社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2009－DPS－138 (28)
2009－CSEC－ 44 (28)
2009／3／6

AIDにおける異常フロー発信元特定方式

河内清人 北澤繁樹 榊原裕之 藤井誠司

三菱電機株式会社 情報技術総合研究所

あらまし 異常検知型のネットワーク侵入検知技術(AID)によって企業内部ネットワークで発生する機密情報の不正なアップロード等を監視する場合、異常検出した後に原因となっている端末の隔離等、即座に対策を打つために異常なトラフィックを発生させている端末が正確に特定できる必要がある。端末数が少数ならば、単純に端末からの通信を個別に異常監視することも可能であるが、ネットワーク規模が大きくなり、端末数が増大するにつれ、このアプローチは計算リソース上困難になる。本稿では上記課題に着目し、観測対象をグループ化して監視対象を削減しつつ異常の発生した端末を特定する方式を提案する。実在するネットワークから収集されたトラフィックデータをもとに評価を行ったところ90%を超える精度で端末の特定に成功した。

A Method for Identifying a Source of Anomalous Flow in Anomaly Intrusion Detection

Kiyoto Kawauchi Shigeki Kitazawa Hiroyuki Sakakibara Seiji Fujii

Information Technology R&D Center , Mitsubishi Electric Corporation

Abstract When using anomaly detection techniques for monitoring enterprise network, it must not only detect anomalous network events, but also identify the cause of them, especially their source terminal in order to react them rapidly. However, it costs too much computational resource to monitor network traffic occurred from each terminal individually. In this paper, we propose a method for solve this challenge. We evaluated our method by a simulation using real traffic data, it resulted more than 90% identification accuracy.

1. はじめに

代表的な不正アクセスの検知方法にSignature-based Network IDS(S-NIDS)がある。S-NIDSでは、ネットワーク上のパケットを監視し、既に知られている脆弱性を突く攻撃データのパターン(シグネチャ)と合致する内容を含んだパケットを検出することで攻撃を検出する。

近年、犠牲者を悪意のWebサイトに巧みに誘導し、不正な実行ファイル(マルウェア)をPC上で実行させることでそのPCを乗っ取り、機密情報の漏洩等を行わせる受動攻撃と呼ばれる攻撃が増加しつつある。このような攻撃においては、脆弱性を攻撃するようなパケットは発生しないため、S-NIDSでは攻撃を検出することは困難である。

一方で、S-NIDSのように攻撃とみなすパターンと合致するパケットを検出するのでは無く、平常時のネットワークトラフィックの特徴を

学習し、そこから統計的に逸脱したトラフィックが発生したときに攻撃を受けたことを検出する異常検知型のネットワーク侵入検知技術(Anomaly-based Network IDS: A-NIDS)も数多く提案されてきている。今後 S-NIDS を用いた監視を補完する形でこれらを適用することで、受動攻撃のような攻撃の発生も検知できることが期待できる。

内部ネットワークを A-NIDS で監視する場合、単に異常を検出するだけでなく、異常検出後に原因となっている端末の隔離等、即座に対策が打てなければならない。そのためには、異常なトラフィックを発生している端末を正確に特定できる必要がある。

しかし、監視対象とすべき端末は数千台に上る可能性がある。A-NIDS では監視対象としている通信それぞれについて学習を行い、学習されたモデルと観測値を比較することで異常の発生を検出する。そのため、数多くの端末から発生する通信を個別に監視するには、計算リソース上の制約から閾値等単純なモデルを使わざるを得ず、監視対象のネットワークが大規模になるほど、検知性能の向上が難しいという課題がある。

本稿では上記課題に着目し、監視対象をグループ化することで個別に監視しなければならない対象を抑えつつ、なおかつ異常発生時に速やかにどの端末からのトラフィックが異常かを特定可能な A-NIDS を提案する。

2. 提案方式

2.1 システム概略

本提案方式は、L3 スイッチ等から容易に収集可能な Netflow[1]を収集し、ネットワーク内で発生した異常なトラフィックを検出する。システムは一定間隔で監視対象のネットワーク上のスイッチから NetFlow を収集し、後述する方法で時系列データ群を生成する。生成された各時系列データは、次に異常判定機能により、過去の傾向と比較され、異常な変動が発生していないかどうかが検査される。最後に各時系列データに対する判定結果を集約し、異常なフローの生成元を特定する。以下に各処理について述べる。

