複数のIDSを用いたログ解析による ネットワーク診断システムについて

井上 和哉[1]　立岩 佑一郎[1]　片山 喜章[1]　高橋 直久[1]

概要：インターネットは今や私たちの生活に欠かせない生活基盤であるため，ネットワークに接続された端末が外部からの不正アクセスなどの攻撃を受けると，それによって情報流出などが生じると甚大な被害が生じる．不正アクセスへの対策を講じる手段に，侵入検知システム（Intrusion Detection System:IDS）の導入が挙げられる．IDS は監視ネットワーク上で行われる不正通信を検知するものであるが，管理対象ネットワークに IDS が 1 台だけ，あるいは複数台設置されていてもそれぞれが個別に運用されている場合には，検知できない攻撃が存在する．そこで，本稿では複数の IDS を設置し，これらの IDS から発生するアラートやログを解析することで管理ネットワークに対して不正アクセスをされているか，またはネットワーク機器の設定見直しが必要であるかを診断し，診断結果をユーザに知らせるシステムを提案する．本稿では，提案システムの実現法に基づいたプロトタイプを示し，プロトタイプを用いて提案システムの評価実験を行った．

A Study on a network diagnosis system using alert logs from multiple IDS

Inoue Kazuya[1]　Tateiwa Yuichiro[1]　Katayama Yoshiaki[1]　Takahashi Naohisa[1]

1. はじめに

1.1 背景

インターネットは今や私たちの生活に欠かせない生活基盤である．そのため，ネットワークに接続された端末が外部からの不正アクセスなどの攻撃を受け，それによって情報流出などが生じると甚大な被害が生じる．不正アクセスへの対策を講じる手段にファイアウォールや侵入検知システム（Intrusion Detection System:IDS）の導入が挙げられる．IDS は通常，管理対象ネットワークに対して監視すべきネットワークセグメント単位で設置され，それぞれが個別に運用される．IDS は事前に設定されたシグネチャと呼ばれる検知ルールに基づき，シグネチャにマッチするパケットを不正アクセスとして検知する．不正アクセスを検知した際に IDS はアラートを発して，ネットワーク管理者に不正アクセスがあったことを知らせる．アラートには，不正アクセスを検出した時刻や，アラート識別子などの情報が含まれている．また，Snort[2] などの一般的な IDS では，アラートを発した原因となったパケットのログを記憶することができる．以上で述べたアラートとパケットログにはさまざまな情報が含まれている．ネットワーク管理者はこれらを参考にし，ファイアウォールの設定を行うなど不正アクセスへの対策を行う．

しかし，IDS には以下のような問題点が存在する．

問題点 1　管理対象ネットワークに IDS が 1 台だけ，あるいは複数台設置されていてもそれぞれが個別に運用されている場合には，検知できない攻撃が存在する．

問題点 2　IDS から大量のアラートが発生する．IDS は定義されたシグネチャによりアラートを発する仕組みであるため，シグネチャにマッチする通信すべてに対してアラートを発することになる．そのため，正常な通信にもかかわらず異常通信として IDS で検知されアラートが発せられるフォー

[1] 名古屋工業大学　大学院工学研究科　情報工学専攻
Nagoya Institute of Technology　Graduate School of Engineering Department of Computer Science and Engineering

ルスポジティブと呼ばれる誤検知が発生することがある．フォールスポジティブが増えてしまうと正しい検知によって発せられたアラートも含め，アラート数が非常に多くなる．アラートやパケットログに含まれる情報量も多いことから，ネットワーク管理者にとって大量のアラートから本当に対処が必要なアラートを見つけ出すことが負担となる．

問題点 1 に対し，この攻撃例の 1 つに IP スプーフィング攻撃がある．IP スプーフィング攻撃とは，攻撃者が送信元 IP アドレスを偽装したパケットを用いて行う攻撃のことである．IP スプーフィング攻撃はさまざまな攻撃と併用されることがあり，DoS 攻撃も IP スプーフィングを利用して行われることが多い．DoS 攻撃とは，サーバなどに対し莫大な数のアクセスを行い，サーバサービスの妨害を行う攻撃である．このことから IP スプーフィング攻撃を防ぐことはとても重要だと考えられる．IP スプーフィング攻撃でも，外部ネットワーク上の攻撃者が攻撃対象の存在するサブネットの IP アドレスに偽装して行う攻撃はファイアウォールなどによって対策している場合が多い．しかし，攻撃対象の存在する LAN 内の端末から，送信元 IP アドレスを外部 IP アドレスに偽装して行う攻撃はファイアウォールで対策することができない．IDS がこのような攻撃を検知すると，アラートには外部 IP アドレスが送信元 IP アドレスとして記録されているため，その IDS に記録されたアラートだけでは外部ネットワークからの攻撃と区別できず，内部 LAN の端末から内部 LAN の端末への攻撃と認識することは不可能である．もちろんこの場合，つまり攻撃者によって送信元 IP アドレスが偽装されたパケットによる IP スプーフィング攻撃は，外部ネットワークとの接続点であるファイアウォールにおいて偽装された IP アドレスに対してアクセス制限を行っても，防ぐことはできない．

次に，問題点 2 について説明する．Snort に代表される IDS は，1 日で多いと何万個のアラートが発生することがある．また，アラートを発生させた要因となるパケットが記録される際，IP ヘッダの内容や，TCP パケットであるならば TCP ヘッダ，UDP パケットであるならば UDP ヘッダと，そのパケットのペイロードなどが記録される．発生したアラートの中には，以上で述べたように誤検出も含まれているが，このような多くの情報を持つアラートやパケットログが大量に発生すると，管理者がこれらを解析しアラートへの対策を行うことは非常に負担となる．この問題に対する対処としては誤検知の原因となるシグネチャを無効化する方法がある．しかし，このシグネチャにより検出されるアラートすべてが誤検知とは限らないため，シグネチャを無効化すると本物の不正アクセスを検知できなくなる可能性がある．また，攻撃パケットのみを検知するようシグネチャを記述することも解決策として考えられる．しかし IDS の特性や管理対象のネットワーク構造の隅々まで把握する必要があるため，このようなシグネチャを記述するのは難しいと言える．

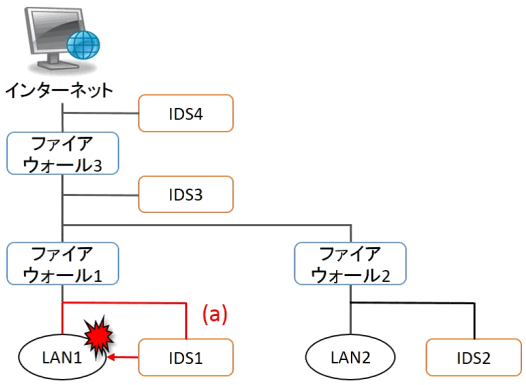

図 1　あるネットワークでの不正アクセス例

次に，**図 1** 中の赤線（a）で表される不正アクセス例と，このような不正アクセスに対する解決方法について説明する．図 1 では，送信元 IP アドレスを外部の IP アドレスに偽装し，LAN1 の端末から LAN1 の端末へ不正アクセスを行っていることを表している．IDS3 または IDS4 の位置にのみ IDS を設置する場合，このような攻撃は LAN1 内での閉じられた通信となるため，IDS はこの攻撃を検知できない．一方，IDS1 の位置に IDS を設置する場合，LAN1 を監視する IDS は，LAN1 内での閉じられた通信であるこの攻撃を検知できる．しかし，発生したアラートの送信元 IP アドレスには外部の IP アドレスが設定されているため，LAN1 からの IP スプーフィング攻撃だと認識できない．つまり，いわゆる不正パケットが IDS を通過しない場合や，仮に通過したとしてもパケットのヘッダ情報が偽装されている場合は単独の IDS だけでは攻撃を検知しても適切な対応が取れない可能性がある．

そこで，管理ネットワーク中に存在する複数の LAN に対して IDS を設置し，それらの発するアラートを総合的に解析することを考える．前述した例において，図 1 中の IDS1 から IDS4 すべての位置に IDS を設置した場合を考える．前述したとおり，実際の不正アクセスの通信経路は LAN1 内部で閉じているため，発生したアラートの送信元 IP アドレスが外部の IP アドレスであるにもかかわらず，IDS2 や IDS3，IDS4 ではこの不正アクセスを検知せず，IDS1 のみがこの不正アクセスを検知する．このことから，LAN1 において IP スプーフィング攻撃が行われていると気づくことができる．

以上より，問題点 1 を解決するためには，管理対象ネットワークに複数の IDS を設置し，かつ，それらから発せられるアラートを集め，実ネットワーク構成情報と総合的に解析することで，1 台あるいは複数台を単独で運用してい

るだけでは発見できなかった不正アクセス（不正パケット）を発見することができるようになると考える．さらに，複数の IDS から発せられるアラートは膨大になるが，その中から管理者が不正アクセスに対処するために必要な情報は限られている．そこで，アラート情報から必要な情報のみを抽出し管理者に提示することで，問題点 2 を解決できると考える．

1.2 実現上の課題と提案システムの特徴

前節で問題点 1，2 を解決する方法として，複数の IDS を設置しそれらから発せられるアラートと実ネットワーク構成情報を総合的に解析すること，およびアラート情報の中から必要な情報のみを抽出し提示することを考えたが，これらを実現するためには以下の課題が存在する．

