital Forensics, log file, file distribution, hash value

1. はじめに

近年、デジタル・フォレンジック(コンピュータ・フォレンジック、以降 DF)の重要性が高まっている。背景には、企業活動や日常生活におけるデジタル化の進展と刑事及び民事訴訟の増大がある。デジタルデータに対する破壊や改竄、漏洩の調査情報、及び訴訟時の証跡情報として用いられるのがログである。ログはコンピュータネットワークの証跡情報の総称であり、事象に関する「発生時間」、「種類」、「処理結果」、「識別情報(id, IP アドレス他)」等の情報が記録される[1]。コンピュータシステムやネットワークシステム(以下、システム)においては、システムを構成する PC/WS、ルータやサーバ等の各機器において、機器の動作状況や操作状況の履歴がシステムログ(syslog)やコマンドログ等のログ情報としてその種類毎のログファイルに保存される。システムの利用者や管理者は、各種ログファイルの内容を参照することにより、システムを構成する通信機器

の動作状況や機器の障害発生の有無、その詳細な内容をリアルタイム及び過去に遡って確認することが可能となる。また、最近は、システムにおける技術的な記録としてだけでなく、悪意のある外部または内部からの攻撃者や不正アクセス者(以下､攻撃者)によるセキュリティインシデントの検出や被害内容の解析、及び、不正なシステム利用やシステム操作の事実を立証する為の証拠としても利用されている。一方で、攻撃者により、彼らにとって不利な証跡となる攻撃に関連するログ情報の改竄や削除、又はログファイル自体が削除される危険性が増している。従って、攻撃者による改竄や削除を防ぎ、発生時の記録のままのログ情報やログファイルを保管するログ保全の仕組みが必要となる。

ログファイルは、通信機器のハードディスクやRAM(Random Access Memory)に保存､または､syslogのように通信プロトコルを用いてログ専用サーバへ送信され、該ログ専用サーバのハードディスク等に保存される。ハードディスク上のファイルに対するセキュアな仕組みも考えられているが、ファイル内容の改竄やファイル自体が削除された場合、セキュリティインシデントの検知やログファイルの復旧は困難であり、ログ専用サーバへ送信する前に改竄や削除される可能性が常に存在する。

そこで本稿では、通信機器における各種ログ情報の発生時に、該ログ情報のハッシュ値を一方向性ハッシュ関数により計算し、該ログ情報と計算したハッシュ値を一組にし、かつ冗長性を持たせて複数のログファイルに分散して保存するログファイル分散方式によるログ保全を提案する｡通常、発生したログ情報はその種類により、該当するログファイルに蓄積され、一定周期毎や指定ファイルサイズに達すると圧縮やファイル分散等の処理を実施の上で任意の格納領域に保管される。暗号化等の処理が実施される場合もあるが、ログファイルとして存在中にファイル内容の改竄や削除等の攻撃を受ける可能性がある。よって、本提案では、存在するログファイルを暗号化や分散保存するのではなく、ログファイルとして存在する前、各ログ情報の発生時に後述のハッシュ値計算処理とログファイルへの分散を実施する。ハッシュ値の利用により、分散後のログファイルにおけるログ情報の改竄や削除の有無を確認可能となる。また、ログ情報の任意の一組が複数のログファイルに存在することにより冗長性を実現し、任意の分散先ログファイルにおいてはログファイル全体の一部分しか存在しない為、削除や漏洩時のリスク低減が可能となる。また、ログ情報の単位はその性質や情報量により複数案が考えられるが、本稿ではログファイル内の一行(一情報)と定義する。

