

属性指向帰納によるネットワークログの特徴抽出と異常検知

山田 明[†] 三宅 優[†]
竹森 敬祐[†] 田中 俊昭[†]

ネットワーク監視には、連続的に出力される膨大なネットワークログから自動的に特徴を抽出し迅速に異常を検知するという課題がある。本論文では膨大なデータベースを要約することで特徴を抽出する属性指向帰納 (AOI: Attribute Oriented Induction) を応用し、侵入検知システムなどのネットワークログを解析する方式を提案する。提案方式はログの特徴から AOI に必要な概念階層と呼ばれる階層構造を生成し、単位時間ごとに AOI アルゴリズムを適用することにより要約を出力する。そして、この要約を過去の要約と比較することにより異常を検知する。実環境において収集したネットワークログを用いて、提案方式の異常検知能力、処理速度を評価した。その結果、提案方式は、埋もれがちな小さな異常も検知でき、高速で処理ができることが示された。

Characterization and Anomaly Detection for Network Log Using Attribute Oriented Induction

AKIRA YAMADA,[†] YUTAKA MIYAKE,[†] KEISUKE TAKEMORI[†]
and TOSHIKI TANAKA[†]

At network management, they are important routines that to extract characteristics events and to detect anomalies from daily network log. In this paper, we propose characterization and anomaly detection for network log using attribute-oriented induction (AOI). The proposed scheme composes concept hierarchy, which is required at AOI algorithm adaptively. Therefore our system doesn't need to prepare concept hierarchy based on each network configuration or network services. Using periodic results of AOI, the proposed system detects anomalies, which are lurking behind a volume of network log. We evaluated our system using log, which is collected at actual network, and presented effectiveness of our system.

1. はじめに

ネットワーク監視において、ネットワークログから異常を検知する課題があるが、トラフィック量の増加にともないネットワーク監視機器から膨大なログが出力されるようになり、重要な情報を短時間で発見することが難しくなっている。ネットワーク機器の中では、特に侵入検知システム (IDS: Intrusion Detection System) のログから異常を検知する研究がさかんに行われている。Julishらは膨大なログを要約することにより、それぞれのログの発生原因を発見する方式を提案している^{9),10)}。Julishらの方式は膨大なデータベースを要約する属性指向帰納 (AOI: Attribute Oriented Induction)⁶⁾⁻⁸⁾ を応用することにより特徴を抽出しており、膨大なログから管理者にとって有益な情報を

抽出する。しかし、IDSのログのみを対象としている点、属性指向帰納を適用するときに必要な概念階層と呼ばれる階層構造をあらかじめ与えなければならない点、連続的なログを解析できない点、異常を検知することができない点などの問題がある。特に、概念階層の適応的な構成に関して、Hanらは動的な概念階層の構築法を提案している⁶⁾。しかし、Hanらの方式ははじめに大きな頻度を持つ節点を取り除き、小さな頻度を持つ節点を集約するという方式であり、ある階層に集約するとき下位の階層をすべて集約するため、ディレクトリやIPアドレスのような中間の階層を持つ場合に適用することは困難である。また、ネットワークログのなかで特にIPアドレスの木構造を利用して要約を行う方式がいくつか提案されている^{4),11)}。しかし、IPアドレス以外の値を扱うことができず、解析結果を利用して異常を検知することができない。

そこで、本論文では汎用的なネットワークログを対象とし、ネットワークログにおいて記録されるIPア

[†] 株式会社 KDDI 研究所
KDDI R&D Laboratories Inc.

ドレス、ドメイン名、ディレクトリ名などの構造を利用して概念階層を適応的に構成し、連続的なログ出力に対応できる、AOI を用いたネットワークログ特徴抽出方式を提案する。また、ネットワークログから抽出される特徴的なログを過去のログと比較することにより異常を検知する方式を提案する。提案方式を実装し、ある企業 LAN ネットワークにおけるネットワークログを用いて評価を行う。その結果、提案方式は、連続的で膨大なログから特徴的なログを抽出し、従来検出が困難であった小規模な異常も検知でき、高速に処理が可能であることを示す。

以下、2 章において AOI によるログ要約とその問題について説明し、3 章において一般的なネットワークログに適用する方法を、4 章において概念階層の適応的構成方法を、5 章において高速化について、6 章において異常検知方法をそれぞれ示す。そして、7 章で実装、評価を行い、8 章で考察を行う。最後に 9 章でまとめる。

2. 背景

2.1 ネットワークログ解析の課題

ネットワーク機器は定期的に機器の状態をログとして出力する機能や、機器になんらかの異常が発生したときに警告のログを出力する機能を持つ。ネットワーク管理者は複数の機器から連続的に出力される膨大なネットワークログを監視して、異常が発生したとき迅速に対応しなければならない。特に、異常対応としてネットワークログから適切に特徴を抽出しネットワークの状態を把握しなければならない。また、ネットワークログは時系列順に出力されるため非常に膨大かつ連続的なデータである。したがって、ネットワークログの解析には以下の課題がある。

課題 1 膨大、連続的なログを解析すること。

課題 2 自動的に特徴を抽出すること。

課題 3 異常を検知すること。

ここで、異常とは、過去の状態と比較して発生する確率が低い状態とする。たとえば、通信量の変化が正規分布に従うとき、平均から標準偏差の 3 倍以上離れた値になる確率は 1% 以下となる。したがって、3 倍以上外れた値が繰り返し発生し、1% よりも大きな確率になるとき異常と判断する。このとき、利用者の悪意の有無は関係ないこととする。つまり、正規の利用者が過去と比較して膨大な通信を発生させる場合には通信量における異常と判断される。

