分散データベースを用いた大規模ログ管理システム

神尾 政和† 石田 常竹† 大室 学‡ 箱田 貴久†

†安川情報システム株式会社 〒215-0004 神奈川県川崎市麻生区万福寺 1-2-3 アーシスビル 5F
‡NTT アドバンステクノロジ株式会社 〒180-0006 東京都武蔵野市中町 1-19-3 武蔵野 YS ビル 5F

E-mail: † { kamio, isidat, hakoda}@ysknet.co.jp ‡ manabu.ohmuro@ntt-at.co.jp

あらまし コンピュータネットワーク上の不正アクセスへの対策を行うためには発生した事象に関する調査が必須であり，調査の大部分はノードが出力したログに対して行われる．現在，この様な調査，解析はネットワーク管理者の技術に大きく依存している．しかし，コンピュータネットワークの拡大に伴う管理対象ノードの増加，構成の複雑化によって事象の調査は急速に困難となってきている．そこで我々は，分散配置された多数のノードから構成される大規模なネットワーク環境下で効率的，かつ，有効なログの管理，解析を実施可能とするため，分散データベースに基づいたログ収集管理システムを研究開発している．本論文ではこのログ収集管理システムの構想について説明する．

キーワード エージェント技術，分散コンピューティング，P2P，管理

Log Management System Based on Distributed Database

Masakazu KAMIO† Tsunetake ISHIDA† Manabu OHMURO‡ Takahisa HAKODA†

†Yaskawa Information Systems Corporation 1-2-3 Manpuku-ji, Asao-ku, Kawasaki, 215-0004 Japan
‡NTT Advanced Technology Corporation 1-19-3 Nakamachi, Musashino-shi, Tokyo, 180-0006 Japan

E-mail: † { kamio, isidat, hakoda }@ysknet.co.jp ‡ manabu.ohmuro@ntt-at.co.jp

Abstract It is necessary to investigate the phenomena in order to perform the countermeasures to illegal access on computer network, and it is aimed against the logs from network nodes. Now these investigations and analysis greatly depend on network administrators with advanced technology. But the number of the nodes is increasing along with the computer network expansion, and the investigations are becoming more difficult. So we are researching and developing the log management system based on the distributed database, which enables efficient log management and analysis under the large-scale network environment.

Keyword Agent technology, Distributed computing, Peer-to-Peer, Management

1. はじめに

コンピュータネットワークの発達に伴い不正アクセスやコンピュータウィルス，ワーム等による被害が増加してきている[1]．これに対し，Firewall によるアクセス制限やコンピュータウィルス等に対する“ワクチンソフト”の導入は一般的なものとなった．また，Intrusion Detection System（IDS）や Intrusion Prevention System （IPS）によるネットワーク監視やセキュリティ機器との連携も行われるようになってきている．しかし，これらによって不正アクセスを完全に防御できていないことは被害が増加傾向にあることからも明らかであり，進入に対する防御を行うだけではなく，異常の調査，対策を繰り返す必要がある．ところが，コンピュータネットワーク利用の拡大に伴いノード数は増加し，構成も複雑化してきている．さらに，大規模な組織においては地理的に分散した複数の拠点によって構成される場合も多い．そのため，コンピュータネットワークを構成する様々なノードの調査を行うことは困難になってきている．しかし，現状ではこの様な調査は高度な技術を有する管理者に依存している．加えて，全ての組織において優れた管理者を用意できるわけではない．更に地理的に分散した複数のコンピュータネットワークからなる場合は各地点に管理者が必要となる．

この様な問題に対して近年ではマルチベンダによる異なる機器によって構成されるコンピュータネット

ワークを効率的に管理するための方法が考えられている．代表的なものに Web Based Enterprise Management（WBEM）[2][3]や Application Resource Management（ARM）[4]がある．これらの仕組みは単一組織において様々な機器を管理するための手法としては有効である．しかし、複数の異なる組織が運営するネットワークをこの様な強力な統合管理手法によって集中的に管理することはできない。なぜならば、それぞれの組織が運営するネットワークの管理権限を外部の人間に付与することはできないからである。

しかし，我々は関係のある組織において異常を発見した際の情報の通知，関連情報の問い合わせ，といった緩やかな連係によっても調査，対策に効果があると考えた．そこで，ログを収集管理と異常検出を行う機能を持つログ収集管理サーバを分散配置し，いずれかのログ収集管理サーバが異常を検出した際にはそれぞれのログ収集管理サーバが検出した異常情報や関連するログの通知と収集を行うログ収集管理システムを考案し研究開発を行っている[5][6][7].