課題 1 不正アクセスが行われた際，複数の IDS で同一パケットを追跡することは難しい．複数の IDS からのアラートを解析するにあたり，同一パケットが異なる IDS で検知されることによって生じる複数のアラートを見つけ出す必要がある．しかし，これは大変困難である．そこで，同一パケットによるアラートはほぼ同時刻に発生するであろうという仮説の下で，全 IDS の時計を同期することでこのようなアラートを求めることを考える．つまり，全 IDS の時計の同期が課題となる．

課題 2 図 1 の例で説明した通り，複数 IDS からのアラートと実ネットワーク構成情報の両方を総合的に解析することで，不正アクセスを発見できる．つまり，システムは監視対象のネットワーク構成と，各 IDS が監視している通信が監視対象ネットワークのどの部分であるかを認識しておく必要がある．

本稿では以上の課題に対し，管理対象ネットワークに対して NTP で時計が同期している複数の IDS を設置し，これらの IDS から発生するアラートやログを解析することで管理ネットワークに対して不正アクセスをされているか，ネットワーク機器の設定見直しが必要かというようなネットワーク状態を診断し，診断結果をユーザに提示するシステムを提案する．

文献 [3], [4], [5], [6] でも複数の IDS を用いて不正アクセスを検知する研究を行っている．文献 [3] では，既知の不正アクセスを定義したものを攻撃シナリオと呼ぶ．攻撃シナリオ中には，ネットワークやサーバ上で発生するさまざまなイベントが含まれ，これらのイベントは状態遷移図として表現される．ネットワーク上に複数配置された IDS で，ある IDS が攻撃シナリオに含まれるイベントを検知すると，他の IDS は検知されたイベントの遷移先のイベントの検知を行うことで，不正アクセスを検知する．文献 [3] では検知できる攻撃例として，本論文と同様に IP スプーフィングを挙げている．しかし，あらかじめ管理するネットワーク上で起こりうる IP スプーフィングのパターンに対して攻撃シナリオを定義しないと検知できない．ここでのパターンとは攻撃者，攻撃対象，攻撃者の偽装対象の IP アドレスや攻撃に関連する端末のインターフェースの組み合わせのことを言う．文献 [4] は，このような攻撃シナリオから IDS の検知能力を下げないよう，それぞれの IDS への負荷が最小となるような IDS の設置位置や攻撃シナリオの分割を行うアルゴリズムを研究している．しかし，文献 [4] もあらかじめ管理するネットワーク上で起こりうる IP スプーフィングのパターンに対して攻撃シナリオを定義しないと検知できない．文献 [5] でも複数設置された IDS の負荷や無駄なシグネチャの削減を行うことで IDS の性能向上を目的としている．文献 [6] では複数の IDS から多くのログを収集し，普段発生するアラートとは異なるものを抽出することで，未知の攻撃を発見するようなシステムが提案されているが，アラートを多く収集することを目的として複数の IDS を設置している．つまり，文献 [5], [6] では IP スプーフィングの発見を目指していない．

本稿で提案するシステムでは，複数の IDS でアラートを発する要因となる同一パケットを発見することで，それぞれの IDS に対して個別にシグネチャを設定することなく IP スプーフィングなどのパケットヘッダ偽装による攻撃を検知することができる．

提案システムの特徴を以下に示す．

特徴 1 複数の IDS から発せられる大量のアラートに対する参照や検索を迅速に実行可能な機能を提供するために，アラート情報の一部のみで構成されるキャッシュデータベーステーブルを有する．

特徴 2 複数の IDS から発せられるアラートを時間軸上に正しく整列させることにより，同一パケットにより引き起こされた可能性のあるアラートを抽出する機能を有する．

特徴 3 IDS がアラートを発する原因となったパケットの通信経路を求める機能を有する．

特徴 4 通信経路上に存在する IDS の検知パターンにより通信の分析を行い，IP スプーフィング攻撃を受けているか，またファイアウォールにより通信を遮断しているかを診断する機能を有する．

以上の特徴により，提案システムでは，ネットワーク管理者が大量のアラートから素早く目的のアラートを検索することが可能となり，またあるパケットが引き起こしたと考えられる複数のアラートを抽出することで当該パケットの通信経路を見つけることが可能となる．これによって，

今までは困難であった大量アラートの検索やIPスプーフィング等のパケットヘッダ偽造による攻撃パケットへの対応が容易になる．

2. 提案システムの概要

本稿で提案するシステムは，管理ネットワーク内に設置された複数のIDSから発生したアラートから，ある1つのパケットにより引き起こされた複数のアラートを抽出した後，これらのアラートを実ネットワーク構成情報とともに解析することでネットワーク状態を診断し，診断結果をユーザに知らせる．管理ネットワーク中でIPスプーフィングが行われていると診断した場合は，このパケットが通過した経路と攻撃者が偽装しているIPアドレスを，IPスプーフィングが行われている事と合わせてユーザに知らせる．ファイアウォールが通信を遮断していると診断した場合は，どの位置に設置されたファイアウォールで通信を遮断しているかを合わせてユーザに知らせる．ファイアウォールによる通信の遮断が行われていないと診断した場合は，そのことをユーザに知らせる．提案システムを使用すると，複数のIDSから発生するアラートを分析することで，あるパケットの実際の通信経路が分かる．実際の通信経路が分かることで，従来では検知することが困難であったパケットヘッダが偽装された攻撃を検知でき，このことをユーザに通知できる．ユーザは，パケットヘッダ偽装攻撃がされていることを従来より素早くかつ容易に知ることができ，検知された攻撃への対策をすばやく講じることが可能となる．以下では，提案システムの説明に必要な諸定義と，提案システムの構成および動作について説明する．

2.1 アラートチェインの定義

以降の説明では，監視対象ネットワークの適当な部分に複数のIDSが設置されており，かつそれらから発生するすべてのアラートが一ヶ所に集められていることが前提となっていることに注意する．提案システムでは，ある攻撃者から内部ネットワークの端末に対して攻撃が仕掛けられたとき，この攻撃にマッチするシグネチャが存在すると，攻撃者から攻撃対象の端末に至るまでのネットワーク経路上に存在する複数のIDSからアラートが発生する．このとき，ある1つのパケットによって引き起こされた，同一シグネチャによる異なるIDSから発生したアラートの系列を**アラートチェイン（Alert Chain：AC）**呼ぶ．もしネットワーク管理者がACを見つけられたら，攻撃に用いられたパケットの経路や攻撃の種類が容易に判断できる．しかし，さまざまな攻撃がほぼ同時に連続して行われている場合，IDSから多くのアラートが発生するため，アラートログから特定のパケットによるACを見つけ出すことは困難である．つまり，異なるパケットによってほぼ同時に同一シグネチャによるアラートが複数のIDSから発生する可能性があり，その中から実際にある1つのパケットによって発生したACを求めることはほぼ不可能である．そこで提案システムでは，複数のIDSから発生するアラートから，ACを構成する可能性のあるアラートを抽出し，それらのすべての組み合わせを列挙することで本来のACが含まれている**アラートチェイン候補集合（Alert Chain Candidate Set：ACCS）**を求める．

ACCSはアラートチェイン候補（Alert Chain Candidate：ACC）の集合であり，ACCは，ACに含まれる可能性のあるアラート（の集合）を求め，それらを組み合わせることで導出する．このようなACCを導出するためのアラート集合を**アラートチェイン要素集合（Alert Chain Element Set：ACES）**と呼び，この集合に属するアラートを**アラートチェイン要素（Alert Chain Element：ACE）**と呼ぶ．ACESに含まれるアラートをIDSごとに分別したもののアラート集合を**局所アラートチェイン要素集合（Local Alert Chain Element Set：LACES）**と呼ぶ．

提案システムでは，以上で説明したACを求めるために，まずユーザはアラートを1つ指定する．指定されたアラートを発生させる要因となったパケットによって他のIDSで発生したアラートを求め，得られたアラートの系列をACとする．しかし，以上で示したようにACを見つけ出すことが困難となるため，ユーザに指定されたアラートから，はじめにACESを求める．求めたACESは複数のIDSから発生したアラートの集合であるため，IDSごとに分類したLACESを求める．求めたLACESからアラートの全組み合わせを取得することで，本来のACが含まれる可能性のあるACCを求める．

次に，ACを求めるために必要な情報であるLACES，ACES，ACCSを厳密に定義する．定義に用いるパラメータは以下の通りである．

- n：IDSの台数
- m：ACCS中に含まれるACCの総数
- $\mathrm{IDS}_i (1 \leq i \leq n)$：IDS番号が$i$であるIDS
- $\mathrm{Log}_i (1 \leq i \leq n)$：$\mathrm{IDS}_i$上で発生したアラートの集合
- $a_j^i (j \geq 1)$：Log_iに含まれ，アラート番号がj番であるアラート
- $h(a_j^i)$：アラートa_j^iを引き起こしたパケットのヘッダ情報
- $sig(a_j^i)$：アラートa_j^iを引き起こしたシグネチャ
- $t(a_j^i)$：アラートa_j^iが発生した時刻