2. 関連研究

DFは、管理されたネットワーク環境下で利用されるネットワーク型と、ネットワークとは独立した環境下で利用されるスタンドアロン型に分類される｡スタンドアロン型のログ保全方式としては、セキュリティデバイスとヒステリシス署名を用いた方式が提案されている[2]｡セキュリティデバイスを装着した機器で操作者を特定するPCのような機器での運用が想定されており、ルータ等のネットワーク型の通信機器は対象ではない。また、ヒステリシス署名は過去から現在までの連鎖的な署名の効果を提供するが、仮に一度でも署名検証で不一致の場合は過去のデータは検証されすログ情報が保証されない。ネットワーク型のログ保全方式としては、信頼できる第三者機関(TTP: Trusted Third Party)を設けてログを安全に保管する方式[3]や複数エンティティが参加する場合に各ログエントリーの時刻情報を考慮した順序関係保証方法[4]が提案されている｡第三者機関にてログを安全に保管する方式はJ-SOX法等を背景にタイムスタンプサービス等が提供されているが、組織内の全ての通信機器のログを保管対象とするにはデータ容量的にコスト上の問題やトラフィックの増加、セキュリティ上の懸念がある。また、DFを目的としたファイル分散保存システムの提案[5]もあるが、一般のログファイルを想定しており、既に存在するログファイルの場合、ファイル分散保存の前に改竄や削除される危険性がある。

3. 提案方式

3.1. 適用ネットワーク

提案方式を以下に示す。

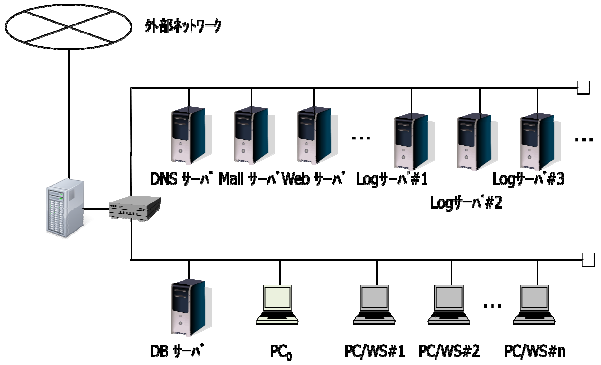

図 1 提案方式適用のネットワーク構成(例)

図 1 は提案方式を適用するネットワーク(以下、NW)構成(例)である。NW は、DNS サーバ,Mail サーバ,Web サーバ,DB サーバのような複数の通信機器及び PC や WS によって構成される。各通信機器は各種ログ情報を生成する。また、そのログ情報を分散して保存する機能を有する専用ログサーバ(以下、ログサーバ)として、複数台の Log サーバ#n (n≧1 の自然数)が配置される。但し、記憶容量及び処理性能等において余力を有する任意サーバが存在すれば、ログサーバの機能を兼務しても良い。図 1 では複数台のログサーバを同一セグメント上に配置しているが、LAN 上の別セグメントへ配置してもよい。PC_0 は、複数のログサーバに分散保存されたログファイル情報から前述の通信機器で検出された全ログ情報で本来作成されるログファイル復元する端末(以下、復元端末)である。

図 2 に各通信機器、ログサーバ、及び復元端末に予め格納される情報を示す。各通信機器は各種ログ情報を分散処理する際に使用する任意かつ共通の鍵付き一方向性ハッシュ関数 H[6][7]と各通信機器固有の鍵情報 K_x (x は通信機器種別)を通信機器内の耐タンパー性の領域に格納する。耐タンパー領域に格納する理由は、ハッシュ関数 H 及び各固有鍵情報 K_x の改竄や削除を防止する為である。同様に、各ログサーバ(Log サーバ#n)と復元端末 PC_0 は、上述のハッシュ関数 H と全通信機器