2.2 ネットワークログの構造

ネットワークログは発生時刻と 1 つ以上の項目に

よって既定される形式の情報が時系列順に発生する。本論文ではログを行列として表現し、行をある時刻におけるネットワークログとし、列をあるログにおける項目とする。ここで各列における要素の集合を各々 $\{T, A_1, A_2, \dots, A_{n-1}\}$ とするとき、ログにおける各行は集合 $|T| \times |A_1| \times |A_2| \times \dots \times |A_{n-1}|$ におけるベクトルとして表現できる。ここで T は時刻であり、 $|T|, |A_i|$ は要素数である。ネットワークログの項目は概念階層に割り当てることができる場合が多い。概念階層とは狭義の概念を下位に配置し、広義の概念を上位に配置する階層構造である。概念階層 \mathcal{H} は $H^{\ell-1}$ を H^ℓ より広義の概念とすると式 (1) として表現できる。 H^0 は最も広義な概念であり「ANY」と表現できる。

$$\mathcal{H}: H^\ell \Rightarrow H^{\ell-1} \Rightarrow \dots \Rightarrow H^0 = \text{ANY} \quad (1)$$

たとえば、あるネットワークの IP アドレスが図 1 のような関係にあるとき、概念階層は式 (2) のように表現できる。

$$\begin{aligned} H^3 &: \{\text{IP1, IP2, } \dots\} \\ \Rightarrow H^2 &: \{\text{DMZ, NAT, } \dots\} \\ \Rightarrow H^1 &: \{\text{Internal, External}\} \\ \Rightarrow H^0 &: \{\text{ANY}\} \end{aligned} \quad (2)$$

2.3 属性指向帰納による特徴抽出

AOI は関係データベースにおいて膨大なデータベースを要約することで特徴を抽出するアルゴリズムである。このアルゴリズムは列の集合 $\{A_1, A_2, \dots, A_n, C\}$ によって規定される行列 T と各列 A_i の概念階層 \mathcal{H}_i を入力とし、 O 行の要約を出力する。 C は、アルゴリズムによって追加される列であり、行の合計行数を意味する。概念階層を持たない列はアルゴリズム開始前に取り除かれるためすべての列 A_i は概念階層を持つ。行列 T のある行を r 、 r における各列を $r[C], r[A_i]$ とするとき、属性指向帰納アルゴリズムを以下に示す。ただし、目標行数 O 、概念階層、列選択関数は別途与える必要がある。

(1) カウンタの初期化として、すべての行に対して

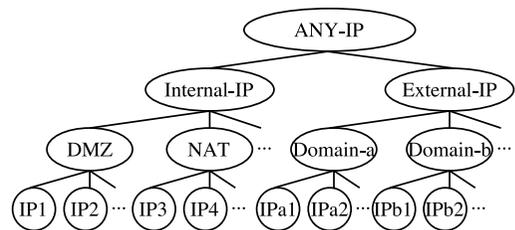


図 1 IP アドレスにおける概念階層の例
Fig. 1 An example of concept hierarchy.

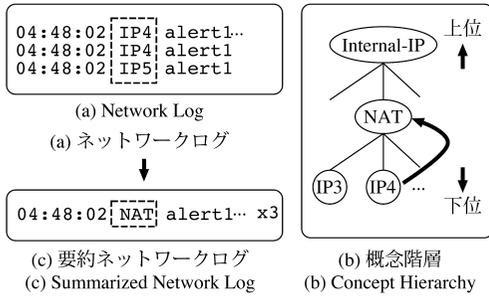


図 2 属性指向帰納によるログ要約
Fig.2 Summarization using AOI.

$r[C] \leftarrow 1$ とする .

- (2) 行列 T の行数が目標行数 O となるまで以下の操作を繰り返す .
- (3) ある列 A_i を選択する .
- (4) すべての行における属性 A_i に対して \mathcal{H}_i における 1 階層広義の階層 H' における値と置き換える .
- (5) 行列 T において同一の行 r と r' が存在する場合カウンタの値を加算し r' を削除する .

$$r[C] \leftarrow r[C] + r'[C]$$

たとえば、図 2 (a) に示す 3 行のログは、IP アドレスの列を選択するとき図 2 (b) の概念階層を構成できるため、図 2 (c) のように要約できる .

2.4 従来方式の問題点

Julisch らは IDS のログにおける IP アドレス、ポート番号、時刻に対して階層構造を定義し属性指向帰納を適用する方式を提案した^(9),10) . この方式は課題 2 のネットワークログからの特徴抽出を実現しているが、IDS のログのみを対象としているため汎用的ではなく、概念階層をあらかじめ定義する必要がある . また、課題 1、課題 3 は実現できていない . したがって、Julisch らの方式は以下の問題点がある .

- 問題点 1 汎用的でない .
- 問題点 2 概念階層をあらかじめ定義する必要がある .
- 問題点 3 膨大で連続的なログを処理できない .
- 問題点 4 異常検知への適用が難しい .

3. 汎用的なネットワークログへの適用

3.1 連続値、離散値、木構造離散値の割当て

一般的にネットワークログにおける各列は連続的に変化する列と離散的に変化する列に分類できる . また、離散的に変化する列は含まれる値がグラフ理論における木に割り当てられる場合が多い . そこで、ネットワークログの各列に連続値、離散値、木構造離散値のいずれかの属性を割り当てることにより様々な形式の

表 1 連続値、離散値、木構造離散値の例

Table 1 An example of sequential, unsequential and tree structural value.

連続値	転送時間、ファイルサイズ
離散値	通信種別、ポート番号、サービス名、フラグ
木構造離散値	IP アドレス、ディレクトリ、ドメイン名、メールアドレス

表 2 xferlog への属性の割当て

Table 2 A assignment attributes to parameters of xferlog.

項目	transfer-time	remote-host	file-size	...
属性	連続値	木構造離散値	連続値	...