あるアラートa_j^kが指定されたとき，IDS_i上で形成されるLACESをLACES_iとすると，LACESの定義式は

$$\mathrm{LACES}_i\left(a_j^k\right)(\text{ただし}, i \neq k) = \{a_m^i \mid h\left(a_m^i\right) = h\left(a_j^k\right) \wedge sig\left(a_m^i\right) = sig\left(a_j^k\right) \wedge |t\left(a_m^i\right) - t\left(a_j^k\right)| \leq T\}$$

となる．式中のTは，各IDSで発生する同一パケットによるアラートの発生時刻の範囲を表すものである．例えば

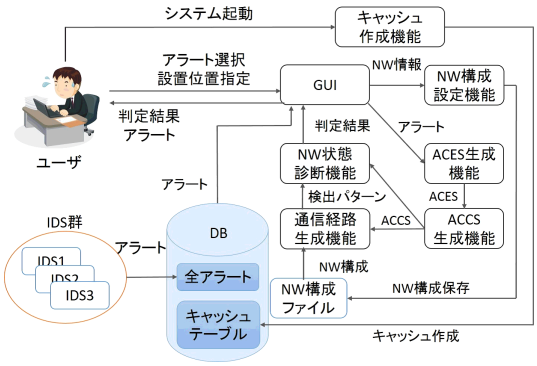

図 2　提案システムの構成図

NTP などで時計がほぼ同期されている場合は，Snort の時刻粒度が秒であるため，$T = 1$ とすればよい．これによって，ほぼ同時に発生するであろう同一パケットによるアラートを指定できる．以上より，$\mathrm{LACE}_i\left(a_j^k\right)$ によって定義される LACES は，IDS_i 以外の IDS で発生したアラートで，$\left(a_j^k\right)$ と同じヘッダ情報を持ち，かつ $\left(a_j^k\right)$ と同じシグネチャによって発生し，かつ $\left(a_j^k\right)$ が発生した時刻の T 時間前後に発生したアラートの集合である．あるアラート a_j^k に関する ACES は LACES を用いて

$$\mathrm{ACES}\left(a_j^k\right) = \bigcup_{p=1}^{k-1} \mathrm{LACES}_p\left(a_j^k\right) \cup \bigcup_{p=k+1}^{n} \mathrm{LACES}_p\left(a_j^k\right)$$

のように定義できる．以上より求められる ACCS の定義式は

$$\mathrm{ACCS}\left(a_j^k\right) = \prod_{p=1}^{n} \mathrm{LACES}_p\left(a_j^k\right)$$
$$\text{ただし}, \mathrm{LACES}_k\left(a_j^k\right) = \left\{a_j^k\right\}$$

の通りである．ACCS には AC になりうるようなアラートの組み合わせである ACC が含まれていることから，あるアラート a_j^k に関する ACCS は

$$\mathrm{ACCS}\left(a_j^k\right) = \{\mathrm{ACC}_1, \mathrm{ACC}_2, \cdots, \mathrm{ACC}_{m-1}, \mathrm{ACC}_m\}$$

と表す．ACCS に含まれる m 個の ACC の中に本来求めたい AC が存在する．

2.2　提案システムの構成

提案システムの構成図を図 **2** に示す．図 2 中の NW とはネットワーク，DB とはデータベースを表す．提案システムはキャッシュデータベーステーブル作成機能とネットワーク構成設定機能，ACES 生成機能，ACCS 生成機能，通信経路生成機能，ネットワーク状態診断機能，IDS から発生したアラートを含んだデータベースから構成される．

2.2.1　キャッシュデータベーステーブル作成機能

管理ネットワーク中の複数の IDS が発したすべてのアラートやアラートを発したパケットのログは，1 台の DB に送信する．まずユーザがシステムを起動すると，キャッシュデータベーステーブル作成機能が DB に格納されている全アラートから直近 10 万件のアラートを対象とし，必要な情報のみを抽出したキャッシュデータベーステーブル（以降，キャッシュと呼ぶ）を作成する．ユーザがアラートを検索したり，提案システムにおいてアラートを解析する際にすべてのアラートが格納された DB を参照するのはとても時間がかかる．本機能は，複数の IDS から検出されるアラートがとても多いことが原因で起こるこのような問題を解決できる．

2.2.2　ネットワーク構成設定機能

ユーザはネットワーク構成設定機能を用い，管理対象ネットワーク中に存在するハブやルータ，計算機をノード，これらの接続関係をエッジとすることで管理ネットワークを木構造で表現する．本機能はユーザから入力されたハブやルータ，計算機自身の持つ IP アドレス，これらの接続関係を入力することで，管理ネットワークをノードに IP アドレスや，どのノードに接続されているかというような情報を持った木構造で表現し，ファイルとして保存する．この機能を用いることで，計算機やハブなどが実ネットワーク上のどの位置に設置されたものなのか，また IDS はどの位置を通る通信を監視しているのかを提案システムから参照できるようになる．

2.2.3　ACES 生成機能

ある特定のパケットによる AC を見つけ出すのは困難であるため，本機能では，ユーザにより指定されたアラートが発生した要因となったパケットに対し，前節で述べた ACES の定義式に合致するような，指定されたアラートが発生した IDS とは異なる IDS から発生したアラートをキャッシュに格納されているアラート群から検索し，求めたアラートの集合を ACES として出力する．

2.2.4　ACCS 生成機能

ACCS 生成機能では，入力された ACES から管理対象ネットワークに設置されている IDS に対してそれぞれ LACES を求める．求めた LACES からアラートを取り出し，全組み合わせを求め，これを ACCS として出力する．このように求めた ACCS には AC が含まれる可能性があり，本機能の出力結果は，パケットが実際に通った経路を見つけるのに有用である．

2.2.5　通信経路上の IDS 抽出機能

通信経路生成機能では，ACCS に含まれる ACC が入力されたとき，ACC に含まれるアラートのパケットヘッダ情報より通信経路を見つけ，通信経路上に存在する IDS を求める．通信経路を求める際は，ネットワーク構成設定機能から出力されたネットワーク構成情報を利用している．

2.2.6　ネットワーク状態診断機能

ネットワーク診断機能では，通信経路生成機能から出力された実際にパケットが通った経路と理論的な通信経路上に存在する IDS を比較することで管理対象ネットワーク

上でどのような問題が発生している可能性があるかを診断し，診断結果を出力する．この機能では，内部 LAN から外部の IP アドレスを用いた IP スプーフィングをはじめとするさまざまなパターンの IP スプーフィングが行われているか，またファイアウォールによる通信の遮断が行われているかを診断することができる．IP スプーフィングが行われていると診断した場合は，ネットワーク構成設定機能における木構造でのこのパケットが通過した経路上のノードと攻撃者が偽装している IP アドレスを割り出す．一方，ファイアウォールによる通信の遮断が行われていると診断した場合は，どの位置に設置されたファイアウォールで通信を遮断しているかを調べることができる．

2.3 提案システムの動作

提案システムの動作の流れを以下に示す．

Step1 システムが起動されたとき，キャッシュ作成機能が全アラートが格納されているデータベース内にキャッシュを作成する．

Step2 ユーザが GUI に対して管理対象ネットワークに存在するハブやルータ，計算機の接続関係やこれらに割り当てられている IP アドレスを入力する．入力された情報を元に管理ネットワークを木構造として表現することでネットワーク構成を設定し，ファイルとして出力する．

Step3 ユーザがキャッシュに格納されたアラートから GUI を通して 1 個選択する．ACES 生成機能は，ユーザに選択されたアラートを用いて，キャッシュからそのアラートに関する ACE を検索し，見つかった ACE を集合（ACES）として出力し，ACCS 生成機能に入力する．

Step4 ACCS 生成機能は，Step3 で求めた ACES から LACES を求めた後，LACES から得られるアラートの全組み合わせを求めることで ACCS を生成し，通信経路生成機能に入力する．

Step5 通信経路生成機能は，Step4 で求めた ACCS に含まれるアラートを引き起こしたパケットのヘッダから理論的な通信経路上にある IDS を割り出し，ネットワーク状態診断機能に入力する．

Step6 通信経路生成機能は，ACCS から ACC を取り出し，ネットワーク状態診断機能に入力する．

Step7 ネットワーク状態診断機能は Step5 と Step6 で入力された通信経路を比較することでネットワーク状態を診断し，診断結果をユーザに知らせる．

表 1 キャッシュの概要

カラム名	型	概要
sid	int	IDS 番号
cid	int	IDS ごとに定められるアラート番号
timestamp	timestamp	アラート発生日時
sig_id	int	シグネチャ ID
sig_name	char	シグネチャ名
ip_proto	int	プロトコル ID
ip_src	int	送信元 IP アドレス
sport	int	送信元ポート番号
ip_dst	int	宛先 IP アドレス
dport	int	宛先ポート番号