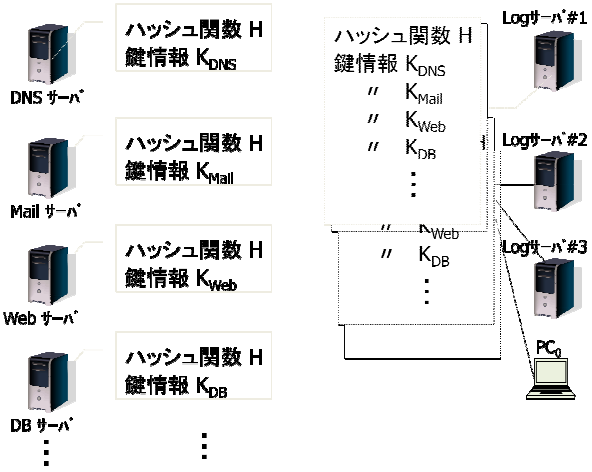

図 2 通信機器他の耐タンパー領域の情報

の各固有鍵情報 K_x を自身の耐タンパー性の領域に格納する。各通信機器固有の鍵情報の受渡しは、事前に実施されることを前提条件とする。

3.2. ログ情報の分散

Mail サーバ上のログ情報を例に、ログ情報分散と保存について説明する。

```
Jul 28 00:00:05 mx postfix/smtpd[84600]: table btree:/usr/local/etc/dracd has changed – restarting
Jul 28 00:00:05 mx postfix/smtpd[84623]: connect from sougouweb.isee.*****.**.jp[133.5.38.254]
Jul 28 00:00:06 mx postfix/smtpd[84623]: 26C3BA937: client=sougouweb.isee.*****.**.jp[133.5.38.254]
Jul 28 00:00:06 mx postfix/cleanup[84583]: 26C3BA937: message-id=<fxnbsdxh.hlbcku.gmllgl@kj.*****.**.jp>
Jul 28 00:00:06 mx postfix/qmgr[63445]: 26C3BA937: from=<owner-ait-shokuin@sougou.isee.*****.**.jp>,
size=3412, nrcpt=1 (queue active)
Jul 28 00:00:06 mx postfix/qmgr[63445]: 26C3BA937: removed
Jul 28 00:00:06 mx postfix/smtpd[84623]: disconnect from sougouweb.isee.*****.**.jp[133.5.38.254]
Jul 28 00:00:28 mx postfix/smtpd[84623]: table btree:/usr/local/etc/dracd has changed – restarting
Jul 28 00:00:30 mx postfix/smtpd[84628]: connect from unknown[94.96.1.186]
Jul 28 00:00:34 mx postfix/smtpd[84629]: connect from unknown[113.250.209.148]
Jul 28 00:00:37 mx postfix/smtpd[84630]: connect from unknown[190.206.205.51]
Jul 28 00:01:00 mx postfix/qmgr[63445]: 7E502AB88: from=<>, size=9848, nrcpt=1 (queue active)
Jul 28 00:01:00 mx postfix/qmgr[63445]: 361F1ABC6: from=<>, size=8169, nrcpt=1 (queue active)
Jul 28 00:01:00 mx postfix/qmgr[63445]: 87A5FABC0: from=<>, size=6445, nrcpt=1 (queue active)
Jul 28 00:01:30 mx postfix/smtp[84635]: connect to www.*****.**.jp[202.214.210.50]: Operation timed out (port 25)
Jul 28 00:01:30 mx postfix/smtp[84636]: connect to www.*****.**.jp[202.214.210.50]: Operation timed out (port 25)
```

図 2 Mail サーバの syslog の情報(例)

図 2 は Mail サーバ上で本来作成されるログファイル(例)である。このログファイルを Log サーバ#1,Log サーバ#2,Log サーバ#3 の 3 つのログサーバに分散保存する。Mail サーバにおけるログ情報生成時、ログ情報に対し、図 4 の”ログ分散処理”フローを実施する。まず、カウンタ i(初期値=1)を該当ログ情報 log_i の先頭に付与し、Log_i とする。

