

# ネットワーク情報を用いたIDSのシグネチャ構築手法

佐藤 淳史

千葉大学大学院自然科学研究科

今泉 貴史

千葉大学総合メディア基盤センター

不正アクセスを検知するためのシステムとしてIDSがあるが、日々新しくなる不正アクセス手法に対応するためのメンテナンスや、False Positiveの多発がネットワーク管理者の大きな負担となっている。本研究では、ネットワークポリシーやセキュリティポリシーなどの情報をもとに、IDSのシグネチャをネットワークに適したものに書き換える手法を提案する。管理者が記述したネットワークの情報に従って書き換えたシグネチャを使用することで、False Positiveの発生自体を抑えることができる。本稿では、シグネチャ書き換えに利用するネットワーク情報の記述方法を議論し、実際にSnortのシグネチャに本手法を適用した場合の記述量や、誤検知数の変化について考察する。

## A Signature Optimizing Method using Information of a Network

SATO Junji

Graduate School of Science and Technology,  
Chiba University

IMAIZUMI Takashi

Institute of Media and Information Technology,  
Chiba University

A signature-based IDS reports a lot of False Positive alerts because of improper signatures. It is dependent on policies of the network whether a reported alert is False Positive or not. In this report, we present a method to eliminate False Positive alerts. To optimize signatures, we use information about the network (i.e. Network Topology and Security Policy). Using this system, you can apply optimized signatures to IDSs, so the IDSs report less False Positive alerts.

## 1 はじめに

インターネットが身近な存在になった今日、不正アクセスなどのネットワーク上の脅威に対する防衛手段が必要となった。不正アクセスなどの侵入検知にはIDSが有効だが、新しい攻撃に対応するためのメンテナンスや、不適切な設定が引き起こす誤検知の多発は、ネットワーク管理者の大きな負担となっている。

誤検知の発生を抑えるには、IDSのシグネチャを適切にチューニングする必要がある。しかし、あるアラートが誤検知かどうかは、ネットワーク構成や運用ポリシーに依存する。したがって、シグネチャのチューニング方法はネットワークごとに異なる。そこで本研究では、IDSを運用するネットワークの情報をもとにして、汎用的なシグネチャをそのネットワークに適したものに書き換える手法を提案する。これにより、False Positiveの発生を抑えることができ、ネットワーク管理者の負担が軽減される。

## 2 IDSの現状

本章では、IDSで問題となる誤検知について説明し、非常に曖昧な存在であるFalse Positiveを、本研究の観点から再定義する。

### 2.1 IDSの誤検知

IDS (Intrusion Detection System) は、バッファオーバーフロー攻撃などの不正な侵入行為を検知するシステムである。本研究で対象とするネットワーク型IDSは、ネットワーク上を流れるパケットを監視し、シグネチャ（あらかじめ登録してある攻撃パターンのデータベース）と一致するパケットを見つけると警告を発する。基本的には単純なパターンマッチングを行うため、しばしば誤検知が発生する。

一般に、IDSの誤検知は次の二種類に分類できる。

- False Negative  
不正アクセスを見逃してしまう誤検知

- False Positive

正常なアクセスを不正と判断してしまう誤検知

False Positive と False Negative はトレードオフの関係にあり、False Negative の発生を避けようとする と False Positive が増加する傾向にある。

## 2.2 False Positive の定義

False Positive の一般的な定義は、2.1 節で述べたとおりである。しかし実際には、どのようなアラートが False Positive であるかは一概に定めることができない。あるネットワークでは False Positive であるアラートが、別のネットワークでは重要な警告となる可能性があるためである。

本研究では、ネットワークのポリシーに反して発生したアラートを False Positive と定義する。つまり、どのようなアラートであっても、それが不要であるというポリシーの下で発生したものは False Positive である。極端な例では、DoS 攻撃に関するアラートは不要としているポリシーの下では、DoS 攻撃を受けて損害が生じたとしても、そのとき発生したアラートは False Positive であると言える。

このように、アラートが False Positive であるかどうかは、ネットワークの運用ポリシーに依存する。そのため、False Positive を削減するためには、ネットワークのポリシーを明確に定義する必要がある。