本論文ではログ収集管理サーバとこれらの連係によるログ収集管理システムの構想について述べる．また，ログからの異常検出の手法として考案した統計的手法による異常検出手法，Automaton を用いたパターンマッチングの高速化手法，及びログ収集管理サーバを柔軟に連係させるための P2P（Peer to Peer）に基づいた連携プロトコルについて説明する．

2. 構成の検討

2.1. ネットワークの規模と構成

我々の考えるログ収集管理システムは数台から十数台程度の管理対象となるノードから構成される単一のコンピュータネットワークからなる小規模なネットワークから，異なる複数の組織によって運営される複数のネットワークからなる大規模なネットワークまで，それぞれの規模に柔軟に対応可能であることを考慮している．まず，ネットワーク規模とそれに対して有効な構成について検討を行った結果について説明する．

表 1 ネットワーク規模と分類

	小規模	中規模	大規模
管理対象ノード数	数～十数	十数～百	百以上
ネットワーク数	1	複数	複数
運営組織の数	1	1	複数

まず，我々はコンピュータネットワークをその規模に応じて表 1の様に分類し，それぞれの環境下において適切と考えられるログ収集管理システムの構成について検討を行った．

小規模なコンピュータネットワークとは単一の組織によって運営される，数台から十数台の管理対象からなる，一つのコンピュータネットワーク，のことを言う．このような環境においてログの収集管理を行う場合，集積すべきログの量，ログの転送にかかるトラヒックが少ないため，一台のログ収集管理を行うサーバで管理する方法が優れていると考えられる．

中規模なコンピュータネットワークとは単一の組織によって運営される，複数の小規模なコンピュータネットワークの集合であるコンピュータネットワークのことをいう．このような環境で一カ所にログを集積する構成をとった場合，ログ収集管理を行うサーバに近づくに従ってトラヒックが増大し，大きな負荷をかける可能性がある．また，一台のログ収集管理を行うサーバにあつまるログ数の増大により，ストレージや処理能力の増大が問題となり，このようなサーバを構成することができなくなる可能性がある．この問題を解決するためには，拠点ごとにログの収集管理機能を分散し，一カ所で運用管理を統一可能な階層構成が可能なシステムであることが望ましいといえる．

大規模なコンピュータネットワークとは複数の組織によって運営される，小規模もしくは中規模のコンピュータネットワークの集合であるコンピュータネットワークのことをいう．このような環境ではそれぞれのコンピュータネットワークは運営する組織が異なるため組織ごとにログの管理を行う必要がある.従って，中規模のネットワークの様に一カ所による集中管理という構成は望ましくない．しかし，それぞれのネットワークは関連しあうため，あるネットワークで発生した異常に関連する調査をそれぞれのネットワークごとに行う必要がある．このような用件を満たすためにはそれぞれの組織に存在するログ収集管理サーバは自身の管理するコンピュータネットワーク内のノードからのログを収集し，必要に応じてほかのコンピュータネットワークに存在するログ収集管理サーバと連携動作するマルチエージェント的な構成であることが望ましいと考えられる．

つまり，ネットワークの規模，構成によってログ収集管理システムとして適する仕組みには違いがあるといえる．しかし，前述した階層構成や単一のサーバへの収集という構成はマルチエージェントによる構成の変形として実現可能である． 図 1にその様子を示す.

まず,図 1 (a)は対等な関係のログ収集管理サーバが連携しあうマルチエージェントによる構成を示している.次に,図 1 (b)ではログ収集管理サーバを木構造に配置している．このように中規模のコンピュータネットワークで有効と考えられた階層的なログ収集管理シ

ステムはマルチエージェント的なシステムの一形態として構成可能である．小規模なコンピュータネットワークに適する単一のログ収集管理サーバによる構成は図 1 (c)に示す様に複数連携可能なログ収集管理サーバのうち一台だけを利用することで同等の仕組みを構築することが可能である．

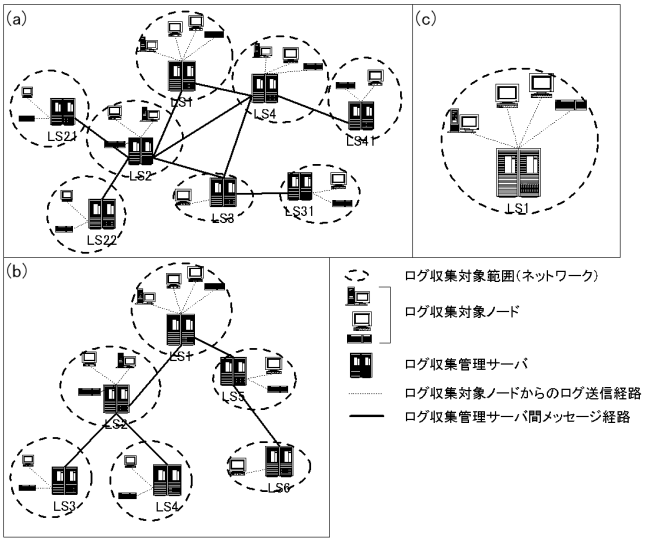

図 1 ネットワーク規模に対する構成

以上の検討から我々はそれぞれが独立に動作するログ収集管理サーバが複数連携動作するマルチエージェントシステムを構築することによってネットワークの規模や構成に対し柔軟な拡張性をもつログ収集管理システムを実現可能であると考えた．