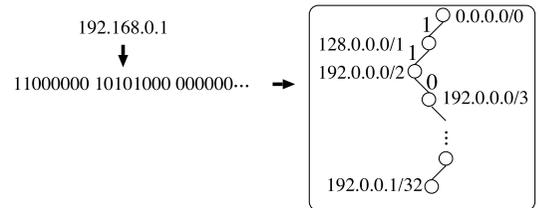


図 3 IP アドレスの 2 進木への割当て
Fig.3 Structure of IP address.

ログを統一的に処理することができる . 連続値、離散値、木構造離散値の例を表 1 に示す . たとえば、転送時間は連続的に変化し、通信種別は離散的に変化し、ドメイン名はドメインツリーとして木に割り当てられる . 表 2 に ftpd のログ形式である xferlog⁽³⁾ の各列に属性を割り当てる例を示す .

3.2 木構造離散値

ネットワークにおけるログの項目として IP アドレス、ディレクトリ、ドメイン名、URL、URI、XPath は離散的に変化し木構造を持つ . つまり、ある項目の値をグラフ理論における木のパスに割り当てることができる . たとえば、IP アドレスは図 3 のように 32 ビット列と見なされ、深さ 32 の 2 進木のパスに割り当てられる . ディレクトリ、ドメイン名も同様である .

また、URL のように、いくつかの組合せにより構成される場合は *hostname* と *path* のように分解することによりそれぞれを木構造に割り当てることができる (図 4) . URI、XPath、e-mail アドレスも URL と同様である .

4. 概念階層の適応的構成

4.1 階層的クラスタリングによる概念階層の構成

AOI は各列を抽象的な概念に置き換えるために概念階層をあらかじめ用意する必要がある . しかし、ネットワークログのすべての列に関してあらかじめ概念階

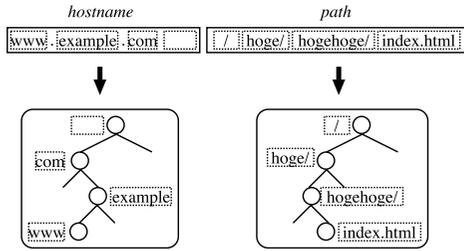


図 4 URL の木構造
Fig. 4 Structure of URL.

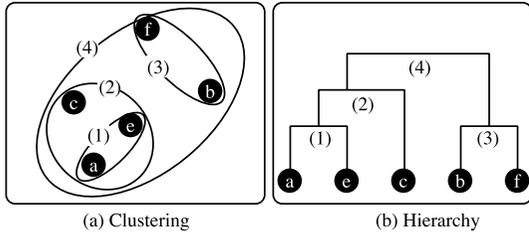


図 5 階層的クラスタリング
Fig. 5 Hierarchical clustering.

層を用意することは困難である．そこで，本論文では階層的クラスタリングによって概念階層を適応的に構成する方法を提案する．クラスタリングとは要素を類似度に基づいてクラスタに分類する方法であり，階層的的手法と非階層的的手法がある．階層的クラスタリングは，はじめに各々の値を異なるクラスタと見なし，クラスタ間の距離を定義して，近いクラスタを1つのクラスタにする手法であり，クラスタリングの過程において階層構造を構成できる．たとえば，図5(a)のような2次元空間上の点をユークリッド距離を用いてクラスタリングするとき図5(b)の階層構造を構成できる．したがって，連続値，離散値，木構造離散値のクラスタリングができれば，階層構造を適応的に構成できる．

4.2 連続値のクラスタリング

ある数値属性 A の要素数が n_A であり，要素とその頻度が $\{\{a_{1,1}|a_{1,2}\}, \{a_{2,1}|a_{2,2}\}, \dots, \{a_{n_A,1}|a_{n_A,2}\}\}$ ， $\{p_1, p_2, \dots, p_{n_A}\}$ であり， $a_{1,1} < a_{2,1} < \dots < a_{n_A,1}$ の大小関係が成り立つとする． $\{a_i|a_j\}$ はその要素が a_i から a_j までの値を含んでいることを意味する ($a_i < a_j$)．このとき，クラスタ間の距離関数として式(3)を定義する．この距離関数によりクラスタリングできる．

$$d(\{a_{j,1}|a_{j,2}\}, \{a_{j+1,1}|a_{j+1,2}\}) = \begin{cases} \frac{p_j + p_{j+1}}{a_{j,2} - a_{j,1}} & (a_{j,2} > a_{j+1,2}) \\ \frac{p_j + p_{j+1}}{a_{j+1,2} - a_{j,1}} & (a_{j+1,2} > a_{j,2}) \end{cases} \quad (3)$$

そして， $\{a_{j,1}|a_{j,2}\}$ と $\{a_{j+1,1}|a_{j+1,2}\}$ を1つのクラスタに集約するとき新しいクラスタ a' およびその頻度 p' は式(4)とする．

$$a' \leftarrow \begin{cases} \{a_{j,1}|a_{j,2}\} & (a_{j,2} > a_{j+1,2}) \\ \{a_{j,1}|a_{j+1,2}\} & (a_{j+1,2} > a_{j,2}) \end{cases} \quad (4)$$

$$p' \leftarrow p_j + p_{j+1}$$

距離関数によりクラスタリングを行い，クラスタ数が $R \times |A|$ となるときの1階層広義の階層とする．ここで， R はあらかじめ与える集約率であり， $0 < R < 1$ を満たす値を選択する．

4.3 離散値のクラスタリング

ある離散属性 A の要素数が n_a であり，要素とその頻度が $\{a_1, a_2, \dots, a_{n_A}\}$ ， $\{p_1, p_2, \dots, p_{n_A}\}$ であり， $p_1 < p_2 < \dots < p_{n_A}$ の関係が成り立つとする．このとき，頻度が小さいから順に ANY クラスタとする．このクラスタリングによりクラスタ数が $R \times |A|$ となるときの1階層広義の階層とする．ここで， R はあらかじめ与える集約率であり， $0 < R < 1$ を満たす値を選択する．