3. 提案システムの実現法

3.1 キャッシュデータベーステーブル作成機能の実現法

本機能で作成する各カラムの情報は，すべてアラートに含まれる情報であり，一部はそのアラートを発生させるに至ったパケットのヘッダ情報から引用されたものである．システム起動時に，管理ネットワーク中に設置されたすべての IDS から発生した全アラートが格納された DB から，**表 1** のようなカラムを持つキャッシュを作成する．キャッシュは全アラートのうち直近 10 万件のアラートの，提案システムで解析を行う際に必要なアラート情報やパケットログのみを保持している．キャッシュにおけるレコードには，1 つのアラートの持つ情報と，そのアラートを引き起こしたパケットログが保存してある．なお，提案システムで用いる IDS から発生したアラートには，IDS を区別するための ID（sid とする），同じ IDS から発生したアラートを区別するための ID（cid とする），シグネチャを区別するための ID（sig_id とする），シグネチャの名称（sig_name とする）の情報が含まれていると想定する．

cid と sid により，DB 上に存在するアラートが一意に定まる．timestamp は IDS がアラートを発した時刻を表す．ip_proto はプロトコル番号 [7] を表していて，これが 1 であれば ICMP パケット，6 であれば TCP パケット，17 であれば UDP パケットである．ip_src と ip_dst はそれぞれ送信元，宛先 IP アドレスを表す．sport と dport はそれぞれ送信元，宛先ポート番号を表す．

キャッシュを作成する手順を以下に示す．

Step1 表 1 で示したすべてのカラムが 1 つのテーブルに格納されていない場合，Step1 のテーブルと表 1 で示したカラムが含まれるテーブルを，cid と sid の組が等しいレコード，すなわち同一アラートのログが 1 つのレコードとなるように結合し，表 1 で示したすべてのカラムを含むテーブルを作成する．

Step2 Step1 のテーブルから直近 10 万件のレコードを選択する．

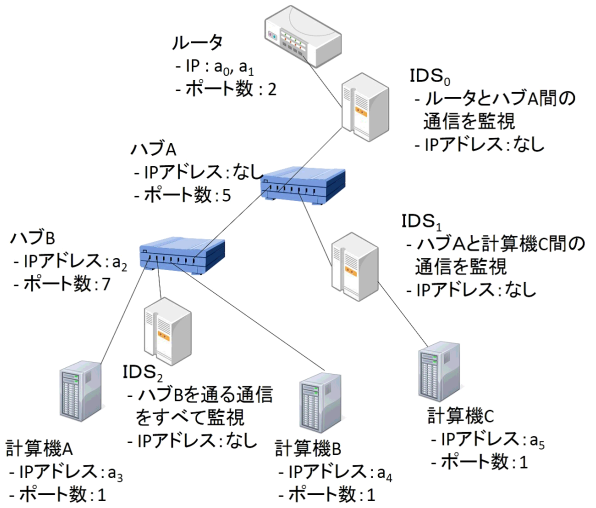

図 3　ネットワーク構成を設定する実ネットワーク図

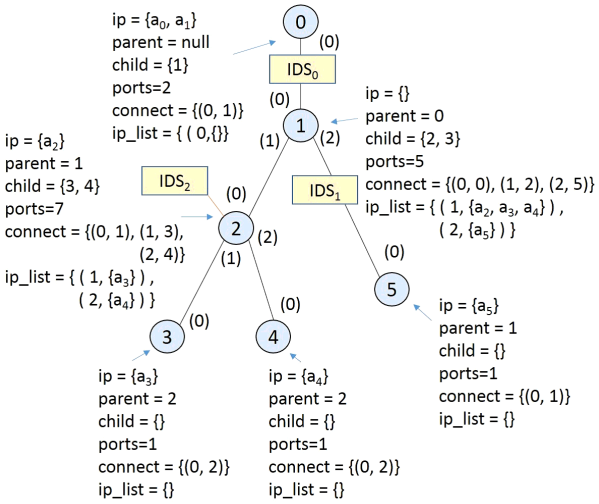

図 4　ネットワーク構成設定機能により生成されたネットワーク構成

Step3　Step2 のテーブルから，表 1 で示したカラムのみが含まれるように他のカラムを破棄し，破棄後残ったテーブルをキャッシュとする．

3.2　ネットワーク構成設定機能の実現法

管理対象ネットワークにおいて，ネットワーク中の各端末間，および外部ネットワークと各端末間の通信経路は唯一であるとする．提案システムでは，管理対象ネットワーク中に存在するハブやルータ，計算機をノード，これらの接続関係をエッジとすることで管理ネットワークを木構造で表現する．例えばハブをあらわすノードは，自身の識別子と IP アドレス，上流に接続されているノードの識別子，そのハブの下流に接続されているノードの識別子集合および，それらが接続されているポート番号，さらにそのハブの下流に接続されるノードの持つ IP アドレスのリストを持つものとする．ハブ以外のルータや計算機も同様に情報を持たせることで，実ネットワークの構成情報をモデル化した木構造グラフが構築可能となる．本機能におけるポートは，ハブやルータ，計算機が持つイーサネットの接続口とする．実ネットワーク構成情報を表現する木構造グラフにおいて，ノードの持つ情報を以下に示す．

- **id**：ノード番号を表す．0 以上の整数で表し，それぞれのノードに対して id はユニークな値を持つ．
- **ip**= $\{a \mid a =$ ノードの持つ IP アドレス $\}$：ノードとして表されるハブやルータ，計算機が持つ IP アドレスのリストを表す．
- **parent**：親ノードの id を表す．つまり，parent は自身のノードの上流に接続されているハブやルータを表すノードの id を表す．
- **child**= $\{ch \mid ch =$ 接続された子ノードの id$\}$：子ノードの id のリストを持つ．
- **ports**：ハブやルータ，計算機が持つイーサネットの接続口であるポートの数を表す．

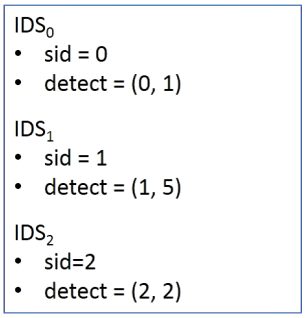

図 5　ネットワーク構成設定機能により生成された IDS 情報

- **connect**= $\{con \mid con = (\text{port\_num}, \text{id})\}$：ports で設定されたそれぞれのポートに対し，ポートの先にどのノードが接続されているかを表す．つまり，connect の要素数は ports に等しい．port_num は例えば ports=3 であれば port_num=0, 1, 2 となるように，0 以上 ports 未満の整数で表す．ノードの識別には id を用いる．
- **ip_list**= $\{alist \mid alist = (\text{port\_num}, \{\text{child.ip}, \text{child.ip\_list}\})\}$：connect で設定されたそれぞれのポートに対し，各ポートに接続されているノードの ip と，そのノードの持つすべての ip_list に含まれる IP アドレスを持つ．ip_list の要素数は ports-1 に等しい．これは親ノードに接続されているポートに対する ip_list は持たないためである．

また，提案システムは管理ネットワーク中の IDS が実ネットワーク中のどの位置を監視しているかを知る必要がある．よって，複数の IDS のうちどの IDS が，管理ネットワークを表現した木構造中のどの位置を監視しているかを表現するための情報を以下に示す．

- **sid**：IDS を区別する識別子である．1 つの IDS はキャッシュに用いられる sid とこの sid で同じ値を持つ．
- **detect**：IDS の監視位置である．監視位置を木構造中のノードの id の組として表す．たとえば，ノード A とノード B 間を通る通信を監視していた場合は {A.

id，B．id}，ハブ（ノード C とする）を通過するすべての通信を監視している場合は {C．id，C．id} のように同一ノードを指定する．

管理者は id，ip，parent，child，ports，connect，sid，detect を，管理ネットワーク中の計算機やハブ同士の接続の仕方により設定する．ip_list は管理者により設定された情報をもとに本機能が自動で更新する．すべての情報を更新後，ファイルとして保存する．以降，sid=i である IDS を IDS_i と表す．

例えば**図 3** のようなネットワーク構成例に対して，ネットワーク構成設定機能は**図 4** のようにネットワーク構成を生成する．図 4 におけるノード番号が id，各節点付近にある（1）のような数字が port_num を表す．図 3 中のハブ A と計算機 A の設定方法について述べる．計算機 A は上流のハブ B に接続されており，自身の IP アドレス a_3 を持っていてポート数が 1 である．このような情報からユーザは id=3，ip= $\{a_1\}$，parent=1，child= {}，ports=1，connect= {(0，1)} のように計算機 A を設定できる．id は自由に値を設定できるが，他ノードの id と等しい値にならないように注意する．一方，ハブ A は下流にハブ B と計算機 C が，上流にルータが接続されており，自身の IP アドレスは持っておらずポート数が 5 である．このような情報からユーザは，id=1，ip= {}，parent=null，child= {2}，ports=5，connect= {(0，0)，(1，2)，(2，5)} のようにハブ A を設定できる．本機能は connect から，id=1 のノードのポート番号 1 に接続される id=2 であるノード，id=1 のノードのポート番号 2 に接続される id=5 であるノードが分かる．connect と id=2 のノードの ip と ip_list，id=5 のノードの ip と ip_list から，ip_list= {(1，$\{a_2, a_3, a_4\}$)，(2，$\{a_5\}$)} と設定できる．

次に，IDS_2 に関する本機能での設定方法を述べる．IDS_2 は IP アドレスを持てば IDS が通信を行う可能性があるため木構造内のノードとして表す必要があるが，IP アドレスを持たないため木構造内のノードとして表さない．IDS_2 は sid=2 とし，id=2 のノードを通過するすべての通信を監視するため detect=（2，2）と設定できる．