## 3 システムの設計

### 3.1 システムの動作

本システムでは、汎用的なシグネチャからネットワークのポリシーに沿ったシグネチャを生成する。本システムの概略図を図 1 に示す。

本システムの入力は、ネットワークトポロジーとセキュリティポリシー（あわせてネットワーク情報と呼ぶ）、汎用的なシグネチャの 3 つである。ネットワーク情報を記述したファイルは、ネットワーク管理者が IDS の運用前に用意する。

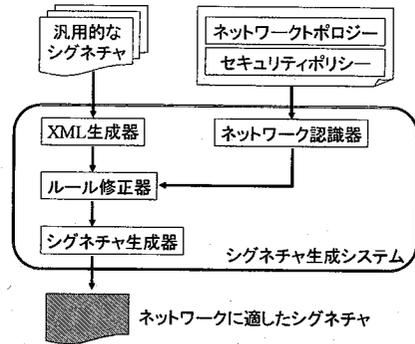


図 1: フレームワーク概略図

### 3.2 ネットワーク情報の記述

本システムでは、ネットワークに適したシグネチャを構築するために、ネットワークの特徴を示す情報を利用する。ネットワーク情報は、ネットワークトポロジーとセキュリティポリシーから構成される。

ネットワーク情報の記述形式については、アクセスリスト生成システムの記述形式を拡張する形で設計した [1]。このシステムでは、ネットワーク情報をルータのアクセスリスト生成に利用している。

#### 3.2.1 ネットワークトポロジーの記述

ネットワークトポロジーは、ネットワークの構造を記述するもので、ネットワークセクションとルータセクションから構成される。ルータセクションは本研究では利用していないので、説明は省略する。ネットワークトポロジーの記述例を図 2 に示す。

ネットワークセクションでは、ネットワークやホストに対して、任意のエイリアスと IP アドレスを記述する。図 2 の例では、*outside*, *dmz*, *inside* の 3 つのネットワークが記述されている。ネットワーク内に別のネットワークを持つことも可能である。また、複数のホストをグループにまとめて扱うことも可能である。図 2 では、*www1* を *HTTP\_SERVERS* というグループに所属させている。

```

/* ネットワークセクション */
network {
  outside {
    IPv4 0.0.0.0/0; }
  dmz {
    IPv4 192.168.100.0 / 24;
    IDS 192.168.100.200;
    www1(HTTP_SERVERS) 192.168.100.1; }
  inside {
    IPv4 192.168.0.0 / 24;
    hostA 192.168.0.1; }
}

```

図 2: ネットワークトポロジーの記述例

### 3.2.2 セキュリティポリシーの記述

セキュリティポリシーは、実現したいポリシーの内容を記述するもので、ルールセクションとサービスセクションから構成される。セキュリティポリシーの記述例を図 3 に示す。

ルールセクションでは、通信の許可・不許可を指定する。ルールの構成要素は、通信の許可・不許可、通信の種類、送信元と送信先それぞれの IP アドレスとポートがある。通信の種類には、次に述べるサービスセクションで定義されているサービスのみ指定できる。IP アドレスには、前述のネットワークセクションで記述したエイリアスやグループ名を利用できる。ただし、ルール間でアドレスの競合が生じた場合はエラーとなる。

block ルールの場合、記述された通信に対してアラートを発するようなシグネチャを作成する。ただし、「dont\_care」オプションが指定されている場合には、シグネチャを作成しない。allow ルールの場合、許可されている通信なので基本的にはシグネチャを生成しない。ただし、「check\_source」オプションを指定した場合は、送信元アドレスを監視対象にする。たとえば、図 3 の 8 行目のルールの場合、SMTP\_SERVERS に対する smtp 通信が hostA 以外から行われた場合にアラートを発生させる。同様に、「check\_service」「check\_destination」オプションはそれぞれ通信サービス、送信先アドレスを監視対象とする。

IDS のシグネチャの中でも特に割合の多い Web 通信を柔軟に監視するために、ホスト名の後ろに括弧付きで URI を指定することもできる。この場合は、

```

/* ルールセクション */
default block;
allow any inside -> any;
allow http outside -> HTTP_SERVERS;
block http outside -> www1(test.cgi);
block http hostA -> HTTP_SERVERS
  {dont_care};
allow smtp hostA -> SMTP_SERVERS
  {check_source};

/* サービスセクション */
service {
  http {
    tcp any -> 80;
    check(widely); }
  telnet {
    tcp any -> 23;
    check; }
  smtp {
    tcp any -> 25; }
  ...}