4.4 木構造属性のクラスタリング

ある階層構造属性 A の要素数が n_A であり，要素とその頻度が $\{a_1, a_2, \dots, a_{n_A}\}$ ， $\{p_1, p_2, \dots, p_{n_A}\}$ ($p_1 < p_2 < \dots < p_{n_A}$) であるとき，各要素を木におけるパスに割り当て，パスの終点にあたる節点を $\{n_1, n_2, \dots, n_m\}$ ($m < n_A$) とする．ここであらためて，ある節点 n の頻度を $n[p]$ ，子の節点における累積頻度を $n[o]$ ，子の節点数を $n[c]$ とするとき以下の処理を行う．ある節点の累積頻度とはすべての子における頻度の和である．ここで，パスの終点ではない節点の頻度は $n[p] = 0$ である． S は全節点の累積数とする．つまり， $S = \sum_j^{n_A} p_j$ である．以下の処理によってクラスタリングを行い，クラスタ数が $R \times |A|$ となるときの1階層広義の階層とする．ここで， R はあらかじめ与える集約率であり， $0 < R < 1$ を満たす値を選択する．図6に概要図を示す．

- (1) 要素数が $|A'_i|$ になるまで以下の処理を繰り返す．
- (2) $Th = \frac{S}{|A'_i|}$ を計算する．
- (3) 頻度が0よりも大きい節点において最も頻度が小さな節点 n_{min} を選択する．ここで，節点 n_{min} の親を n'_{min} とする．
- (4) $n'_{min}[o] < Th$ または $n'_{min}[p] > 0$ ならば n_{min} を n'_{min} に集約する．
- (5) (4) の条件を満たさず， $\frac{n'_{min}[o]}{n'_{min}[c]} < Th$ ならば， n'_{min} の子節点の中で最も頻度が小さな節点 n_q

を探索し, n_{\min} と n_q を n'_{\min} に集約する.

- (6) (4), (5) の条件を満たさない場合, 節点 n'_{\min} の親 n''_{\min} という順に親に対して (4), (5), (6) を行う.

4.5 適応的概念階層構成による AOI

本節において適応的に概念階層構成することにより AOI を行う方法を示す. ログの各列の属性を各々 $\{T, A_1, A_2, \dots, A_{n-1}\}$ とし, A_i は数値, 離散値, 木構造離散値のいずれかとする. また, 集約率は R とする. 各々の処理に置いて L 行を入力とし O 行の要約を出力とする. 図 7 に提案方式のブロック線図を示す.

- (1) AOI を実行するためにカウンタ C を追加する.
- (2) A_i ($i = 1, 2, \dots, n - 1$) に対して属性選択関数 $I(A_i)$ を計算する.
- (3) $I(A_i)$ が最も小さな列 A_{\min} を選択する. このとき列に含まれる要素数を $|A_{\min}|$ とする.

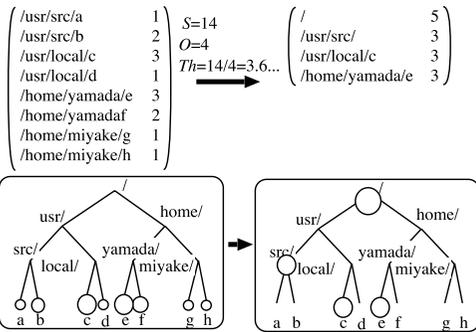


図 6 木構造から階層構築の生成

Fig. 6 Construction of hierarchy from a tree structure.

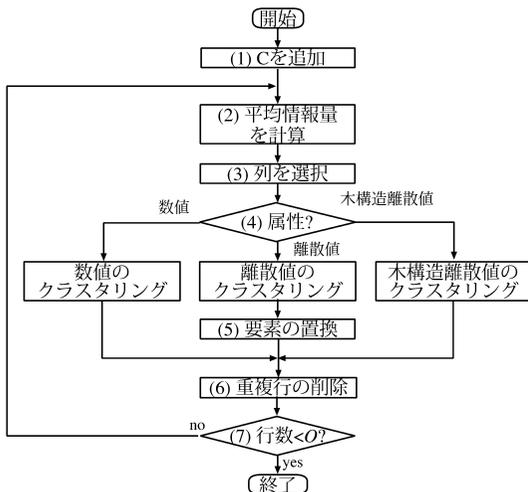


図 7 提案アルゴリズム

Fig. 7 Proposed algorithm.

- (4) 選択される列に対応して, 数値, 離散値, 木構造離散値のいずれかのクラスタリングを行い, 1 段広義の階層を生成する. つまり, A_{\min} に対してクラスタ数が $|A'_{\min}| = |A_{\min}| \times R$ までクラスタリングを行う.
- (5) A_{\min} を広義の階層における属性 A'_{\min} に置き換える.
- (6) 重複行に関してカウンタを加算し, 重複行を削除する.
- (7) (2), (3), (4), (5), (6) をログ行数が O より小さくなるまで繰り返す.