$$Log_i = “i” + “log_i”$$

但し、これらの i, log_i, Log_i は、後述のログサーバへの送信されるまで、通信機器上で改竄や削除の攻撃を受けないことを前提条件とする。

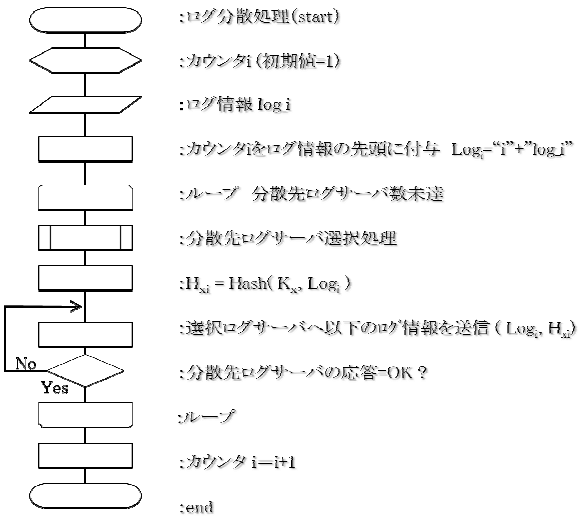

図 4 ログ分散の処理フロー

次に、分散先ログサーバ選択処理により複数の分散先ログサーバを選択する。簡易な例として、以下のように3ログサーバより2ログサーバをログ情報の分散先として選択する処理を用いる。

分散先ログサーバ選択処理：
i: i≧1の自然数、
N:分散先サーバ数 (N=3, サーバ#1,#2,#3)
i mod N-1 =1の場合→ サーバ#1,#2
i mod N-1 =2の場合→ サーバ#1,#3
i mod N-1 =0の場合→ サーバ#2,#3

次に、耐タンパー領域において、上述の Log_i と予め保管する自サーバの鍵情報 K_x(例:K_{mail})とハッシュ関数Hから、ハッシュ値 H_{xi} を計算する。

$$H_{xi} = Hash(K_x, Log_i)$$

このハッシュ値 H_{xi} は、後述のように、ログサーバにおけるログ保存処理(図 5)と復元端末におけるログファイル復元処理(図 7,8)において、ログ情報の正常性、即ち、改竄が実施されていないことの検証に用いる。ハッシュ値を計算後、ログ情報(Log_i, H_{xi})を一組にし、選択した分散先ログサーバへ送信する。分散先ログサーバからの応答がNGならば再送する。OKならば分散先ログサーバ選択処理から同サーバからの応答確認を、分散先ログサーバ数だけ実施し、次カウンタ i を処理する。

任意のログ情報を複数のログサーバに分散して保存することにより、ログ情報の冗長性を保証する。仮に、任意のログサーバ上のログ情報が改竄や削除等の攻撃により失われた場合は、別のログサーバの該当ログ情報から復元可能である。分散先のログファイル数が多いほど冗長性は強化される。一方、ログ記憶容量や処理能力が増加する為、コストや処理時間との、トレードオフになる。

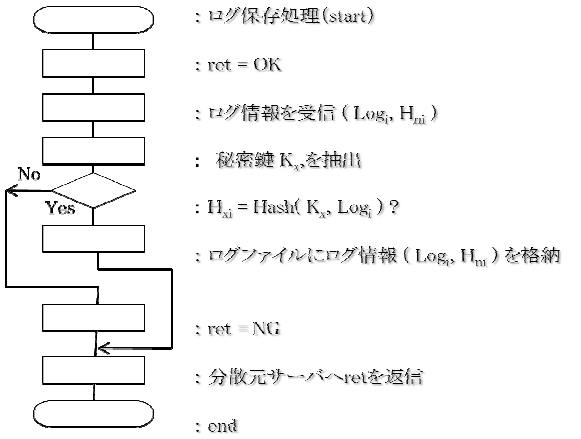

図 5 ログ保存の処理フロー

ログサーバにおけるログ保存処理について述べる。ログサーバは受信したログ情報に対し、図 5の"ログ保存処理"フローを実施する。耐タンパー領域において、受信ログ情報 Log_i と予め保管する送信元の通信機器の鍵情報 K_x とハッシュ関数Hから、ハッシュ値 H_x を計算する。

$$H_x = Hash(K_x, Log_i) \qquad \cdots(1)$$

計算したハッシュ値 H_x と受信ハッシュ値 H_{xi} を比較する。一致すれば正常と判断し、送信元へ OKを送信し、ログ情報(Log_i, H_{xi})を該当ログファイルに格納する。不一致ならば非正常と判断し、送信元へNGを送信し再送を要求し、正常化を図る。