```

図 3: セキュリティポリシーの記述例

指定された URI に対する通信に限りアラートを発生するシグネチャを作成する。

サービスセクションでは、ルールセクションで記述するサービス内容について具体的なポート番号と方向を記述する。サービスに対して「check」指定がある場合は、そのサービスに対するシグネチャを有効にする。さらに、widely オプションがある場合は、そのサービスの監視対象アドレスを特定のサーバに限定せずに、広く監視を行う。たとえば、HTTP 通信に widely 指定があった場合、悪質な HTTP 通信が HTTP サーバ以外に対して行われてもアラートを発するようにする。

### 3.3 ネットワークに適したシグネチャ

snort.org で配布されている汎用的なシグネチャは、ftp.rules のように攻撃の種類別にファイルに分類されており、2004 年 8 月 17 日現在、48 ファイル 2464 ルールで構成されている [2]。各ルールのアドレス部は、どのようなネットワークでもそのまま利用できるように、すべてのアドレスを意味する any を用いるなど、アドレス範囲が広がっている。

この状態では、ネットワークで提供していないサー

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21
(msg:"FTP CEL overflow attempt"; ...;)
```

```
alert tcp $EXTERNAL_NET any -> $FTP_SERVERS
21 (msg:"FTP CEL overflow attempt"; ...;)
```