上記の処理において属性選択の関数 $I(A)$ として平均情報量を用いる. 属性 A の要素が $\{a_1, a_2, \dots, a_{n_A}\}$ であり, 各要素の頻度が $\{p_1, p_2, \dots, p_{n_A}\}$ であるとき $I(A)$ は式 (5) により表される.

$$I(A) = - \sum_{j=1}^{n_A} (p_j \log_{n_A} p_j) \tag{5}$$

5. 処理の高速化

提案方式は長期間のログに適用されるとき, クラスタリングの処理負荷が大きく, ログ総量が時間とともに増加するため, 膨大な処理時間が必要となる. そこで, 筆者らは出力されるログを逐次計算し, 計算結果をまとめて全体を計算することにより, 膨大なログの近似的な計算結果を高速に計算する方式を提案する. つまり, 単位時間ごとのログに対して逐次アルゴリズムを実行することによりすべてのログが揃う前にアルゴリズムを実行できる. また, 単位時間ごとの結果をまとめて再度アルゴリズムを実行することによりすべてのログに対して一度にアルゴリズムを実行する場合に比べて扱うデータのサイズが小さくなるため高速な処理が可能になる. たとえば, 単位時間を 10 分とし 1 時間の結果を出力するとき, 10 分ごとのログに対してアルゴリズムを実行し, 6 回分の結果に対して再びアルゴリズムを実行する. 単位時間を 10 分, 1 時間, 1 日とすると, 10 分の結果を利用して 1 時間の結果を得て, 1 時間の結果を利用して 1 日の結果を得る (図 8). この方法による速度の向上および誤差の評価は 8.1 節において行う.

6. 異常検知

6.1 グループ解析による異常検知

AOI は膨大なネットワークログを任意の行数に要約できるアルゴリズムであり, 要約前の複数行を 1 つのグループとしてまとめることにより要約後の 1 行を生

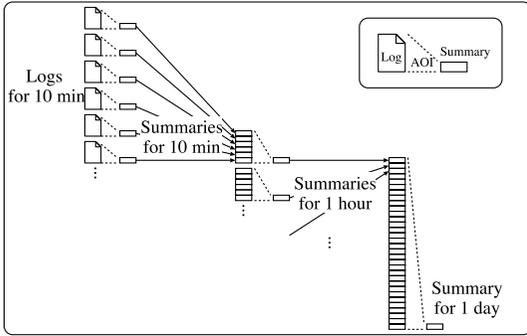


図 8 AOI の繰返し適用
Fig. 8 Iterative process of AOI.

成する。たとえば、10 万行のログを 100 行に要約するとき、複数行が 1 行にまとめられる。ここで、要約結果のそれぞれの行をグループと呼び、連続的に出力される要約結果を用いて、あるグループの時間的な変化を解析することをグループ解析と呼ぶこととする。

グループの解析を行うために 5 章において述べた繰返し解析を少なくとも 2 回以上行う。つまり、1 日のログにおけるグループの時間的な変化を解析するために、あらかじめ 1 時間や 10 分など下位の単位において要約を行い、その結果を利用して上位の単位の要約を行う。以下に 10 分ごとの要約を用いて、1 日のログにおいてグループごとに解析し異常を検知する方法を説明する。

- (1) 10 分ごとに要約を出力する。
- (2) 10 分ごとの要約による複数のグループを、さらにまとめることによって 1 日の要約を作成するとき、1 日の要約のあるグループに含まれる 10 分ごとのグループの対応関係を記憶しておく。
- (3) (2) の対応表を利用して、1 日の要約におけるグループの 10 分ごとの時間的な変化を抽出する。頻度の上位 20 グループに対して解析を行う場合、20 個のグループの時間的な変化を解析する。グループの時間的な変化は各々グラフとして表示する。
- (4) 各々グループごとのグラフにおいて、6.2 節に示す基準によって異常を検知する。

図 9 に全体の時間的な変化とグループごとの時間的な変化の概要を示す。グラフは横軸に時間、縦軸にログの頻度を示し、左側のグラフは全体のグラフであり、右側の複数のグラフは各々のグループごとのグラフである。

6.2 異常判定

一般的に、統計的分析によってネットワークログから異常検知するとき、統計量を定期的に計算し、そ

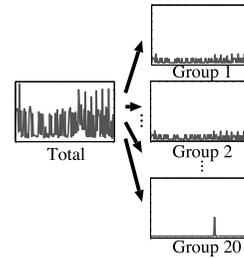


図 9 グループごと解析
Fig. 9 Grouping into some sub groups.

の統計量を過去と比較することで異常を検知できる。提案方式は統計量として、移動分散および最大振幅に対する変化の割合の 2 種類を用いる。時系列順に $x_i (i = 0, 1, \dots, m - 1)$ が与えられるとき、範囲 $n (< m)$ における、移動分散 $s_k (n - 1 < k < m - 1)$ は式 (6) で求められる。

$$\bar{x}_k = \frac{1}{n} \sum_{j=k-n+1}^k x_j,$$

$$s_k = \frac{1}{n} \sum_{i=k-n+1}^k (x_j - \bar{x}_j)^2. \tag{6}$$

また、最大振幅に対する変化の割合 $r_\ell (0 < \ell < m - 2)$ は式 (7) で求められる。

$$A_{\max} = \max_{0 \leq j \leq m-2} (x_j),$$

$$r_\ell = \frac{(x_{\ell+1} - x_\ell)}{A_{\max}} \tag{7}$$

ログの頻度分布が正規分布に似ている場合には移動分散を用い、ログの頻度が正規分布から異なる場合には変化の割合を用いることにより異常な値を検知できる。

7. 実装と評価

7.1 システム構成

実ネットワークのログを用いて提案方式を評価するために、提案アルゴリズムを組み込んだシステムを実装する。システムは IDS や FTP サーバなどのネットワーク機器からログを定期的に収集し解析する。また、解析結果は Web ブラウザを用いて閲覧できる。システムは Java で実装した。実装に用いたソフトウェアおよびライブラリを表 3 に、実装環境を表 4 に示す。

7.2 評価データ

ある企業 LAN のゲートウェイに設置した IDS²⁾ のログおよび FTP サーバ¹⁾ のログを用いて評価を行った。評価期間およびデータのサイズを表 5 に示す。

7.3 評価結果

表 6、表 7 に FTP サーバおよび IDS のある 1 日

表 7 xferlog の要約結果
Table 7 A summary of xferlog.