1,H_{x1},H_{p1},Log_1="Jul 28 00:00:05 mx postfix/smtpd[84600]: table btree:/usr/local/etc/dracl has changed – restarting"
2,H_{x2},H_{p2},Log_2="Jul 28 00:00:05 mx postfix/smtpd[84623]: connect from sougouweb.isce.*****.**.jp[133.5.38.254]"
4,H_{x4},H_{p4},Log_4="Jul 28 00:00:06 mx postfix/cleanup[84583]: 26C3BA937: message-id=fmbsdxh.hlbcku.gnilgt@kj.*****.**.jp"
5,H_{x5},H_{p5},Log_5="Jul 28 00:00:06 mx postfix/qmgr[63445]: 26C3BA937: from=<owner-ait-shokuin@sougou.isce.*****.**.jp>, size=3412, nrcpt=1 (queue active)"
7,H_{x7},H_{p7},Log_7="Jul 28 00:00:06 mx postfix/smtpd[84623]: disconnect from sougouweb.isce.*****.**.jp[133.5.38.254]"
8,H_{x8},H_{p8},Log_8="Jul 28 00:00:28 mx postfix/smtpd[84623]: table btree:/usr/local/etc/dracl has changed – restarting"
10,H_{x10},H_{p10},Log_{10}="Jul 28 00:00:34 mx postfix/smtpd[84629]: connect from unknown[113.250.209.148]"
:

図 6.1 サーバ#1のログファイル情報

1,H_{x1},H_{p1},Log_1="Jul 28 00:00:05 mx postfix/smtpd[84600]: table btree:/usr/local/etc/dracl has changed – restarting"
3,H_{x3},H_{p3},Log_3="Jul 28 00:00:06 mx postfix/smtpd[84623]: 26C3BA937: client=sougouweb.isce.*****.**.jp[133.5.38.254]"
4,H_{x4},H_{p4},Log_4="Jul 28 00:00:06 mx postfix/cleanup[84583]: 26C3BA937: message-id=fmbsdxh.hlbcku.gnilgt@kj.*****.**.jp"
6,H_{x6},H_{p6},Log_6="Jul 28 00:00:06 mx postfix/qmgr[63445]: 26C3BA937: removed"
7,H_{x7},H_{p7},Log_7="Jul 28 00:00:06 mx postfix/smtpd[84623]: disconnect from sougouweb.isce.*****.**.jp[133.5.38.254]"
9,H_{x9},H_{p9},Log_9="Jul 28 00:00:30 mx postfix/smtpd[84628]: connect from unknown[94.96.1.196]"
10,H_{x10},H_{p10},Log_{10}="Jul 28 00:00:34 mx postfix/smtpd[84629]: connect from unknown[113.250.209.148]"
:

図 6.2 サーバ#2のログファイル情報

2,H_{x2},H_{p2},Log_2="Jul 28 00:00:05 mx postfix/smtpd[84623]: connect from sougouweb.isec.*****.**.jp[133.5.38.254]"
3,H_{x3},H_{p3},Log_3="Jul 28 00:00:06 mx postfix/smtpd[84623]: 26C3BA937: client=sougouweb.isec.*****.**.jp[133.5.38.254]"
5,H_{x5},H_{p5},Log_5="Jul 28 00:00:06 mx postfix/qmgr[63445]: 26C3BA937: from=<owner-uit-shokuin@sougou.isec.*****.**.jp>, size=3412, nrcpt=1 (queue active)"
6,H_{x6},H_{p6},Log_6="Jul 28 00:00:06 mx postfix/qmgr[63445]: 26C3BA937: removed"
8,H_{x8},H_{p8},Log_8="Jul 28 00:00:28 mx postfix/smtpd[84623]: table btree:/usr/local/ete/dracl has changed – restarting"
9,H_{x9},H_{p9},Log_9="Jul 28 00:00:30 mx postfix/smtpd[84628]: connect from unknown[94.96.1.186]"
11,H_{x11},H_{p11},Log11="Jul 28 00:00:37 mx postfix/smtpd[84630]: connect from unknown[190.206.205.51]"
:

図 6.3　サーバ#3 のログファイル情報

(1)を満たすログ情報だけを保存し、満たさない場合は保存しない。これにより、保存時のログ情報の正常性を保証する。よって、復元処理においてハッシュ値不一致の場合、改竄を検知可能となる。分散先の各ログサーバで上記を実施することにより、図 6.1,6.2,6.3 のように、複数のログサーバ上に、ログ情報が分散して保存される。