2.2. ログ収集管理サーバ

次に我々はログ収集管理を行うそれぞれのサーバに関する検討を行った．

ログ収集管理を行うサーバは様々なアプリケーションを提供するサーバ，ルータやスイッチングハブ，Firewall や IDS といった様々なノードからのログを収集する必要がある．また，収集対象となるログとしては UNIX システムで一般的に利用される syslog，WindowsNT で用いられる Event Log，ルータやスイッチングハブで用いられることの多い SNMP（Simple Network Management Protocol），IDS などで管理者への通知手段として利用される電子メール，などがある．更に，ノードによってはそれぞれが独自に実装した方法でのログ収集に対応する必要もある．しかし，これら全てに対応することが不可能なことは明らかである．つまり，受信可能なログ形式，プロトコルは拡張可能でなければならない．これに対応する手段としては(1)ログ収集管理用のログフォーマット，プロトコルを新たに定義し，ログ収集対象のノードにログ収集用のエージェントを導入することでログの収集を実現する，(2)ログ収集管理を行うサーバ側で受信可能なログのフォーマット，プロトコルを必要に応じて追加可能にする，の 2 通りが考えられる．(1)の方法を用いた場合，新規に対応が必要となったノード用のエージェントを開発すればそのほかの部分の変更なしに対応可能であり拡張が容易だが対象とするノードによっては追加のプログラムを導入できない場合がある．そのため，我々は(2)の方法によって収集対象とするログフォーマット，プロトコルの拡大に対応することとした．

また，本ログ収集管理システムではログのネットワーク管理者による検査，解析を補助するため収集したログに対して自動的にログの検査，異常の検出を行う機能が必要であると考えた．我々の考案するログ収集管理システムでは異なる複数のログ形式を対象としている．そこで，(1)ログの種類に応じた検査機能を導入可能であること，(2)複数のログ，形式の異なるログを横断的に検査可能であること，が可能な仕組みであることを考えた． (1)を実現することによって管理対象とするログの形式に対して柔軟に検査機能を提供可能である．また，(2)を実現することによって複数の事象に基づいた異常検出，例えば IDS による異常検出結果とサーバから収集した syslog の結果を照らし合わせることによってより効果的な異常検出を実現すること，が可能となると考えた．

ここで，本システムでは受信可能なログは必要に応じて拡張可能であるとしている．しかし，ログの検査機能は受信可能なログの内容に依存して決定されるため新たなログの形式に対応した場合，それまでに実装済みのログ検査機能では新たに対応したログを検査可能である可能性は低い．従って，受信可能なログを拡張した場合には同様にログの解析機能についても拡張可能でなければならない．

以上より我々の考えるログ収集管理サーバでは必要に応じて機能の追加，変更を容易に行える必要がある．そこで我々はログ収集管理サーバを機能ごとにモジュールとして構成し，これらのモジュールを組み合わせることによってログ収集管理サーバとすることとした．その概念を図 2に示す．

図 2に示すようにログ収集管理サーバ全体はそれぞれの機能単位に分割されたモジュールの集合として構成する．また，機能単位にモジュールに分割したとしても特定のモジュール間に強い依存関係がある場合，追加，削除，変更に対する柔軟性が低くなる．そこで，我々はログ収集管理サーバを階層に分けることにより各階層のインターフェースを統一することとした．これによりモジュール間の依存性を下げることが容易になりモジュール単位での機能の追加，修正，削除が可能となり拡張の容易性を確保することができる．