No.	A	B	C	D	E	F	G	H	I	J	K	L	M	lines
1	1	xxx.xxx.xxy.x/32	0 - 851	/10/NetBSD	b	-	o	a	xxxxx@xxxxx.ac.jp	ftp	0	*	c	7773
2	1	xxy.xxx.xyy.x/32	2 - 2218	/10/NetBSD	b	-	o	a	xxxxx@xxxxx.ac.jp	ftp	0	*	c	2066
3	1	0.0.0.0/0	59860 - 155286	/	b	-	o	a	Any	ftp	0	*	c	1648
4	1	0.0.0.0/0	104463 - 390255	/	b	-	o	a	Any	ftp	0	*	c	1495
5	1	xxz.xzx.xxz.yz/32	0 - 2218	/00/Linux/ packages / Caldera	b	-	o	a	guest@null	ftp	0	*	c	1493
6	1	0.0.0.0/0	6881 - 9008	/	b	-	o	a	Any	ftp	0	*	c	1414
7	1	0.0.0.0/0	0 - 2218	/	b	-	o	a	Any	ftp	0	*	c	1392
8	1	0.0.0.0/0	81286 - 185861	/	b	-	o	a	Any	ftp	0	*	c	1355
9	1	xxy.xxy.x.xzx/32	15400 - 51874	/00/Linux/packages/ Red-Hat /redhat/linux/9/en/doc /RH-DOCS	Any	-	o	a	xxxx@xx.com	ftp	0	*	c	1300
10	1	0.0.0.0/0	136112 - 777166	/	b	-	o	a	Any	ftp	0	*	c	1291
11	1	xxy.xxy.z.xzx/32	2231 - 18790	/00/Linux/packages/ Red-Hat /redhat/linux/9/en/doc /RH-DOCS	Any	-	o	a	xxxx@xx.com	ftp	0	*	c	1234
12	1	xxz.xzx.xxz.yz/32	54 - 5602	/00/Linux/ packages / Caldera	b	-	o	a	guest@null	ftp	0	*	c	1217
13	1	xxy.xyy.xzx.xyx/32	104463 - 390255	/00/Linux/packages/ Mandrake /Mandrake-devel	b	-	o	a	xxxxxx@xxxxxftp.co.jp	0	*	c	1212	
14	1	y.0.0.0/1	0-851	/00/Linux/packages	b	-	o	a	null	ftp	0	*	c	1198
15	1	xxy.xxy.z.xzx/32	9220 - 40994	/00/Linux/packages/ Red-Hat /redhat/linux/9/en/doc /RH-DOCS	b	-	o	a	xxxx@xx.com	ftp	0	*	c	1188
16	1	xxy.xxy.z.xzx/32	36283 - 99146	/00/Linux/packages/ Red-Hat /redhat/linux/9/en/doc /RH-DOCS	b	-	o	a	xxxx@xx.com	ftp	0	*	c	1144
17	1	0.0.0.0/0	0-4543	/10	b	-	o	a	Any	ftp	0	*	c	0143
18	1	xxz.xzx.xyx.zx/32	2231 - 18790	/00/Linux/packages	b	-	o	a	xxxxxx@xxxxxftp.co.jp	0	*	c	1131	
19	1	yx.0.0.0/7	104463 - 390255	/10	b	-	o	a	Any	ftp	0	*	c	1088
20	1	xxz.xzx.xyx.zx/32	54 - 5602	/00/Linux/packages	b	-	o	a	xxxxxx@xxxxxftp.co.jp	0	*	c	1085	

A:transfer-time, B:remote-host, C:file-size, D:filename, E:transfer-type, F:special-action-flag, G:direction, H:access-mode, I:username, J:service-name, K:authentication-method, L:authenticated-user-id, M:completion-status

しているグループと正規分布に類似していないグループに分離されることが分かる。

たとえば、2004年8月15日のログは正規分布に類似していたが、正規分布に類似しているグループ1と類似していないグループ18に分離された。各々のグループに対する6.2節の異常判定基準を表8に示す。グループ18は全体およびグループ1に比べて大きな値を示しており異常と判定される。

7.5 速度の評価

ログ行数に対する処理時間の評価を行う。評価は以下の3種類の解析に必要な処理時間を比較する。

- (i) 繰返し実行なし。
- (ii) ログを6個に分割して処理し、その結果をまとめて処理する2段階の処理。
- (iii) ログを144個に分割して処理し、その結果を6個ずつまとめて処理し、さらに、その結果をまとめて処理する3段階の処理。

図14に評価結果を示す。処理時間はログの行数に対して指数関数的に増加する。また、ログの行数が少ないとき繰返し実行により処理時間が増加するが、ログの行数が多いとき繰返し実行により高速に処理が可能となる。特に、約100,000行のログにおいて、(ii)

は(i)に比べ4.9倍、(iii)は(i)に比べ7.7倍の速さで処理を実行できる。

8. 考 察

8.1 処理速度と近似誤差

提案方式は、逐次、繰返し解析により近似的に高速に処理を行う。100,000行のログを約700秒で処理できることから、毎時間500,000行のログを処理できる。また、近似計算の際に発生する誤差を評価するために、全体を繰返しなしで解析した結果(i)と繰返しありで解析した結果(ii),(iii)の比較を行う。比較は解析結果における異なる項目の数である編集距離(Edit Distance)を用いる。比較結果を図15に示す。(i)と(ii)および(i)と(iii)の出力の編集距離は各々約10%であり、ログの行数が多くなるにつれて誤差は小さくなる。ここで、実際に異なる項目は概念階層における上位と下位概念にあたる包含関係にあるものが多かった。解析結果の近似誤差は、ネットワーク管理者にとって大きな影響が発生していないと思われる。