3.3. 分散ログファイルからの復元

Log サーバ#1,#2,#3 のような複数のログサーバに分散された各ログファイルから、本来作成されるログファイルを復元する。ここでいう復元とは、復元端末において、各ログサーバの分散ログファイルを用いて、通信機器(例:Mail サーバ)で本来作成されるログファイルを再構成することである。今、同 NW に接続される端末 PC_0 で復元を実施する。PC_0 は前述のハッシュ関数 H と全通信機器の各固有鍵情報 K_x を自身の耐タンパー領域に格納している。また、PC_0 は対象ログファイルの分散先ログサーバを認識していることを前提条件とする。

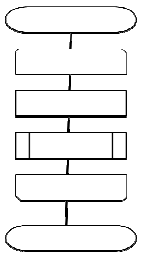

図 7 ログファイル復元処理フロー

ログファイルの復元は図 7 及び図 8 のフロー処理により実施する。図 7 の”ログファイル復元処理フロー”において、分散先ログサーバの一つを指定して、図 8 の”ログ情報復元処理フロー”を動する。耐タンパー領域において、復元する通信機器の鍵情報 K_x を抽出する。分散ログファイルより、カウンタ i(初期値=1)に該当するログ情報 Log_i とハッシュ値 H_{xi} を抽出し、耐タンパー領域において、ログ情報 Log_i と上述の鍵情報 K_x とハッシュ関数 H から、ハッシュ値 H_x を計算する。

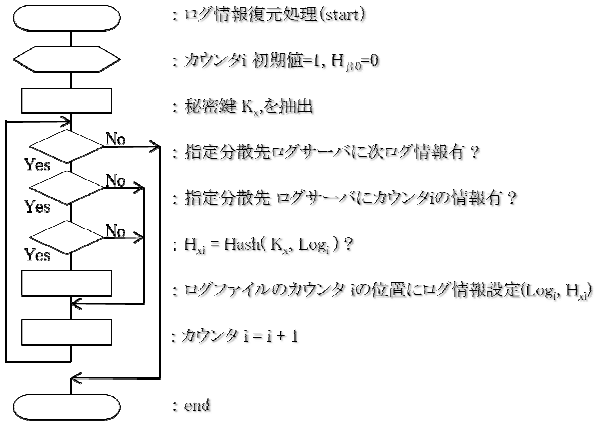

図 8 ログ情報復元処理フロー

$$H_x = Hash(K_x, Log_i)$$

計算ハッシュ値 H_x と分散ログファイルより抽出のハッシュ値 H_{xi} を比較する。一致すれば正常と判断し、復元ログファイルへ追加する。不一致ならば非正常と判断し、復元ログファイルへは追加しない（他の分散ログファイルより復元する)。

$$H_{xi} = Hash(K_x, Log_i) \qquad \cdots(2)$$

を満たすログ情報だけで復元する。分散先ログサーバの分散ログファイルの全ログ情報に対し上記の処理を実施する。さらに、カウンタ i で特定される未復元のログ情報に対し、各ログサーバの分散ログファイルで上記を実施することにより、生成時と同じ正常なログ情報だけにより、本来作成されるログファイルが復元される。

Log サーバ#1,#2,#3 の各分散ログファイルからの復元を示す。但し、各分散ログファイルにおける改竄や削除はないとする。まず、”ログファイル復元処理”を起動し、Log サーバ#1 を選択して”ログ情報復元処理”を起動する。図 6.1 のログファイル情報より、カウンタ 1,2,4,5,7,810,…のログ情報を復元する。さらに、Log サーバ#3 を選択して、同様に復元処理し、図 6.3 のログファイル情報より、カウンタ 3,6,9,…のログ情報を復元する。以上の処理により、図 2 に示す Mail サーバ上で本来作成される syslog ログファイルが復元される。