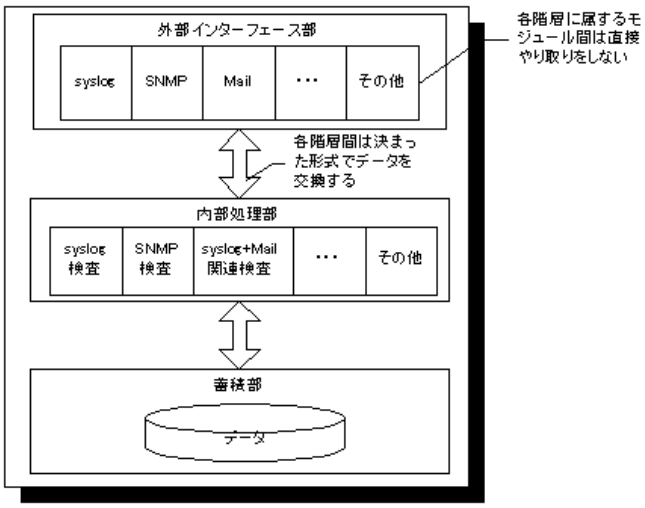

図 2 ログ収集管理サーバ構成概念図

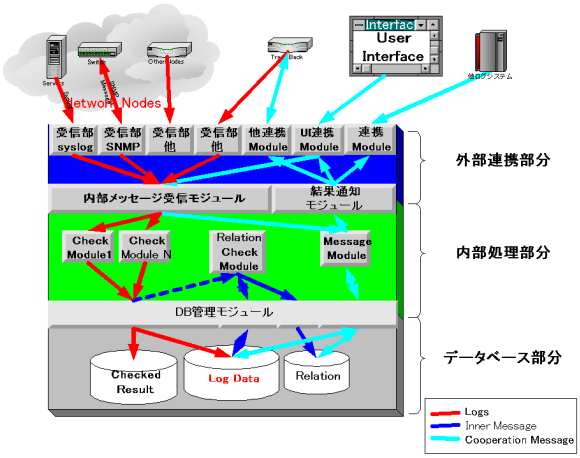

図 3 ログ収集管理サーバ構成図

また，ログ収集管理サーバは(1)ほかのノードからのログの受信，ログ収集管理サーバ間の連係といった外部とのコミュニケーションを受け持ち，内部で使用するデータ形式と外部のノードとのコミュニケーションを行うためのデータ形式の変換を行う部分として，外部インターフェース部分，(2)受信したログの検査や，蓄積したデータの関連性を検査するといった機能や，異常を検出した際のメッセージの作成，連携開始などの機能を実装する部分として，内部処理部分，(3)受信したログを蓄積するデータベースを隠蔽し，上位のモジュール群に共通のインターフェースを提供するための階層であるデータベース部分の 3 層による構成とした．

3. 提案方式

3.1. ログ収集管理サーバ

図 3にログ収集管理サーバ具体的な構成を示す．ここでは図を用いてログ収集管理サーバの動作について説明する．図 3の中央の三層からなる矩形部分がログ収集管理サーバであり，周辺に配置したものはログ収集管理対象のノード，連携対象となるほかのログ収集管理サーバ，管理のためのユーザインターフェースである．ログ収集管理サーバの各層は前述のように外部連携部分，内部処理部分，データベース部分である．それぞれの階層に配置したオブジェクトはログ収集管理サーバを構成するモジュール群である．図中の矢印は受信したログデータがデータベースに蓄積されるまで（Logs），ログ収集管理サーバ内部での検査などのためのデータ（Inner Message），他ログ収集管理サーバとの連携のためのデータ（Cooperation Message），の流れを示す．

3.1.1. 外部連係部分

外部連携部分にはログの受信，ログ収集管理サーバ間やユーザインターフェースとの連係を行うモジュールを配置する．ログ受信用モジュールはそれぞれが対象とするログを受信し，ログ収集管理サーバ内部での利用形式に変換する．

また，内部メッセージ受信モジュール，結果通知モジュールは外部連携部分と内部処理部分との間のデータの流れを制御する．これにより外部連携部分と内部処理部分のモジュール間の依存性を下げ，それぞれの独立性を保つ．

3.1.2. 内部処理部分

受信したログは内部処理部に渡され、ログの異常検査モジュールによって検査され検査結果とともにデータベースへと登録される（矢印：Logs）．この検査モジュールはログの種類ごとに用意することが可能である。逆に一種類のログに対して異なる異常検出手法を 2 以上適用することも可能であり、この様な場合は一つのログに対し複数の検査結果が関連づけられる。

また、受信したログがデータベースに蓄積されるまでの間に検査を行うだけではなく、受信済みの複数のログを時間的，ノード横断的に検査を行う（矢印：Inner Message）．

そして，これらの検査結果は検査結果を監視するモジュールによって監視され，必要に応じてほかのログ収集管理サーバに関連情報の検査を要求することでより大きな範囲での情報の収集を行う（矢印：Cooperation Message）．

3.1.3. データベース部分

ログ収集管理サーバの最下層はデータベースであり受信したログや検査結果を蓄積する。データベースは利用規模に応じて変更可能とするため、内部処理部

分とデータベース部分の間に DB 管理モジュールを配置し、データベースシステムを隠蔽し，インターフェースを統一する．

3.2. 連係機能

本システムはログ収集管理サーバをエージェントとしたマルチエージェントによる分散システムである。また、異なる組織が運営するコンピュータネットワーク間での連携を考慮している。この様な環境ではネットワーク全体の構成を知ることなく、柔軟に各エージェントを配置可能な P2P ネットワークを利用した方法が有利であると考え，P2P ネットワークによって連携する仕組みを考えた．