3.3 アラートチェイン要素集合機能の実現法

アラートチェイン要素集合（ACES) 生成機能は，入力されたアラートの sid と cid から，入力されたアラートを引き起こしたパケットヘッダとアラートを発したシグネチャが等しく，発生時刻の差が設定時間以内であるようなアラートすべてを含む ACES を求める．ACES を求める手順を以下に示す．

Step1 キャッシュから，入力された sid と cid が等しいレコードの，timestamp，sig_id，ip_proto，ip_src，sport，ip_dst，dport を取得する．

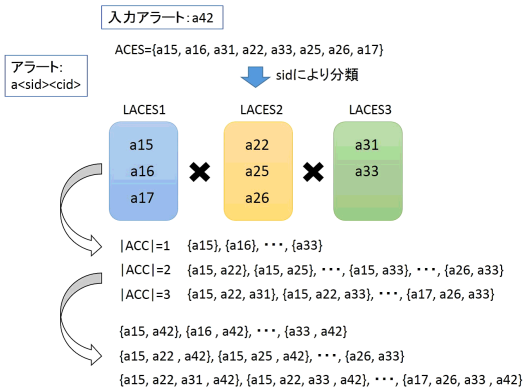

図 6 ACCS の生成の流れ

Step2 キャッシュから，Step1 で取得した値 sig_id，ip_proto，ip_src，sport，ip_dst，dport が等しく，取得した timestamp の時刻との差が設定時間以下となるようなすべてのレコードの sid と cid を取得する．このとき，入力されたアラートが発生した IDS とは異なる IDS から発生したアラートを選択する．つまり入力された sid とは異なる sid を持つレコードを選択する．

Step3 Step2 で取得したすべてのアラートの集合を ACES とし，出力する．

3.4 アラートチェイン候補集合生成機能の実現法

アラートチェイン候補集合（ACCS）生成機能は，入力された ACES を LACES に分類し，AC となりうるアラートの組み合わせである ACC を求め，これらすべてを集合である ACCS として出力する．AC に含まれるアラートの条件を満たすアラートが ACE であるが，実際に AC を構成するようなアラートの組み合わせを見つけることは困難である．よって，ACE の発生元である IDS により分類し，分類後できた集合の直積を求めることで，考えられるアラートの全組み合わせを得られる．こうして生成された ACCS 中に AC が含まれる可能性がある．以下に**図 6** を例として ACCS を求める手順を示す．

Step1 入力された ACES を LACES に分類する．図 6 中の 8 つのアラートは 3 種類の IDS（sid が 1，2，3 のもの）から発生しているため，3 つの LACES が生成される．

Step2 Step1 でできた LACES の直積を計算し，ACC を求める．このとき，ACC の要素数が小さい順に直積を求める．図 6 のように，要素数が 1 である ACC から順に直積を求める．この例では 3 つの LACES があるため，ACC の要素数は最大で 3 となる．

Step3 Step3 取得したすべての ACC に対し ACES 生成機

能で入力されたアラートを加え ACC を更新する．

Step4　Step4 までで求めたすべての ACC を集合 ACCS とし，出力する．

3.5　通信経路上の IDS 抽出機能実現法

通信経路上の IDS 抽出機能は，入力された ACCS 中の ACC に含まれるアラートのパケットヘッダ情報が持つ送信元 IP アドレス（srcIP とする）と宛先 IP アドレス（dstIP とする）を求め，これとネットワーク構成設定機能で設定されたデータを利用し，通信経路上に存在する IDS を出力する．本機能を実行する際に使用する経路 path とは，通信経路上に現れるネットワーク構成を表すノードの id を保持する順序付集合であり，パケットヘッダから分かる理論的な通信経路をあらわす．path の先頭には通信経路の始点となるノードの id，末尾には終点となるノードの id が格納される．path は通信経路上に存在する IDS を求める際に必要となる．count[i] は，IDS$_i$ の detect に設定されているノード対の両方が path に含まれている場合は 2，ノード対の一方が含まれている場合は 1，ノード対の両方が含まれていない場合は 0 となる．また本機能が出力する，通信系路上に存在する IDS の sid の系列を idsOnPath とする．本機能の実行手順を以下に示す．

Step1　入力された ACCS から ACC を 1 つ取り出す．

Step2　取り出した ACC に含まれるアラートのパケットヘッダ情報（hdrInfo とする）が持つ srcIP を求める．

Step3　srcIP を自身の IP アドレス（ip）として持つノードがネットワーク構成情報中に存在するか調べる．

（a）　存在した場合，そのノードを path の先頭に追加し，このノードを注目ノードとする．

（b）　存在しない場合，ネットワーク構成情報中の最上流ノードを path の先頭に追加し，このノードを注目ノードとする．

Step4　hdrInfo が持つ dstIP と srcIP を比較する．

（a）　等しい場合，Step6 へ進む．

（b）　異なる場合，注目ノードの ip_list に dstIP を持つ要素が存在するか調べる．

（i）　存在した場合，ip_list と connect から，その ip_list を持つノードの id を求め，path の末尾にその id を追加し，注目ノードとする．

（ii）　存在しない場合，注目ノードの parent から親ノードの id を求める．

（A）　parent に親ノードの id が設定されていない場合，Step6 へ進む．

（B）　parent に親ノードの id が設定されていた場合，path の末尾にその id を追加し，注目ノードとする．

Step5　注目ノードの ip と dstIP が等しいか調べる．

（a）　等しい場合，Step6 へ進む．

（b）　異なる場合，Step4（b）に戻る．

Step6　path の先頭の id を取り出す．

Step7　取り出した id が IDS 情報中のどの IDS の detect に含まれるか調べる．

（a）　取り出した id が，IDS$_i$($0 \leq i \leq n-1, n$: 管理ネットワーク中の IDS の台数) の detect に設定されているノード id 対の両方と等しい場合 count[i] に 2 を足し，ノード対の一方のみと等しい場合 count[i] に 1 を足す．足した後，Step6 に戻る．

（b）　取り出した id がどの IDS の detect にも含まれていなかった場合，Step8 へ進む．

Step8　count[i]=2 である IDS$_i$ の sid を idsOnPath の末尾に追加し，path 中にノードの id があるか調べる．

（a）　path 中に id がある場合，Step6 へ戻る．

（b）　path 中に id がない場合，Step9 へ進む．

Step9　idsOnPath を出力する．

このようにして出力された idsOnPath には，パケットヘッダから分かる理論的な通信経路上に設置された IDS すべてが含まれている．これらの IDS は，ヘッダどおりの通信が行われていれば本来アラートを発生すべき IDS である．

本機能の動作例を図 4 を用いて説明する．入力として与えられた ACCS から，IDS$_0$ と IDS$_1$ から発生したアラートを含む ACC を取り出したとする．hdrInfo に設定された srcIP は a_5，dstIP は a_{10} とする．ip に a_5 を含むノードは id=5 のノードである．3 を path の先頭に追加し，このノードを注目ノードとする．

次に，srcIP と dstIP は異なるため，id=5 のノードの ip_list に a_{10} を持つ要素を探す．しかし，id=5 のノードの ip_list にはそのような要素は存在しないため，このノードの parent から親ノードの id を求める．求まる id は 1 であるため，path の末尾に 1 を追加し，id=1 のノードを注目ノードとする．現在，path=(5,1) となっている．

id=1 のノードの ip と dstIP である a_{10} は異なるため，Step4（b）に戻り，id=1 のノードの ip_list に a_{10} を持つ要素を探す．しかし，id=1 のノードも ip_list にそのような要素は存在しないため，このノードの parent から親ノードの id を求める．求まる id は 0 で，path の末尾に 0 を追加し，id=0 のノードを注目ノードとする．現在，path=(5,1,0) となっている．

id=0 のノードの ip と a_{10} は異なるため，Step4（b）に

戻り，id=0 のノードの ip_list に a_{10} を持つ要素を探す．しかし，id=0 のノードの ip_list にそのような要素は存在しない．ここで，id=0 のノードの parent には id が設定されていないことが分かる．よって，通信経路は path=(5,1,0) で確定する．

Step6 へ進み，path の先頭の 5 を取り出したとき，IDS_1 の detect に id=5 が 1 つ含まれるため count[1]=1 となる．次に，path の先頭の id を取り出す．取り出した id は 1 であり，IDS_1 と IDS_0 の detect には 1 がそれぞれ 1 つ含まれる．よって count[1]=2，count[0]=1 となり，IDS_1 の sid である 1 を idsOnPath の末尾に追加する．まだ path には id があるため，再び path の先頭の id である 0 を取り出す．IDS_0 の detect には 0 が 1 つ含まれ，count[0]=2 となる．よって IDS_0 の sid である 0 が idsOnPath の末尾に追加される．path にはもう id が含まれていないため，idsOnPath=(1,0) が出力される．