4. 提案方式における考察

4.1. 分散ログファイルからの復元

3.3 項に記述のように、その構成するログ情報の改竄や削除が無い場合、複数の分散ログファイルにより､元のログファイルの復元が可能である。3.2 項の分散先ログサーバの組合せのそれぞれで、元のログファイルの復元は可能である。

4.2. 分散ログファイルの改竄の検知と耐性

分散ログファイルにおいて、その構成するログ情報の一部、又は、全体が改竄された場合は、復元時に、ログ情報 Log_i と上述の鍵情報 K_x とハッシュ関数 H から、ハッシュ値 H_x を計算すると、

$H_{xi} \neq Hash(K_x, Log_i)$

となり、改竄の事実を検知可能である。これは、ハッシュ関数の特徴と該関数が耐タンパー領域に格納されていることから、Log_i と H_{xi} の改竄値を一致させることが容易ではないことによる。

また、改竄を検知した該当カウンタ i のログ情報については、同じログ情報を複数のログファイルに分散させた冗長性により復元可能である。即ち、他の分散ログファイルのログ情報(Log_i, H_{xi})を基に､ハッシュ値を比較する前述の方法により、改竄が無いことを確認すれば復元が可能である。

4.3. 分散ログファイルの削除の検知と耐性

3.3 項に記述のように､復元端末 PC_0 は対象ログファイルの分散先ログサーバを認識している。よって、分散先ログサーバにおいて、該当する分散ログファイルが存在しなければ、削除を検知可能である。ログファイルにおいて、一部のログ情報が削除された場合は、カウンタ i の該当ログ情報が存在しないことにより､削除を検知可能である。また、削除を検知した該当カウンタ i のログ情報又はログファイルについては、4.2 項と同様に、同じログ情報を複数のログファイルに分散させた冗長性により復元可能である。

4.4. ログファイルの復元の限界と冗長性

4.2,4.3 項で述べたように、分散先のログファイル又はその一部が改竄や削除された場合、ログファイルの分散による冗長性により、他の分散ログファイルに該当ログ情報が存在する限りにおいて、本方式は改竄や削除に対する耐性を有する。しかし、任意のログ情報に関し、全ての分散ログファイルが改竄や削除により冗長性が損なわれた場合、本方式だけによる復元は困難である。

5. まとめ及び今後の課題

本稿では、通信機器のログファイルに対し、可能な限りログ保全することを目的として、通信機器における各種ログ情報の発生時に複数のログファイルに保存するログファイル分散方式を提案した。ログ情報の発生時に、該ログ情報のカウンタ値とログ情報と通信機器固有の鍵情報からハッシュ値を算出し、ログ情報とハッシュ値を複数ログサーバに分散保存する冗長性により、ログファイルの改竄や削除が発生した場合の検知、及び、本来作成されるログファイルの復元方法を示した。

今後は、分散処理やハッシュ値処理に関する効率や有効性についての検証、及び、プロトタイプにおける実装評価を行う予定である.

参考文献

[1] 辻井重男監修：デジタル・フォンレンジック事典,デジタル・フォレンジック研究会, 2006.12

[2] 芦野佑樹,佐々木良一：セキュリティデバイスとヒステリシス署名を用いたデジタルフォレンジックシステムの提案と評価, 情報処理学会論文誌, Feb.2008 Vol.49 No.2 p999-1009

[3] B. Schneier and J. Kelsey:“Cryptographic Support for Secure Logs on Untrusted Machine,” In Proc. of the 7th USENIX Security Symposium, pp.53-62, Jan. 1998.

[4] 安東学,松浦幹太,馬場章：分散環境で保存されるログファイルにおける各ログエントリー間の順序関係保証方法に関する考察,CSS2002

[5] 東森ひろこ, 手塚伸, 宇田隆哉：デジタルフォレンンジックを目的としたファイル分散保存システムの提案,CSS 2008

[6] hash function（ハッシュ関数） http://www.ipa.go.jp/security/rfc/RFC2828-03HJA.html

[7] 暗号理論入門 J.A.ﾌﾞｰﾌﾏﾝ[ｼｭﾌﾟﾘﾝｶﾞ･ｼﾞｬﾊﾟﾝ]