この仕組みでは，ログ収集管理サーバは次の様に動作する．まず，ログ収集管理サーバはいくつかのログ収集サーバを近傍として設定し，相互に通信可能とする．それぞれのログ収集管理サーバはログを収集，検査し，異常を検出した場合，P2P 通信によって近傍のログ収集管理サーバに通知，関連情報の検索依頼を送信する．メッセージを受信したログ収集管理サーバは蓄積したログの検索を行い，関連があると判断したログデータを発見した場合、起点となったログ収集管理サーバに結果を返信する．また，メッセージの送信元以外で自身の近傍に設定されたログ収集管理サーバにメッセージを転送する．この繰り返しによりネットワーク全体のログの検査を行い，結果を知ることができる．この様子を図 4に示す．

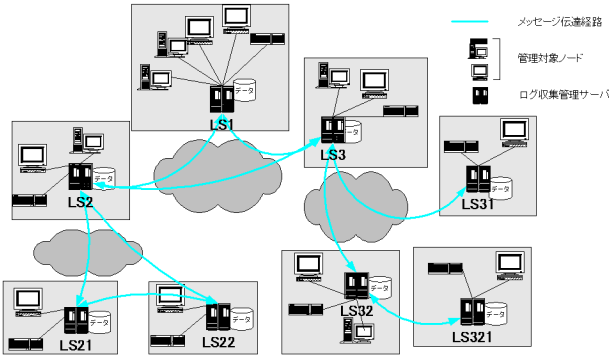

図 4 ログ収集管理サーバ連携の様子

例えば，図 4ではログ収集管理サーバ LS1 は LS2，3 と，LS2 は LS1，21，22 と近傍である．ここで LS1 が発したメッセージは LS1 から LS2，LS2 から LS21，22 へと伝達され，LS1 は直接存在を知らない LS21，22 とも連携が可能となる．同様にそのほかの部分とも連携が可能である．

しかし，途中にループ構造を含む場合，同一のメッセージの重複が発生し，余分なトラヒックを生じる．例えば先ほどの例で，図 4では LS21，22 は互いに近傍であるため LS1 からのメッセージを互いに送信しあってしまう．代表的な P2P プロトコルである Gnutella [8]では重複メッセージを破棄することである程度のトラヒック抑制を実現している．しかし，これだけでは参加する機器の数の二乗に比例してトラヒックが増加しネットワーク帯域を圧迫することとなる．分散データベースとして P2P を利用している Chord[9]ではデータの配置と検索手順を工夫することで P2P ネットワークの柔軟性を生かしながらトラヒックの抑制を実現している．しかし，本システムでは検索対象のログがどこに存在するかは未知であるためメッセージを全体に通知する必要があるため Gnutella 同様にブロードキャスト可能である必要がある．また，Chord の方式は経路の帯域幅や状態を考慮しないためメッセージの伝達に低速な経路を選択する可能性がある．そこで我々はログ収集管理サーバ間の連携を実現するため，近傍ノード間における経路の最適化機能を持った P2P プロトコルを考案した．図 5は我々の考案した P2P プロトコルでの経路制御手法を説明したものである．

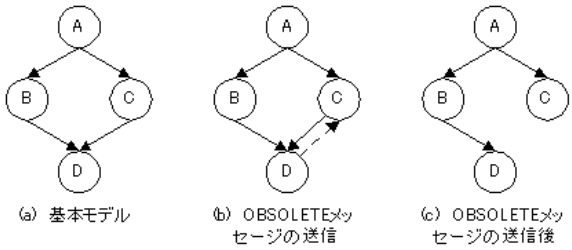

図 5 経路制御方法

図 5は，ノード A の近傍にノード B，C が位置し，ノード B,C の近傍にノード D が位置する構成を示す(図 5(a))．この構成では A が発したメッセージを D は B，C 両方から受信することになる．前述したように Gnutella プロトコルでは D は B ないし C から受け取ったメッセージのいずれかを破棄する．しかし，このような重複メッセージは常に送信されるため，明らかに無駄である．そこで，あるノードが同一の要求を受信した場合，2 つめ以降の要求を発行したノードに対して，送信停止要求(OBSOLETE)を送信し，OBSOLETE を受信したノードは，次回から要求をそのノードに送信しないという制御を考案した．例えば，D が同一の要求を B，C から受信した場合，D は後からメッセージを送信した側に対して OBSOLETE を送信する(図 5(b))．ここで D が後から受信したメッセージの送信ノードを C とすると，C は，OBSOLETE 受信後，ノード A からの要求をノード D へ送信しない(図 5(c))．この仕組みにより以降の要求伝達には無駄なトラフィックは生じない．そのため OBSOLETE による経路制御前の時点では Gnutella と同様に $O(n^2)$のトラヒックが発生