つまり，IDS_1 と IDS_0 がパケットヘッダ情報から分かる経路上に存在する IDS である．

3.6 ネットワーク状態診断機能

ネットワーク状態診断機能は，前節で求めたパケットヘッダによる通信経路上の IDS，つまり理論上アラートが発生するはずの IDS の集合と ACC を構成するアラートを発した IDS，つまり実際にアラートを発生した IDS の集合を比較することで，管理対象ネットワークがどのような攻撃を受けたか，もしくは，管理対象ネットワーク内のどのセキュリティシステムの設定を見直すべきかを診断し，診断結果を出力する．ACC はパケットが通った経路上に存在する IDS から発生したアラートの集合であるため，ACC からは実際にパケットが通った通信経路が分かる．本機能では IDS を比較する際に，IDS がアラートを発したかどうかをビットパターンで表す．この際，サイズが n（n：管理ネットワーク中に存在する IDS の台数）であるリスト変数 find を新たに定義する．sid=i（$0 \leq i \leq n-1$）である IDS からアラートが発生していた場合，find[i]=1，アラートを発していない場合 find[i]=0 というように表す．パケットヘッダによる通信経路上の IDS に対する find は $find_{HDR}$，ACC を構成するアラートを発した IDS に対する find は $find_{ACC}$ とする．ネットワーク状態診断機能の動作の手順を以下に示す．

Step1 全 IDS のうち，通信経路生成機能により出力された sid=$i(0 \leq i \leq n-1, n$：IDS の台数) である IDS に対して $find_{HDR}[i]$=1 とし，それ以外の sid=$j(i \neq j, 0 \leq j \leq n-1)$ である IDS に対して $find_{HDR}[j]$=0 とする．

Step2 ACCS から ACC を 1 つ取り出す．Step1 と同様に，全 IDS のうち取り出した ACC 内のアラートを発した sid=i である IDS に対して $find_{ACC}[i]$=1 とし，それ以外の IDS に対して $find_{ACC}[j]$=0 とする．

Step3 Step1 と Step2 で得られた 2 通りの find に対して，$result[k] = find_{HDR}[k] \oplus find_{ACC}[k]$ のように排他的論理和を，$(0 \leq k \leq n-1)$ を満たすすべての k について計算する．result とは演算結果を格納するサイズ n のリストである．

Step4 Step3 の演算結果からネットワーク状態の診断を行う．

Step5 ACCS 内のすべての ACC に対して Step2 と Step3 を実行する．

Step6 以上で求めた診断結果すべてを出力する．

次に，Step4 の診断方法を示す．

3.6.1 IP スプーフィングの診断方法

本機能に入力されるパケットヘッダによる通信経路上の IDS（idsOnPath とする）と ACC を構成するアラートを発した IDS，上記で示したネットワーク状態診断機能の動作中の Step3 で求めた result から IP スプーフィングの診断を行う手順を以下に示す．

Step1 本機能に入力された idsOnPath の末尾の sid を取り出し，これを p とする．result[p]=0 が成り立つか調べる．

- (a) 成り立つ場合，idsOnPath の先頭の sid=$q(q \neq p, 0 \leq q \leq n-1)$ を取得し，Step2 へ進む．
- (b) 成り立たない場合，IP スプーフィングは行われていないと診断結果を出力し診断を終了する．

Step2 result[q]=1 が成り立つか調べる．

- (a) 成立するとき，管理ネットワーク内で IP スプーフィングが行われていると診断する．Step3 へ進む．
- (b) 成り立たないとき，idsOnPath に sid がまだ含まれるか調べる．
  - (i) まだ含まれる場合，idsOnPath の先頭の sid を取得し，これを q として Step2 へ戻る
  - (ii) 含まれない場合，IP スプーフィングは行われていないと診断結果を出力し診断を終了する．

Step3 ACC に含まれるアラートのパケットヘッダ情報が持つ送信元 IP アドレスが偽装に使用した IP アドレス（fakeIP）である．

Step4 リスト spoof を定義する．spoof にはスプーフィング攻撃の際にパケットが通過したノードの id を

	find[0]	find[1]	find[2]	診断結果
HDR	1	1	0	id=2のノードを通過して外部IPアドレスを用いてIPスプーフィング
ACC	0	1	1	
result	**1**	**0**	**1**	

図 7 IP スプーフィングの診断例

	find[0]	find[1]	find[2]	診断結果
HDR	1	0	1	IDS_2の直前でファイアウォールにより遮断
ACC	1	0	0	
result	**0**	**0**	**1**	

図 8 ファイアウォールの設定診断例

格納する．

Step5　result[s] = 1 かつ $\mathrm{find}_{\mathrm{ACC}}[s] = 1$ となるすべての $s(0 \leq s \leq n-1)$ を探索する．

（a） s が見つかるとき，スプーフィング攻撃の際に IDS_s の detect に設定されたノード対の両方を通過しているとし，spoof に IDS_s の detect に設定された 2 つのノードの id を追加する．spoof に含まれるノードを通過して fakeIP を用いて IP スプーフィングが行われたという診断結果を出力し診断を終了する．

（b） s が見つからないとき，fakeIP を用いて IP スプーフィングが行われたという診断結果を出力し診断を終了する．

図 4 を例として，この診断方法について説明する．本機能に入力された ACC には IDS_1 と IDS_2 から発生したアラートが含まれていて，idsOnPath として (0,1) が入力されたときを考える．このとき result を計算すると**図 7** のような結果となる．idsOnPath の末尾の sid=1 を取得する．このとき result[1]=0 が成り立つため，idsOnPath の先頭の sid=0 を取得する．result[0]=1 が成り立つため，本機能は IP スプーフィングが行われていると判断する．fakeIP は外部ネットワークで使用されている a_{10} とする．result[s] = 1 かつ $\mathrm{find}_{\mathrm{ACC}}[s] = 1$ となるような $s = 2$ が求まる．よって，スプーフィング攻撃の際に IDS_2 の detect に設定されたノード，すなわち id=2 のノードを通過していることが分かる．最後に，id=2 のノードを通過して a_{10} を用いて IP スプーフィングが行われたという診断結果を出力する．

3.6.2　ファイアウォールの設定診断方法

本機能に入力されるパケットヘッダによる通信経路上の IDS（idsOnPath とする）と ACC を構成するアラートを発した IDS，上記で示したネットワーク状態診断機能の動作中の Step3 で求めた result からファイアウォールの設定の診断を行う手順を以下に示す．

Step1　本機能に入力された idsOnPath の先頭の sid=$p(0 \leq p \leq n-1)$ を取り出す．

Step2　result[p]=0 が成り立つか調べる．

（a） 成り立つ場合，idsOnPath の要素数を調べる．

（i） 要素数が 0 の場合，ファイアウォールは通信を許可しているという診断結果を出力し，診断を終了する．

（ii） 要素数が 1 以上である場合，idsOnPath の先頭を取り出し，これを p として Step2 へ戻る．

（b） result[p]=1 となる場合，idsOnPath の要素数を調べる．

（i） 要素数が 0 の場合，IDS_p の直前のファイアウォールにより通信を遮断しているという診断結果を出力し，診断を終了する．

（ii） 要素数が 1 以上である場合，idsOnPath の先頭を取り出し，これを q として Step2（c）へ進む．

（c） result[q]=1 が成り立つか調べる．

（i） 成り立つ場合，Step2（b）へ戻る．

（ii） result[q]=0 となる場合，何も出力せず，診断を終了する．

図 4 を例として，この診断方法について説明する．本機能に入力された ACC には IDS_0 から発生したアラートのみが含まれていて，idsOnPath として (0,2) が入力されたときを考える．idsOnPath の先頭の sid=0 を取り出すと，result[0]=0 となる．まだ idsOnPath には要素があるため，先頭の sid=2 を取り出す．result[2]=1 となり，idsOnPath の要素数も 0 である．よって IDS_2 の直前のファイアウォールにより通信を遮断しているという診断結果を出力する．

4. プロトタイプシステム

4.1　プロトタイプシステムの概要

本研究は，以下の環境においてプロトタイプシステムを実装した．

オペレーションシステム　Ubuntu 12.04 [8]
プログラミング言語　Java [9]
データベース　MySQL [10]
IDS　Snort 2.9.5.3 [2]

Snort はオープンソース IDS である．以上のもの以外に，Snort から検出されたアラートをデータベースに格納するソフトウェアとして Barnyard2 [11] を使用した．これは，Snort から出力されるアラートファイル（パケットログも含む）を解析し，その結果をデータベースに格納するソフトウェアである．提案システムのうち，キャッシュ作成機能と ACES 生成機能，ACCS 生成機能をプロトタイプシステムで実装した．

図 9　システム起動時のウインドウ

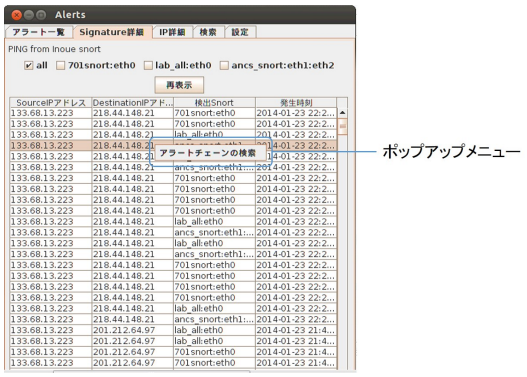

図 10　Signature 詳細ウインドウ

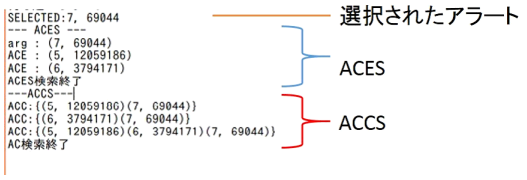

図 11　ACES と ACCS の出力結果

4.2　アラートの閲覧

システム起動時に表示されるウインドウを図 **9** に示す．図 9 の上部には管理している IDS の一覧を表示する．この部分において，ある IDS を選択した後再表示ボタンをクリックすると，選択された IDS から検出されたアラートを表示する．この例では，”all”が選択されているが，このときすべての IDS から発生したアラートを表示していることを示す．次に，アラート表示部を説明する．個の部分ではキャッシュに格納されているアラートをシグネチャ名によるアラートの分類を行い，シグネチャ名とそのアラートが検出された回数，最新の検出日時を表示する．この表においてあるシグネチャ名をクリックすると，クリックされたシグネチャにより発生したアラートに関する情報を Signature 詳細ウインドウで閲覧できる．