8.2 関連研究

IDSのログを解析する方式として、クラスタリングによって正しいログ(True Positive)と誤っている口

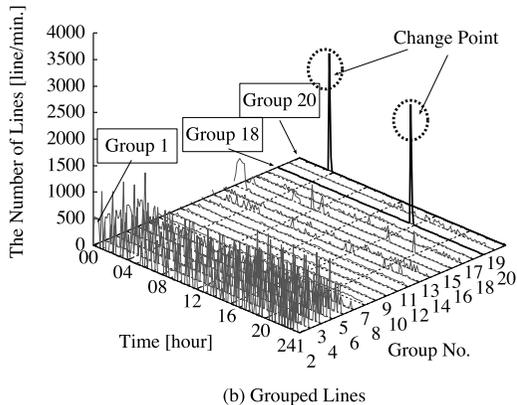
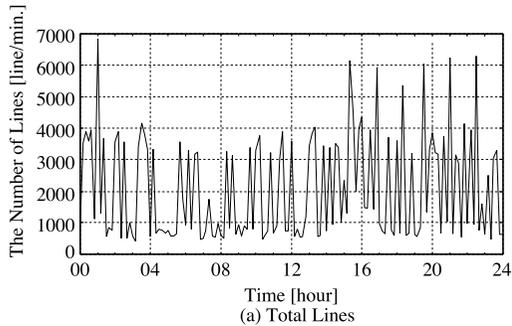


図 10 IDS ログのグループ解析結果
Fig.10 Grouping analysis for IDS's log.

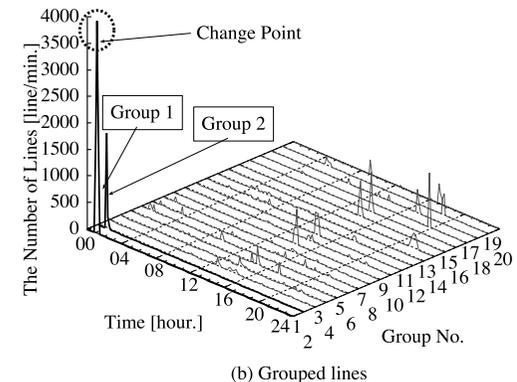
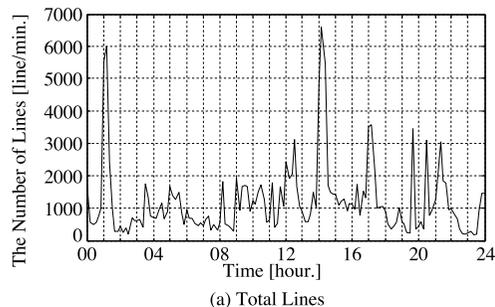


図 11 ftp サーバログのグループ解析結果
Fig.11 Grouping analysis for Ftp server's log.

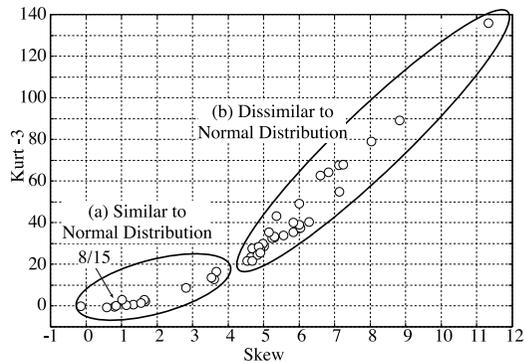


図 12 ログの全体における歪度および尖度
Fig.12 Skew and kurt of total log.

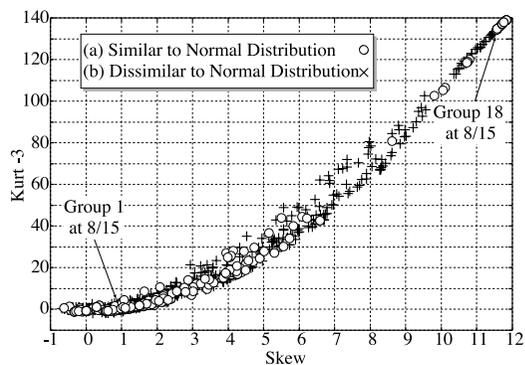


図 13 (a), (b) の要約における各グループ歪度および尖度
Fig.13 Skew and kurt of grouped logs.

表 8 2004 年 8 月 15 日のログに対する異常判定
Table 8 Statistical values of the logs.

Criterion	Total	Group 1	Group 18
Standard	2.962	2.889	11.857
Amplitude	—	0.160	0.285

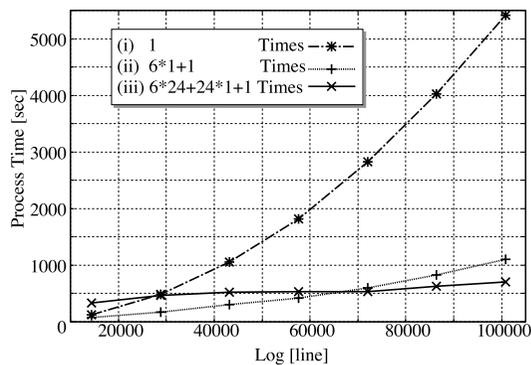


図 14 処理速度
Fig.14 Process time.

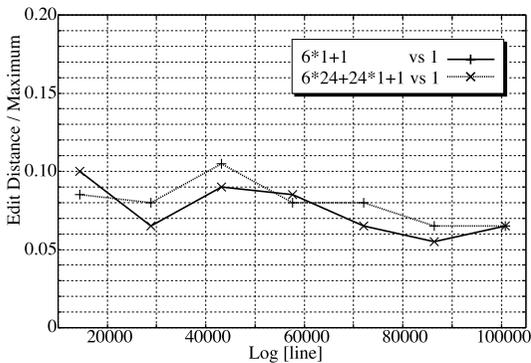


図 15 編集距離

Fig. 15 Edit distance.