するが，OBSOLETE による経路制御が行われた時点以降のトラヒックは *O(n)*となり効果的にトラヒックを削減する．

しかし，この方法ではメッセージの中継ノードに障害が生じるとその先のノードへのメッセージを伝達できなくなるという問題ある．図 6はその様子を示したものである．

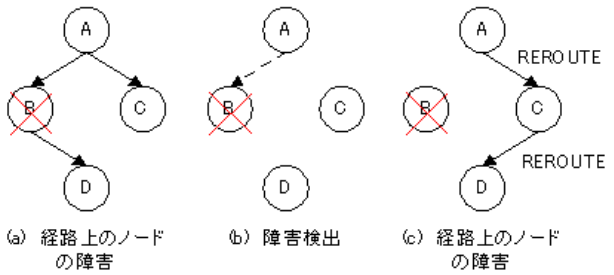

図 6 障害発生時の再経路設定

例えば，前述の伝達経路が確定した状態で B に障害が生じた場合，D にはメッセージが伝達されなくなる．しかし，障害を生じた B にメッセージが伝達不能であることを A は検出可能である．そこで我々は送信先ノードの障害を検出した場合，OBSOLETE を無視し，再ルーティングを行うことでこの問題を解決した．例えば，B に障害を検出した場合，A は B に接続不能であることを接続確立時に検知することができる．このようにして障害を検出した A は C に OBSOLETE 解除要求を送信したうえでメッセージを送信する．これによって C は D にメッセージを送信可能となる．

以上に説明したプロトコルを用いることにより，P2P ネットワークの特徴である柔軟な構成変更を実現しながら，Gnutella などの P2P プロトコルの問題点であったトラフィックの増加を抑制し，本ログ管理システムに参加するすべてのログ収集管理サーバに対して情報の伝達，結果の収集を行うことが可能である．

4. ログ検査機能

我々はそれぞれのログ収集管理サーバでの異常検出を行うことを考えている．実際に利用する異常検出の仕組みにはすでに利用されているパターンマッチングによる異常検出や SNMP で収集可能なトラヒックパターンによる異常検出を利用することが可能である。

我々はこれら以外の異常検出の仕組みとして，ログに含まれる単語の出現頻度に基づく分類手法と，高速にパターンマッチングを行うための手法として Automaton を利用する方法を考案した．以下，それぞれの手法について説明する．

4.1. 単語の出現頻度に基づく分類

本手法は既知のログをサンプリングし，それを分類した場合に，それぞれのログに含まれる単語がどの分類にどの程度出現するのか，という出現頻度行列を作成し，これを判定関数として用いて新たに受信したログを分類するという手法である．

この方法は過去のデータに基づいてそれぞれの環境に適した判定関数を作成可能である．そのため，パターンマッチングに比べて(1)未知のログを受信した場合にも異常を検出できる可能性がある，(2)サンプリングしたログを元に環境に応じた判定関数を生成可能である，という特徴がある．

4.1.1. 判定関数の作成

判定関数となる単語の出現頻度行列は単語と分類をそれぞれ行，列とし，各要素はそれぞれの単語の各分類における出現頻度を表したものである．また，この行列とそれに対応する単語の集合を対応づけたものを辞書と呼ぶ．辞書は図 7の様な構成となる．

$$\begin{pmatrix} & (\mathrm{cat}_1 & \mathrm{cat}_2 & \mathrm{cat}_3) \\ \begin{pmatrix} word_1 \\ word_2 \\ \cdot\ \cdot\ \cdot \\ \cdot\ \cdot\ \cdot \\ word_n \end{pmatrix} & \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ e_{n1} & \cdot & e_{n3} \end{bmatrix} \end{pmatrix}$$

図 7 辞書の構成

この辞書および判定関数は次の手順で作成する．(1)元データとなるログを収集し，それぞれのログを分類する．(2)1 で分類したログのメッセージを形態素解析し，単語を抽出する．(3)ホワイトスペース，数字のみからなる単語，句読点を除く．(4)2 で抽出した単語が新規に出現したものの場合はその単語を示す行を 1 行増やし，現在の分類に 1 を加える．すでに存在する単語の場合は既存の値に 1 を加える．(5)4 で作成した行列の各列をその分類に当てはめたログ数で割る．(6)5 の結果の各行の要素を各行の要素の合計で割る．

これらの捜査の結果作成される行列を判定関数として用いる．

4.1.2. 判定方法

ログの判定は次の手順で行う．(1)ログに前述の方法で作成した辞書の単語が出現したか，しなかったかを調べる．単語が出現していた場合は辞書中の単語に対応する場所に 1 を，出現しなかった場合には 0 を記録

する．(2)1 の操作によって辞書中の単語の出現，非出現に対応した 0,1 の値からなるベクトルが作成される．(3)2 のベクトルと判定関数であるベクトルを掛け合わせる．(4)3 の結果算出されたベクトルの各要素中で最大のものを選択する．その最大の要素の番号と同じ番号になる分類が判定対象のログの位置する分類となる．