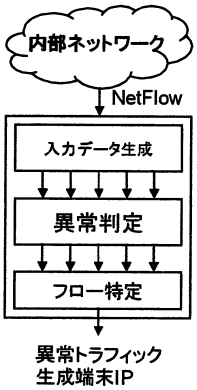

図 2-1 提案システム概略

2.2 入力データの生成

はじめに、集計間隔内に収集された NetFlow 情報(以下フロー情報)群(flow_i [1≦i≦n])に対し、その内容(発信元アドレス等)に基づき m ビット固定長で表現される識別子 id_i が割り当てられる。

フロー情報の内容から識別子をどのように割り当てるかは、侵入検知システムの設計時に決定すべきパラメータとなる。例えば、特定サーバに対してアクセスするクライアント群からの通信を監視し、異常なアクセスを行っているクライアントを識別するのであれば、フロー情報に含まれる発信元 IP アドレスに基づいて識別子を割り当てることが妥当と考えられる。

次にフロー情報群から異常判定機能への入力ベクトル ivec を生成する。ivec は m 個の要素からなる数値ベクトルであり、k(1≦k≦m)番

目の要素は k 番目の異常判定処理への入力となる。

入力ベクトルは図2-2に示す手順で生成される。最初に、入力ベクトルの各要素を0に初期化する。次に各フロー情報を数値化関数 F に入力し、flow_i に含まれる内容から数値 v を算出する。数値化関数としては、例えば flow_i に含まれるデータサイズを返す関数や、最も単純なものとしては、常に1を返す恒等関数、つまり個数を数えるのみの場合も考えられる。

flow_i から v を算出後、ivec の1つ以上の要素に v を加算する。ivec の k 番目の要素に v を加算するかどうかは、flow_i に対応する識別子 id_i を m 桁の2進数で表現した場合に k 番目のビットが1であるかどうかで判定する。k 番目のビットが1ならば k 番目の要素に v を加算する。

以上の処理を全ての flow_i に対して行い、入力ベクトルが生成される。

```
1:  ivec ←[ 0,0, ... 0 ]
2:  for i in 1 .. n
3:     v ← F( flow_i )
4:     id_i ← getId( flow_i )
5:     for k in 1 .. m
6:        if getbit( id, k ) = 1 then
7:           ivec[k] ← ivec[k]+v
8:        end if
9:     end for
10: end for
```

図 2-2 入力ベクトル計算アルゴリズム

2.3 異常判定

前節の処理を経て生成された入力ベクトル ivec の各要素に対し、異常判定を行う。本提案では、過去に入力された数値を時系列データとして学習し、最近の入力値の変動の異常を検出するもの(例えば文献[2])を想定している。各要素の異常判定は、他の要素とは独立に行われる。即ち、ivec の k 番目の要素(ivec[k])について異常判定を行う場合には、過去に入力された ivec[k]によって学習された状態情報を用いて異常判定を実施する。

2.4 発信元の特定

最後に、図2-3に示す手順で異常の発生したフローを特定する。

入力ベクトルの各要素に対して異常判定を行った結果を元に、m 個の要素からなる出力ベクトル ovec を生成する。ovec の k(1≦k≦m) 番目の要素(ovec[k])は、入力ベクトルの要素 ivec[k]に対する異常判定の結果が格納される。判定の結果異常と判断された場合には1、そうでなければ0が格納される。

前述の通り、フロー情報から算出された数値は、識別子のビット表現に従って一つ以上の ivec の要素に加算される。理想的な異常判定アルゴリズムを仮定すると、ある識別子に属するフロー上で異常なトラフィックが発生した場合、そのフローが入力された全ての ivec の要素に対する異常判定で異常が検出されるはずである。従って、ovec の m 個の要素を m 桁の2進数とみなすことで、異常の発生したフローの識別子を特定できる。なお、ovec の要素が全て0、つまり算出された id が0であった場合には異常無しと判定する。

以上のように、フローに固定長ビットの識別子を与え、フロー情報から数値化された値を識別子のビットパターンに従って入力ベクトルに

```
1:id ← 0
2:for i in 1 .. m
3:   if ovec[i] = 1 then
4:        setbit( id, i )
5:   end if
6:end for
```