グ (False Positive) を分類する方式¹³⁾ や複数箇所に設置している IDS のログの相関を用いて解析する方式^{5),14)} が提案されている。

本論文の方式は IDS の Alert ログだけでなく、アクセスログなどを含む汎用的なネットワークログを対象としている。あるネットワークログのフォーマットが与えられるとき、それぞれの列を連続値、離散値、木構造離散値のいずれかに分類して、4.2, 4.3, 4.4 節に示す方法によって適応的に概念階層を構築することができるため、様々なフォーマットに対応できる。ただし、それぞれの列を表 1 のように割り当てる作業はネットワーク管理者が行う必要がある。

一方、ネットワークログから変化点を検出する方式¹²⁾ が提案されている。文献 12) は高い精度で変化点を検出できるが、検出対象とする変数をあらかじめ指定する必要がある。提案方式はネットワークログに対して適応的に処理を行うため、あらかじめ監視する変数を指定する必要がある。

9. む す び

筆者らは、データベース上の膨大なデータを要約することで特徴を抽出する属性指向帰納を応用し、多量に出力されるネットワークログを要約して異常を検知する方式を検討した。提案方式は、ログの木構造的な性質から AOI に必要な概念階層を適応的に生成し、グループごとに統計的な解析を行うことにより、大量のログに埋もれている異常なグループを検出できる。また、AOI の処理は繰返し処理による近似計算により高速に実行できる。

実環境において収集したネットワークログを用いて提案方式の異常検知能力の評価を行った。至度および尖度の評価によって、提案方式は全体の解析では検知できない、埋もれている頻度の小さな異常を検出でき

たことを示した。また、速度評価において、提案方式は 1 時間に 500,000 行のログを処理できることを示した。

謝辞 本論文は、独立行政法人情報通信研究機構 (NICT) の委託研究「広域モニタリングシステムに関する基盤技術の研究開発」の一環として行われた。ここに深謝する。

参 考 文 献

- 1) <http://ftp.ne.jp/>
- 2) Snort - the de facto standard for intrusion detection/prevention. <http://www.snort.org/>
- 3) xferlog. <http://www.wu-ftpd.org/man/xferlog.html>
- 4) Estan, C., Savage, S. and Varghese, G.: Automatically Inferring Patterns of Resource Consumption in Network Traffic, *Applications, technologies, architectures and protocols for computer communications* (2003).
- 5) Debar, H. and Wespi, A.: Aggregation and correlation of intrusion detection alerts, *Recent Advances in Intrusion Detection* (2001).
- 6) Han, J. and Fu, Y.: Dynamic Generation and Refinement of Concept Hierarchies for Knowledge Discovery in Databases, *KDD Workshop* (1994).
- 7) Han, J. and Fu, Y.: Exploration of the Power of Attribute-Oriented Induction in Data Mining, *Advances in Knowledge Discovery and Data Mining* (1996).
- 8) Han, J., Cai, Y. and Cercone, N.: Data-driven discovery of quantitative rules in relational databases, *IEEE Trans. Knowledge and Data Engineering*, Vol.5, No.1, pp.29-40 (1993).
- 9) Julisch, K.: Clustering Intrusion Detection Alarms to Support Root Cause Analysis, *ACM Trans. Information and System Security*, Vol.6, No.4, pp.443-471 (2003).
- 10) Julisch, K. and Dacier, M.: Mining Intrusion Detection Alarms for Actionable Knowledge, *8th ACM International Conference on Knowledge Discovery and Data Mining* (2002).
- 11) K. Cho, R.K. and Kato, A.: Aguri: An Aggregation-based Traffic Profiler, *Quality of Future Internet Services* (2001).
- 12) Yamanishi, K., Takeuchi, J., Williams, G. and Milne, P.: On-line Unsupervised Outlier Detection Using Finite Mixtures with Discounting Learning Algorithms, *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2000).
- 13) Pietraszek, T.: Using Adaptive Alert Classi-

fication to Reduce False Positives in Intrusion Detection, *Recent Advances in Intrusion Detection* (2004).

- 14) Valdes, A. and Skinner, K.: Probabilistic Alert Correlation, *Recent Advances in Intrusion Detection* (2001).

(平成 17 年 11 月 28 日受付)

(平成 18 年 6 月 1 日採録)



山田 明

2001 年神戸大学大学院自然科学研究科電気電子工学専攻博士前期課程修了, 同年 KDDI (株) 入社. 現在, (株) KDDI 研究所ネットワークセキュリティグループ研究員. タイムスタンプ, インターネットセキュリティの研究に従事. 電子情報通信学会, ACM 各会員.



三宅 優 (正会員)

1990 年慶應義塾大学大学院理工学研究科電気工学専攻前期博士課程修了, 同年 KDD (株) 入社. 現在, (株) KDDI 研究所ネットワークセキュリティグループリーダー. 高速通信プロトコルの実装, インターネットアクセス, インターネットセキュリティの研究に従事. 1989 年度電気・電子情報学術振興財団猪瀬学術奨励賞, 1995 年度情報処理学会学術奨励賞受賞. 電子情報通信学会会員.



竹森 敬祐 (正会員)

1994 年慶應義塾大学大学院理工学研究科電気工学専攻前期博士課程修了, 同年 KDD (株) 入社. 2004 年同大学院博士課程修了. 現在, (株) KDDI 研究所ネットワークセキュリティグループ研究主査. 通信ネットワークおよびインターネットセキュリティの研究に従事. 2002 年度電子情報通信学会学術奨励賞受賞. 電子情報通信学会員.



田中 俊昭 (正会員)

1986 年大阪大学大学院工学研究科通信工学専攻前期博士課程修了, 同年 KDD (株) 入社. 現在, (株) KDDI 研究所情報セキュリティグループリーダー. 暗号プロトコル, 著作権保護, モバイルセキュリティ, インターネットセキュリティの研究に従事. 電子情報通信学会員.