Signature 詳細ウインドウでは，上部にアラート一覧タブで選択されたシグネチャ名を表示する．このタブでも，ある IDS を選択した後再表示ボタンをクリックすると，選択された IDS から発生したアラートを表示する．アラート表示部では，アラート一覧タブとは異なりシグネチャ名が等しいアラート情報を 1 つずつ表示する．ここにはアラートを引き起こしたパケットの送信元 IP アドレスと宛先 IP アドレス，アラートを発した IDS 名，検出時刻を表示する．このアラート表示部において，あるアラートをユーザが右クリックすると図 **10** のようにポップアップメニューを表示し，選択されたアラートに関する ACCS を求める．

4.3　ACES・ACCS 生成機能

図 10 でポップアップメニューを選択すると，はじめに ACES を求め，その後自動で ACCS を求める．このときの出力結果を図 **11** に示す．アラートを，sid と cid を用いて（sid, cid）と表現している． はじめにユーザに選択されたアラートを表示し，その後このアラートに関する ACES を表示する．その後，求めた ACES を用いて ACCS を求める．ACES の下に，1 行ずつ ACC を表示する．これらすべてを要素として含んだものが ACCS である．

5.　評価実験

5.1　実験の目的

本実験は，提案システムにおいて，以下の項目が実現できているかを確認するために行う．

目的 1　キャッシュを用いた場合，提案システムの機能を使用する上でメリットがあるか．

目的 2　提案システムで生成した ACCS には本来の AC が含まれているか．

目的 3　ネットワーク状態診断結果は実際のネットワーク状態に当てはまるような正しい診断結果であるか．

5.2　実験方法

前節で述べた目的を実現できているか検証するため，3 つの実験を行う．実験時のネットワーク環境は図 **12** のようになっている．Router1 以下につながるネットワークは，100 台以上の計算機が稼動しており，IDS_6 からは多いと 1 日約 5 万ものアラートが発生する．IDS_5 からは多いと 1 日約 1 万ものアラートが発生する．Hub2 と Hub3 の間では Web サーバやメールサーバなど，あらゆるサーバが稼動している．Hub3 以下ではユーザが実際に触る計算機が多く稼動している．一方，Router2 以下につながるネットワークで稼動している計算機は 10 台以下であり，すべて Hub5 以下に接続されていて，Web サーバ以外のサーバは設置されていない．IDS_7 からは 1 日多くても約 100 くらいのアラートしか発生しない．以下に 3 つの実験方法を示す．

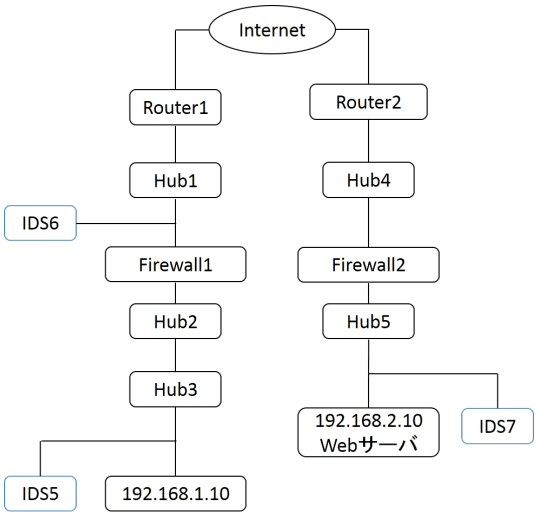

図 12　実験時のネットワーク構成

5.2.1　キャッシュに関する実験

(1) キャッシュのサイズとすべてのアラートやパケットログ情報のデータが格納されているデータベースの合計サイズを比較する．

(2) キャッシュを利用した場合と，すべてのデータベーステーブルを利用した場合で，ACES を生成するためにデータベースを参照する際の実行時間を計測する．ACES の要素数などに偏りがあるため，10 個のアラートに対して ACES を生成し，それぞれの操作でかかった時間を計測する．

5.2.2　ACCS 生成に関する実験

(1) 既存のシグネチャに対し新たにシグネチャを 1 つ追加し，このシグネチャにマッチするようなパケットを送信し，ACCS を生成できるか確認する．このとき作成したシグネチャは

- 192.168.1.10 がいずれかの Web サーバ（宛先ポート番号 80 番）にアクセスしたときアラートを発するシグネチャ

の通りである．既存のシグネチャも存在するのは，ACCS に含まれないようなアラートを発生させ，多くのアラートから ACCS を生成し，生成した集合内に本来の AC があるか確認するためである．

(2) 次に，図 12 中の Firewall2 の設定を，192.168.1.10 が送信元 IP アドレスでありポート番号 80 番に対する tcp パケットを遮断するように変更する．このような状況で，再び以上で示したシグネチャにマッチするパケットを送信し，このパケットに対する ACCS を生成し，生成した集合内に本来の AC があるか確認する．

5.2.3　ネットワーク状態診断に関する実験

実験 2 において生成された ACCS に含まれる ACC を構成するアラートを発した IDS の find と，実験に用いたパケットヘッダにより生成される理論的な通信経路上に存在する IDS の find を比較し，パケットヘッダの偽装があるのか，またファイアウォールによりパケットの遮断が行われているかを提案システムのネットワーク状態診断機能を用いて診断する．この診断結果が実際のネットワーク状態に当てはまるのか確認する．

表 2　各テーブルのサイズ

	レコード数	総容量 (MB)	データ容量 (MB)	インデックス容量 (MB)
event	15217077	1532	606	926
iphdr	15216009	1436	795	641
signature	546	0	0	0
tcphdr	15047145	1622	757	865
udphdr	5288	0	0	0
合計	**15217077**	**4590**	**2158**	**2432**
キャッシュ	**99483**	**11**	**11**	**0**

表 3　ACES を生成した際の速度の比較

	プロトコル番号	ACES の要素数	全 DB 速度 (ms)	提案システム速度 (ms)
A	1	6	75760	383
B	6	10	167765	370
C	6	10	169800	380
D	1	2	76906	375
E	6	0	169558	1089
F	6	0	170446	351
G	254	0	76994	360
H	6	0	227518	532
I	6	0	210087	1003
J	254	1	76417	803

5.3　実験結果と考察

5.3.1　キャッシュに関する実験結果と考察

はじめに，キャッシュのサイズと提案システムに必要なカラムが格納されている各テーブルのサイズを**表 2** に示す．

総容量とは，データ容量とインデックス容量を足した容量である．すべてのアラートやパケットログ情報のデータが格納されているデータベースでは，複数のテーブルに分かれてデータが格納されていたため，データ部分 1 行目から 5 行目までの値の和がすべてのアラートが格納されているデータベースの値となる．すべてのアラートが格納されているデータベースでは，event テーブルのレコード数が実際に発生したアラート数であるため，event テーブルのレコード数を採用する．以上より，すべてのアラートが格納されているデータベースの合計の総容量は表 2 のデータ部分 6 行目より，4590MB である．一方，表 2 においてキャッシュの総容量は表 2 のデータ部分 7 行目より 11MB であることが分かる．直近の最大 10 万件のアラートを用

いて，提案システムに必要なカラムのみを格納してキャッシュを作成しているため，データベースの容量をこれだけ少量にできたと考えられる．

次に，キャッシュを使って ACES を求める場合と，従来のデータベースを使って ACES を求める場合の速度の実験結果について述べる．sid と cid はそれぞれ ACES を生成する際に選択したアラートの sid と cid である．プロトコル番号について，1 が ICMP，6 が TCP，254 は実験やテストに使用されるプロトコルである．**表 3** によると，提案システムのほうがより速く動作し，提案システムでは約 1 秒で ACES を生成できていることが分かる．これは，参照するテーブルの数や，レコード数が少なくなっているためだと考えられる．また，すべてのアラートが格納されているデータベースにおける速度で TCP 以外のプロトコルでは少し早くなっていることも分かる．すべてのアラートが格納されているデータベースでは IP のヘッダ情報や TCP のヘッダ情報などがそれぞれ別のデータベーステーブルに格納されている．そのため，アラートを発生させる要因となったパケットが TCP パケットである場合は IP ヘッダと TCP ヘッダが格納されているテーブルを参照しなければならないが，ICMP やプロトコル番号が 254 であるプロトコルだと IP ヘッダが格納されているテーブルのみを参照すればいい．そのため TCP 以外のプロトコルでは少し早くなっているのだと考えられる．