4.2. Automaton を用いた特徴一致

本手法はログの出現パターンに基づいたパターンマッチングによる異常検出を高速化するために考案した手法である．我々はいくつかの連続した状態からなる Automaton を複数並列動作させることで，複数のログの関連性に基づいて異常検出を行う際の検査数の軽減と，多数のパターンを同時に検査可能とすることで高速化の実現を試みた．

4.2.1. Automaton の構成と動作

Automaton は図 8に示す様にいくつかの状態をノードとして構成され，それぞれのノードは状態として次の状態への遷移を判断するための正規表現からなる条件式を持つ．ログ収集管理サーバでは受信したログを順次 Automaton に渡し，検査を行う．

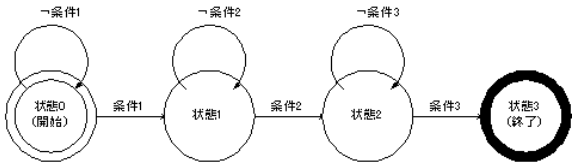

図 8 Automaton の構成

Automaton での検査は次のように行われる．(1)Automaton はログを受け取ると次の状態へ遷移するための条件と一致するかを検査する．(2)次の状態への遷移条件に一致する場合は状態を遷移する．(3)このとき，終了状態に達した場合は異常を検出したことになり，その結果を状態遷移に関わったログデータを記録して初期状態に戻る．それ以外の場合は 1 以降を繰り返すことによってログを順次検査する．

また，ログの関連性を判断する際には非常に離れた時間をおいて受け取ったログは関連性のないものとして判断した方が適当な場合がある．そこで，それぞれのノードには状態を破棄するまでの時間間隔を設定し，それ以上の時間が経過した場合は初期状態に戻ることとした．

4.2.2. Automaton の並列動作

前述のような Automaton を検査したいパターンに応じて複数準備し，これらを並列動作させる．

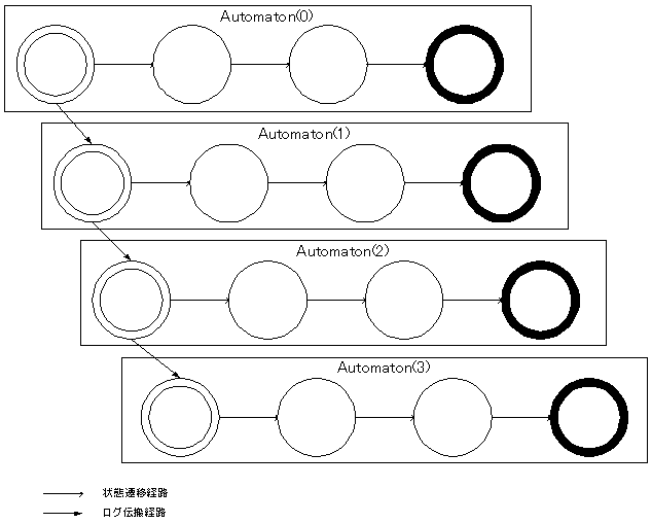

図 9 Automaton の並列動作

図 9は並列動作する Automaton のイメージを示したものである．図 9に示した Automaton の集合は次のように動作する．(1)ログ収集管理サーバが受信したログは連鎖している Automaton の先頭（図 9では Automaton(0)）にログを渡す．(2)ログを受け取った Automaton は検査，状態の遷移を行い，次の Automaton にログを転送する．(3)一つのログに対して 2 を繰り返すことによりすべての Automaton を検査する．(4)新たなログを受信するたびに 1 移行を繰り返す．

このような連鎖的に配置された Automaton による検査は，それぞれの Automaton は常にいずれかの状態にいるため，一つのログに対する検査時間は Automaton の数にのみ依存する．また，連鎖的に配置される Automaton が並列に動作しているため，連続してログが到達する場合においても Automaton の数に等しい数のログを同時に処理可能である．

5. 結果と考察

我々はこれまでに述べた構想に基づき，ログ収集管理システムのプロトタイプの実装を進めている．同時に実装した機能についていくつかの予備的な動作検証および性能の評価を行っている．以下，それらの結果について簡単に説明する．

連携プロトコルについて，我々の考案した方式では OBSOLITE による経路制御が行われる前には Gnutella と同様に $O(n^2)$のトラヒックが発生し，その後に関しては経路が論理的に木構造となるためトラヒックが $O(n)$になることが予想された．我々はこの予測に基づき実際のトラヒック測定を行うとともに Gnutella との比較を行った．