```
alert tcp $EXTERNAL_NET any -> $www1 80
(msg:"local 001"; uricontent:"test.cgi");
alert tcp !$hostA any -> $SMTP_SERVERS 25
(msg:"local 002");
```

図 5: ルールの作成例

図 4: (上) 汎用的なシグネチャ

(下) ネットワークに適したシグネチャ

ビスに関するアラートや、サーバ以外のアドレスに関するアラートが発生しやすい。また、これらのアラートは False Positive となる可能性が高い。そこで、不要なサービスに関するシグネチャを無効にしたり、ルールにサーバの具体的なアドレスを指定したりすることで、False Positive を削減できる。ネットワークに適したシグネチャの例を図 4 に示す。

## 4 システムの実現

本研究では、オープンソースの IDS として広く利用されている Snort(2.1.3) を対象 IDS として実装を行った。開発言語には Java を用いた。

### 4.1 システム構成

本システムは、シグネチャを中間表現である XML 形式に変換し、ネットワーク情報をもとにルールを修正する。その後、シグネチャを XML 形式から IDS で使用できる形式に変換して出力する。

図 1 において、XML 生成器は、IDS のシグネチャを XML 形式の中間表現に変換する。ネットワーク認識器は、ネットワーク情報を読み取り、シグネチャの書き換えに必要な情報の整理を行う。また、ネットワーク情報の記述誤りや、ルールの矛盾などのエラーチェックも行う。ルール修正器は、読み込んだネットワーク情報をもとに、XML 生成器が出力したルールに変更を加える。シグネチャ生成器は、ルール修正器が出力したシグネチャを、XML 形式から実際に IDS で使用できる形式に変換する。

### 4.2 ルールの修正

ネットワーク情報をもとにしたシグネチャの書き換えとして、次に挙げるの 3 つの作業を行う。

#### 4.2.1 不要なシグネチャを無効にする

サービスセクションにおいて check オプションがないサービスの監視を無効にする。

たとえば、Web サーバのみが存在するネットワークを考えた場合、FTP 通信に関するアラートは、ネットワークの運用ポリシーによっては False Positive となる。そこで、サービスセクションの HTTP のみに check オプションをつけることで、不要なシグネチャを無効にし、False Positive を削減できる。

#### 4.2.2 エイリアスの範囲を狭める

汎用的なシグネチャは、監視対象となるアドレス範囲が広いルールで構成されているため、本来関係ないアドレスに関するアラートなどは、False Positive となる可能性がある。

そこで、サービスセクションにおいて check オプションに widely 指定がない場合、そのサービスに関するルールのアドレス部を、そのサービスを提供するサーバのアドレスに限定する。たとえば図 3 の場合、HTTP 通信に関するルールのアドレス部を、any から HTTP サーバ群を表す HTTP\_SERVERS に狭める。修正すべきルールかどうかは、シグネチャのファイル名から判断する。HTTP 通信ならば web-attacks.rules などが当てはまる。

#### 4.2.3 ポリシーに沿ったルールを新規に生成

セキュリティポリシーのルールセクションの記述に沿ったローカルルールを作成することにより、False Negative の発生を避ける。ルール作成のアルゴリズムは、3.2.2 小節で述べたとおりである。図 3 の例で作成されるルールを図 5 に示す。

1 つめのルールは、外部から www1 の test.cgi への HTTP 通信を監視するもので、2 つめのルールは、hostA 以外から SMTP\_SERVERS への SMTP 通信

表 1: 記述量の比較

|     | 汎用   |     | 本手法  |     |
|-----|------|-----|------|-----|
|     | 変更箇所 | 文字数 | 変更箇所 | 文字数 |
| (1) | 29   | 29  | -    | 600 |
| (2) | 1    | 24  |      |     |
| (3) | 1    | 125 |      |     |
| (4) | 1    | 1   | 1    | 6   |
| (5) | 18   | 356 |      |     |
| (6) | 47   | 408 | 0    | 0   |

を監視するものである。

## 5 考察

図 2、図 3 のネットワーク情報を持つネットワークについて、汎用的なシグネチャを使用した場合と、本手法を適用したシグネチャを使用した場合について、記述量や誤検知数の比較を行う。

### 5.1 記述量の比較

Snort の典型的な初期設定および、設定変更の例として、以下に挙げる 6 つの作業を考える。

Snort 運用前に以下の初期設定を行う。

- (1) HTTP 通信以外を監視対象から除外
- (2) HTTP に関するルールのアドレス部に HTTP サーバのアドレスを指定
- (3) 図 3 のポリシーに沿ったローカルルールを作成  
Snort 運用後に、以下の設定変更を行う。
- (4) DNS 通信を監視対象に追加
- (5) DNS に関するルールのアドレス部に DNS サーバのアドレスを指定
- (6) 以上のポリシーを維持したまま、Snort.org が更新した新しいシグネチャにアップデートする

記述量の比較を表 1 に示す。本手法を使用した場合、IDS 導入前にネットワーク情報の記述が必要になる。そのため、(1)~(3) では記述量が比較的多くなっている。しかし、一度ネットワーク情報を記述してしまえば、設定変更時の作業量を極力抑えること

表 2: 誤検知数の比較

|         | 汎用    | 本手法   |
|---------|-------|-------|
| (1)FN   | 5     | 5     |
| (2)FP   | 5684  | 0     |
| 総警告数    | 44524 | 38840 |
| 検知した攻撃数 | 1     | 1     |