5.3.2 ACCS 生成に関する実験結果と考察

まず，パケットを遮断せず 5.2.2 節で示したシグネチャにマッチするパケットを送信するために，図 12 中の 192.168.1.10 から 192.168.2.10 の Web サーバにアクセスする．このとき，ブラウザ上から Web サーバに接続できることが確認できた．図 12 より，この通信の経路上で検知を行っている IDS は sid=5 である IDS_5，sid=6 である IDS_6，sid=7 である IDS_7 の 3 つである．ここで，提案システム上で ACCS を生成する．ACCS 生成結果を**図 13** に示す．IDS_5 から発生し，宛先が Web サーバの IP アドレス（192.168.2.10）であるアラートを選択すると，図 13 のように IDS_6，IDS_7 から発生したアラートが ACES として取得でき，ACCS を生成した．ACCS には，ACC が 11 個含まれていることが分かる．本実験では，192.168.1.10 の IP アドレスを持つ端末から 192.168.2.10 の Web サーバへアクセスしているが，アクセスする際に IDS_5，IDS_6，IDS_7 の 3 つの検知場所を通過している．つまり，AC に含まれるアラートを発生させた IDS は 3 個存在するはずである．よって，図 13 より，本来の AC は {(6, 3794097), (7,69041), (5, 12058446)}, {(6, 3794097), (7, 69039), (5, 12058446)}, {(6, 3794097), (7, 69040), (5, 12058446)}, {(6, 3794108), (7, 69041), (5, 12058446)}, {(6, 3794108), (7, 69039), (5, 12058446)}, {(6, 3794108), (7, 69040), (5, 12058446)} の 6 個の内のどれかである．しかし，生成された ACCS に意図して引き起こしたアラートではなくヘッダ等が等しいような他の要因により発生したアラートも存在すると考えられるため，生成された ACCS に本来の AC だけでなく多くの ACC が生成されてしまったことが，改善すべき点だと考えられる．

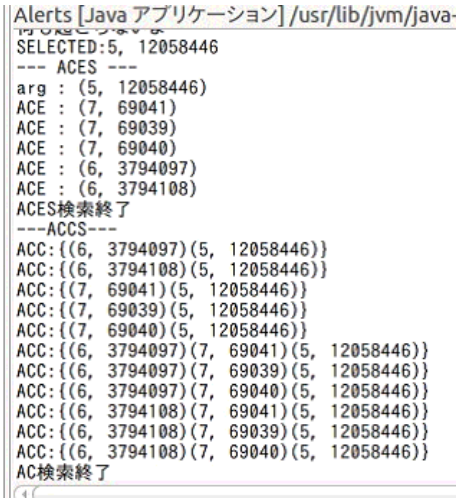

図 13 パケットを遮断しない場合の ACCS 生成結果

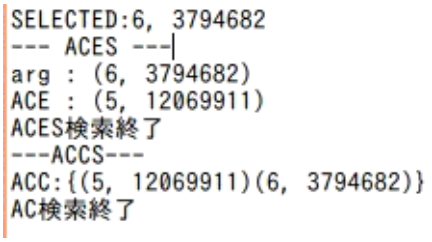

図 14 パケットを遮断した場合の ACCS 生成結果

次に，図 12 中の Firewall2 を，192.168.1.10 が送信元 IP アドレスでありポート番号 80 番に対する tcp パケットを遮断するように変更し，再び先ほどと同様に，Web サーバへアクセスを試みた．しかし，Firewall2 によりアクセスが遮断され，Web サーバへアクセスすることはできなかった．このときの ACCS 生成結果を**図 14** に示す．IDS_7 から発生するアラートが含まれないような ACC を取得し，図 14 のような ACCS が生成された．ACCS には 1 つの ACC のみ存在し，この ACC には図 12 中の IDS_7 から発生するアラートは含まれていない．これは，IDS_7 は本実験のアクセス対象である Web サーバと Hub5 間の通信を監視していて，Firewall2 で遮断し Web サーバに到達しないようなパケットを IDS_7 は検知できないためである．つまり，本来の AC に IDS_7 から発生するアラートは含まれず，本実験で生成した ACCS には 1 つの ACC しか存在しないことから，生成された ACC は本来の AC であると考えられる．

5.3.3 ネットワーク状態診断に関する実験結果と考察

以上で求めた ACCS から，ACC を取り出し，ACC を構成するアラートを発した IDS の find と，パケットヘッダから分かる通信経路上に存在する IDS の find の排他的論

	find[5]	find[6]	find[7]	診断結果
HDR	1	1	1	-----
ACC1	1	1	0	IDS7の直前のファイアウォールで通信をドロップ
result1	**0**	**0**	**1**	
ACC2	0	1	1	IDS6の監視場所を通過して192.168.1.10を用いてIPスプーフィング
result2	**1**	**0**	**0**	
ACC3	1	1	1	全Firewall通信許可
result3	**0**	**0**	**0**	

図 15 パケットを遮断しない場合の診断の様子

	find[5]	find[6]	find[7]	診断結果
HDR	1	1	1	-----
ACC1	1	1	0	IDS7の直前のファイアウォールで通信をドロップ
result1	**0**	**0**	**1**	

図 16 パケットを遮断した場合の診断の様子

理和を計算し，result を求めることで診断を行う．

まず，パケットをファイアウォールで遮断していない場合の診断結果を**図 15** に示す．ACC1 に対しての診断結果はヘッダから求めた経路の終点の result1[7] が 1 となるため，ヘッダから分かる経路上で IDS_7 の直前に通るファイアウォールがパケットを遮断していると分かる．本実験では IDS_7 の直前に通る Firewall2 により，192.168.1.10 から Web サーバへのアクセスを遮断していないため，この診断結果は実際にネットワーク上に起きている問題とは異なる結果だと言える．

ACC2 に対しての診断結果はヘッダから分かる経路の始点に最も近い位置を監視する result2[5] が 1 となる．よって，ヘッダから分かる経路の始点となるノードの IP アドレスを用いて，IDS_6 が監視する場所すなわち，192.168.2.10 と通信する上で Hub1 と Firewall1 を通るような位置に存在する計算機から IP スプーフィング攻撃を行ったと診断される．この診断結果も，本実験では IP スプーフィングを行い Web サーバへアクセスしていないため，実際にネットワーク上に起きている問題とは異なる結果が出力されたと言える．本来の AC である ACC3 に対しての診断結果は，result3[5]，result3[6]，result3[7] のすべてが 0 となる．つまり，すべてのファイアウォールはこのパケットを遮断していないと診断できる．よって，ネットワーク状態診断機能から本実験環境に合致した結果が出力されたと言える．

次に，Firewall2 においてパケットを遮断した場合の診断結果を**図 16** に示す．ACCS に含まれるのは本来の AC1 個のみであるため，診断結果は 1 通りのみの出力である．これに対しての result1[7] が 1 となる．つまり，ヘッダから求めた経路上で IDS_7 の直前に通る Firewall2 がパケットを遮断していると診断できる．本実験では Firewall2 で 192.168.1.10 から Web サーバへのアクセスを遮断した上で，192.168.1.10 から Web サーバへのアクセスを試みた．もちろんこのとき，192.168.1.10 から Web サーバへのアクセスを行うことはできなかったことが確認できた．この実験結果とネットワーク状態診断機能の診断結果と合致していることから，この場合におけるネットワーク状態診断機能は正しく診断結果を出力できていることが分かる．

6. おわりに

本稿では，複数の IDS を用いたアラートの発生パターン比較によるネットワーク診断システムを提案した．提案システムの今後の課題としてネットワーク状態診断機能において本稿で取り上げた以外の診断方法を見つける，発生したアラートが誤検知によるものでないと保証できるか，などが挙げられる．

謝辞 本研究は JSPS 科研費 23500084 および 24500037 の助成を受けたものです．この場をお借りして感謝いたします．

参考文献

[1] 日吉 龍：不正侵入検知 [IDS] 入門，株式会社技術評論社 (2004).
[2] Snort :: Home Page，入手先 ⟨http://www.snort.org/⟩（2014．05．13）.
[3] Vigna Giovanni, Kemmerer Richard A.: “NetSTAT: A network-based intrusion detection system.”, Journal of Computer Security, Vol. 7 Issue 1, pp.37-71(1999).
[4] 王 静，新田 直也，関 浩之：ネットワークの安全性を保証する分散型侵入検知システムの自動構成法，電子情報通信学会技術研究報告，KBSE，知能ソフトウェア工学 103（484），25-29，（2003）.
[5] 花岡 美幸，河野 健二，廣津 登志夫：協調型ネットワーク侵入検知システム Brownie の提案と評価，情報処理学会研究報告，計算機アーキテクチャ研究会報告（2009）.
[6] 竹森 敬祐，三宅 優，中尾 康二：IDS ログ分析支援システムの提案，情報処理学会研究報告，CSEC，[コンピュータセキュリティ]（2003）.
[7] Protocol Numbers，入手先 ⟨http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml⟩（2014．05．13）.
[8] Ubuntu Japanese Team，入手先 ⟨http://www.ubuntulinux.jp/⟩（2014．05．13）.
[9] Oracle and Java — Technologies，入手先 ⟨http://www.oracle.com/jp/technologies/java/overview/index.html⟩（2014．05．13）.
[10] MySQL :: 世界でもっとも普及している、オープン ソース データベース，入手先 ⟨http://www-jp.mysql.com/⟩（2014．05．13）.
[11] firnsy/barnyard2 GitHub，入手先 ⟨https://github.com/firnsy/barnyard2⟩（2014．05．13）.