その結果は図 10に示すように我々の予想通り，経路制御が行われる前には Gnutella と同様のトラヒック

が生じたが，それ以降についてはノード数に比例したトラヒックに落ち着くことを確認した．

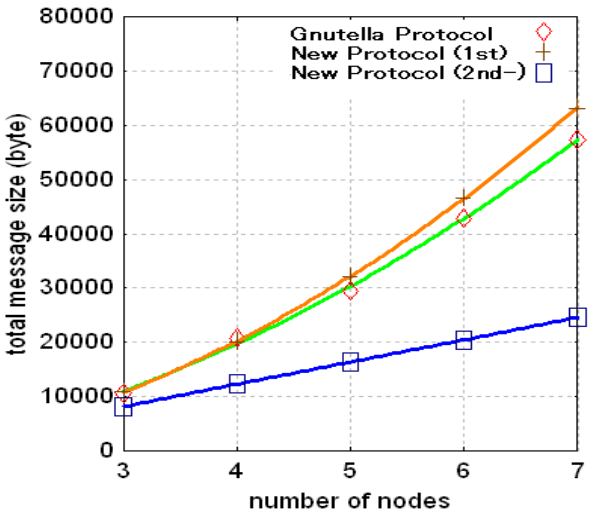

図 10 トラヒック測定結果

単語の出現頻度に基づく分類による異常検出手法がどの程度ログを分類可能かの試験を行った．まず，辞書を作成するために疑似攻撃ツール Nessus[10]によって Linux サーバを攻撃し，そのリスクごとに 3 つの分類に分けて syslog を収集し，辞書を作成した．その辞書を用いて再度 Nessus による攻撃で発生したログの検査を行った．その結果を表 2 に示す様に，88.5%の割合で適切な分類がなされるという結果になった．

表 2

	不明 安全	警告	危険	合計
ログ数	62	7	18	87
正しい	62	4	11	77
間違い	0	3	7	10
正答率	100	57.14	61.11	88.50

これと比較するため，最近 SPAM メール用のフィルタとして利用されることが多く，類似の統計に基づいたアルゴリズムであるベイズフィルタ[11]を用いて同様の試験を行った．ただし，ベイズフィルタではログから抽出可能な単語数が少ないため単語ではなく trigram を用いた．この結果，96.63%の割合で適切な分類が可能であった．この差は，ベイズフィルタでは trigram を用いるため 3 文字より長い単語では辞書中の複数の特徴的な単語が重複して一致することによるのと，検査には最も特徴的な trigram を 15 個選択するという仕組みのためノイズとなりやすい中間的な値が除去されやすいことによると考えられる．

このベイズフィルタとの比較結果から我々の考案した単語の出現頻度に基づく分類方法についてもベイズフィルタの場合と同様に各分類における最も特徴的な単語のみを選択するなどの改良を加えることで精度の向上が見込めると考えている．

6. まとめ

以上で述べたように複数のログ収集管理サーバの協調動作によりログの統一管理が可能なシステムを考案した．そして，そのために必要と考える連係動作機能，容易に機能の追加，変更が可能なログ収集管理サーバのプロトタイプの開発を行った．現在我々はこのプロトタイプを用いた予備的な機能検証を行い，その結果から本システムが有効に機能するであろうという感触を得ている．今後，このプロトタイプの拡張，修正を行いながら各機能の検証，性能の改善を進め，本システムが有効に機能することの実証を行う．

7. 謝辞

本研究は，独立行政法人情報通信研究機構における委託研究テーマ「大規模ネットワークセキュリティの確保に向けた研究開発」によっている．ここに記して謝意を表す．

文　献

[1] http://www.npa.go.jp/cyber/toukei/html/html18.htm
[2] http://www.dmtf.org/standards/wbem
[3] Distributed Management Task Force, Inc. , “Common Information Model (CIM) Specification”, http://www.dmtf.org/standards/cim/cim_spec_v22, Jun. 2004
[4] IBM Corp., “Tivoli Management Framework Planning for Deployment Guide Version 4.1.1”
[5] 福田尚弘，他，“大規模ネットワークセキュリティ確保に向けた研究開発”, JNSA Network Security Forum 2003 (NSF2003)，Oct. 2003
[6] 石田常竹，神尾政和，箱田貴久，塚本克治，清水弘，“トラフィック削減のための P2P サービスにおける経路制御手法，” 2004 電子情報通信学会総合大会，no.B-16-20,pp.629，Mar. 2004
[7] 神尾政和，石田常竹，“ログの統一管理及び異常検出に関する研究”，情報処理学会研究報告，Dec. 2004
[8] http://www.jnutella.org/docs/gnutellang/gnutella_protocolv4.shtml
[9] L. Stoica, et al. , “Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications ”, http://pdos.lcs.mit.edu/chord/
[10] http://www.nessus.org
[11] Graham, Paul. , “ Better Baysian Filtering ”, http://www.paulgraham.com/better.html，2003.1