ができる。特にシグネチャのアップデートは、新しいシグネチャを本システムに適用するだけで、ネットワーク情報に沿ってローカライズされたシグネチャを得ることができる。

また、記述したネットワーク情報は、ファイアウォールやルータのアクセスリスト生成にも容易に流用できるため、ネットワーク全体のポリシー管理としても有効性が高い。

サービスセクションはネットワークトポロジーへの依存が弱いので、あらかじめ一般的なサービスについて記述したファイルを用意できる。また、ルールセクションに関しても、`HTTP_SERVERS` などのグループ名を用いて記述すれば、ネットワークトポロジーへ依存せずにルールを用意できる。そのため、あらかじめ推奨ルールを用意することも可能である。

### 5.2 誤検知数の比較

MIT Lincoln Labs の 1998 年 3 週目水曜日のデータセットを利用して、汎用的なシグネチャと本手法を適用したシグネチャについて、以下の誤検知の数を比較する [3]。

- (1) MIT Lincoln Labs の定義による False Negative
- (2) False Positive の数

誤検知であるかどうかは、Lincoln Labs による攻撃の定義と、図 2,3 のネットワーク情報をもとに判断する。実験結果を表 2 に示す。

(1) について、汎用的なシグネチャで False Negative が発生しているの、本手法においても同様の False Negative が発生している。本手法を適用することで False Negative が増加した場合には、必要であればネットワーク情報を書き直すことで、False Negative を発生させないシグネチャを作成できる。

(2) について、汎用的なシグネチャでは全部で 44524 個のアラートが発生し、そのうち 5684 個は、図 2,3 のネットワーク情報で不要とされた警告であった。それに対して本手法を適用したシグネチャでは、必要なアラートを残して False Positive を削減できていることがわかる。

### 5.3 関連研究との比較

False Positive を削減するための研究の多くは、学習によるアラートのフィルタリングといった事後処理である。宮地らの研究では、機械学習によって False Positive の特徴を学習し、ニューラルネットワークを用いて False Positive のパターンを識別している [4]。別のアプローチとして、阿部らの研究では、システムコール呼び出し時のスタックを検査し、バイナリコードが規定する制御フローとプログラムがたどる制御フローを比較することで、False Positive を出さずに異常を検知するシステムを提案している [5]。本研究では、ネットワーク情報を記述することでその組織での False Positive を明確に定義し、ポリシーに沿ってシグネチャを修正することで、誤検知の発生自体を抑えている。また、ネットワーク情報を導入することで、設定変更やアップデートなどのメンテナンスも容易になる。ネットワーク情報は、ファイアウォールやルータなど、ネットワーク全体のセキュリティ管理にも応用できる。

Snort のシグネチャを自動更新するソフトウェアとしては、oinkmaster がある。oinkmaster は、自分で変更を加えたルールファイルは更新しないといったローカル設定の保持が可能であるが、これではルールの修正や追加、削除に対応しきれない。それに対して本手法では、更新の際にすべてのルールを修正しなおす。そのため、最新のルールセット全体に対してポリシーを適合したシグネチャを得ることができる。

ポリシー記述の標準化の動きとしては、IETF における PCIM および PCIme が挙げられる [6][7]。これらは大規模システム、分散システムなどの全体的なポリシー管理において有効である。本研究で提案するセキュリティポリシーの記述言語は、IDS やフ

ィアウォールなどのネットワークセキュリティに特化したもので、可読性が高く、特別な知識を必要としない簡単な記述形式になっている。

## 6 まとめ

本稿では、誤検知削減のためにポリシーを明確にすることの必要性を述べた。また、ポリシーを記述したネットワーク情報をもとに IDS のシグネチャを書き換える手法を提案し、評価を行った。

本稿で提案した手法で、管理者が思い描くポリシーをすべて記述できるとすると、誤検知は発生しないことになる。それでも誤検知が発生するとするならば、それは IDS 自体が持つ潜在的な問題であると考えられる。

今後の課題としては、ポリシーをより柔軟に記述できるように、文法を拡張する必要がある。また、Snort のシグネチャには exploit.rules といった、特定のサービスに依存しないものもある。そのため、このようなシグネチャに対するネットワーク情報の記述方法を考える必要がある。

## 参考文献

- [1] 谷津文平, 今泉貴史, セキュアな IPv6 ネットワーク構築のためのアクセスリスト生成システム. 情報科学技術フォーラム 2002 講演論文集, L-9 19-20, 2002.
- [2] snort.org, <http://www.snort.org/>
- [3] MIT Lincoln Laboratory - DARPA Intrusion Detection Evaluation, <http://www.ll.mit.edu/IST/ideval/>
- [4] 宮地玲奈, 小宅宏明, 川口信隆, 重野寛, 岡田謙一, 機械学習によるネットワーク型 IDS の false positive 削減手法の提案. 情報処理学会 CSEC 研究会, 2003-CSEC-21, pp.53-58, 2003.
- [5] 阿部洋丈, 大山恵弘, 岡瑞起, 加藤和彦, 静的解析に基づく侵入検知システムの最適化. 情報処理学会, コンピューティングシステム, Vol. 45, pp.11-20, 2004.
- [6] RFC3060, Policy Core Information Model - Version 1 Specification. <http://www.ietf.org/rfc/rfc3060.txt>, 2001.
- [7] RFC3460, Policy Core Information Model (PCIM) Extensions. <http://www.ietf.org/rfc/rfc3460.txt>, 2003.