図 2-3 異常判定結果からのフロー識別

加算し、異常発生時には識別子と同じビットパターンを持つ異常判定結果が出力されるようにすることで、m 回の異常判定で 2^m-1 種類のフローに対する個別の異常検出が可能となる。

2.5 誤検知・検出漏れへの対応

本方式ではフローに異常が発生した場合、そしてその場合に限り、対応する入力ベクトルの要素に対する異常判定結果が 1 とならなければならない。しかし、実際の異常判定では誤検知や検知漏れといった誤判定が発生するため、その場合、誤ったフロー識別子が算出されてしまう。

異常判定における誤判定により識別子の特定を誤るという課題は、データ通信における通信伝送路上におけるビット誤りの問題と類似しているといえる。通信では伝送路符号化において送信したいデータに誤り訂正符号を付加することによって受信側での誤り検出、及び訂正を行う方法が良く知られている。

この冗長ビットを付加するという考えを用いて、本方式では異常判定における誤判定の問題を軽減させる。

まず§2.2 と同様、入力する flow_i に対応する id_i、及び v を求める。次に、id_i を入力として誤り訂正符号を付加したビット列 id'を生成する。この時、id'は誤り訂正ビット数分だけ

```
1:  ivec ←[ 0,0, ... 0 ] /* 要素数 m' */
2:  for i in 1 .. n
3:     v ← F( flow_i )
4:     id ← getId(flow_i)
5:     id' = encode(id)
6:     for k in 1 .. m'
7:        if getbit( id', k ) = 1 then
8:           ivec[k] ← ivec[k]+v
9:        end if
10:    end for
11: end for
```

図 2-4 誤り訂正ビットの追加

id よりも長いビット数(m'ビット)を必要とする。次に、§2.2 と同様 id'及び v に基づいて ivec を生成する。

生成された ivec (要素数 m'個)の各要素に対して、異常判定を実施後、各異常判定の結果から異常なフローを特定する処理が実施されるが、§2.4 とは異なり、ovec から得られるビット列を復号することで、元の id を取り出す。m'回の異常判定のどれかで誤判定が起こったとしても、誤り訂正符号の能力以内であれば、訂正が行われ、正しい id が特定できる。

```
1:id' ← 0
2:for i in 1 .. m'
3:   if ovec[i] = 1 then
4:        setbit( id', i)
5:   end if
6:end for
7:id ← decode( id' )
```

図 2-5 復号処理手順

3. 評価

本提案方式の有効性を検証するために、評価を行った。

3.1 評価条件

評価用の元データとして、某企業内のサブネットワーク上のクライアント PC(IP 数 100)から、DMZ 上にあるプロキシサーバへの通信で生じたフロー情報を 31 日間分採取した。

次に採取したフロー情報を発信元 IP アドレス毎に分類し、時系列データを生成した。時系列データは単位時間(10 分間)あたりに発生したフロー情報の個数を数え上げることで生成した。従って時系列データあたりのデータ点数は 4464 点となる。

このようにして得られた時系列データのうち、一つをランダムに選択し、学習に使用する最初の 1008 点を除き、ランダムな箇所に攻撃を模

擬した時系列データを加算することで、異常の発生した時系列データとした。

なお、攻撃を模擬した時系列データとして、攻撃開始時点から単位時間毎に 20 フローが発生し、それが 48 単位時間継続するモデルを使用した(図 3-1)。

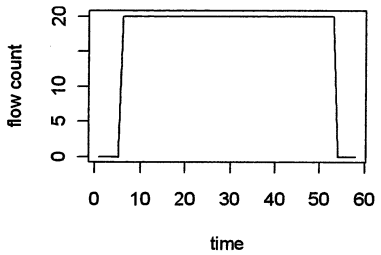

図 3-1 模擬攻撃時系列データ

このようにして得られた時系列データを元に表 3-1 に示した条件で評価実験を 100 回繰り返し行った。従って、総分析回数は(4464-1008) x 100 = 345600 となる。

表 3-1 本評価における条件設定

異常判定方式に関するパラメータ	
アルゴリズム	文献[2]記載の方式
学習期間	1008 単位時間
その他	ウィンドウサイズ: 24 マハラノビス距離閾値: 1.2

本提案方式に関するパラメータ	
識別子割り当て	発信元 IP のドット表記における辞書式出現順序(7bit)
誤り訂正方式	(15,4)ハミング符号[3]

3.2 評価項目

評価項目として、以下を計測することとした。

- 検知率
- 誤検知率
- フロー特定誤り率
- 誤り訂正の発生率、訂正成功率

検知率

本方式の目的は攻撃検出後の異常発生もとの特定であるため、攻撃を挿入したフローを正しく特定できた割合を検知率とした。

検知率=特定成功回数/全攻撃数(=48 x 100)

誤検知率

攻撃が発生していない時刻に誤って異常と判定される事を誤検知とし、その割合を計測した。

誤検知率=誤検知数/(総分析回数-全攻撃数)

フロー特定誤り率

攻撃が発生している時刻に異常を検出したが、誤ったフローを特定した割合を計測した。

フロー特定誤り率=誤特定回数/全攻撃数

誤り訂正発生率、訂正成功率

異常判定の後に実施した誤り判定でビットの修正が発生した率、及び発生した誤り訂正のうち、正しい出力に訂正できた率を計測する。

誤り訂正発生率=誤り訂正回数/総分析回数
訂正成功率=訂正成功回数/誤り訂正回数

3.3 評価結果

今回行った 100 回の試験の結果を表 3-2 に示す。

表 3-2 評価結果

測定項目	測定結果(実測値)
検知率	91.3%(4382/4800)
誤検知率	4.96%(16890/340800)
フロー特定誤り率	7.33%(352/4800)
誤り訂正発生率	5.46%(18855/345600)
誤り訂正成功率	64.3%(12123/18855)

以上のように、本評価においては 90%以上の割合で異常フローの発生元を特定できることが確認できた。

特定誤りが約 7%発生している。特定誤りが発生した場合、個々のフローについて異常かどうか検査する必要があるが、出力された id とハミング距離の近いものから順に検査していくことで、より短時間で真の異常フローを特定できると考えている。

誤検知・検出漏れ対策として導入した誤り訂正

であるが、全分析回数(345600)の 5.5%で訂正が発生しており、そのうち 64%(12123)で有効に機能していることが判った。内訳を精査すると、その 99%以上(12002)が誤検知を異常無しに訂正していた。つまり、誤り訂正が無ければ誤検知率が 8.5%程度まで悪化していたはずであり、本対策を導入した効果は十分にあったと考えられる。

4. 関連研究

A-NIDS に関する研究は従来からも数多く行われている。多くは集積されたデータからいかに鋭敏に異常な変化を検出するかに主眼が置かれている。しかし近年は、異常の発生を検知するだけでなく、対策するために必要な情報を同時に提供する試みがなされてきている。

例えば[4]では、検出された異常がワームや DoS といった既知のインシデントのいずれに近いかを分析する手法について提案している。国内では大河内らが、自己組織化マップ(SOM)を用いて、発生しているインシデントの分類を視覚的に行う手法について検討を行っている[5]。また、竹森らは ISP 間で発生した異常同士の関連を分析して、ワーム等の拡大予測を行う手法を提案している[6]。

本稿も異常検知後の対策に必要な情報を提供することを目的としている点では同様であるが、異常の発生元をホスト単位で特定するための手法を提案したものは筆者の知る限り無く、本方式と前記文献で挙げられているような異常の種別判定などとを組み合わせることで、より効果的な対策実施が可能と考えている。

5. まとめ

異常検知型のネットワーク侵入検知技術(AID)によって企業内部ネットワークを監視する場合、異常検出した後に原因となっている端末の隔離等、即座に対策を打つために異常なトラフィックを発生させている端末が正確に特定できる必要がある。しかし、そのためには各端末からの通信それぞれについて学習を行い、異常かどうか判定しなければならず、数多くの端末から発生する通信を限られた計算リソース内で個別に監視することは困難であった。

本稿では、識別したいフロー情報に固定長ビット数で表現可能な識別子を割り振り、そのビットパターンに応じて構成したグループ単位で異常判定を行うことで、端末からのトラフィックを個別に監視することなく異常トラフィックを発生させている端末を特定する方式について提案を行った。

実企業から採取したフロー情報をもとに評価を行ったところ、本方式により 90%を超える精度で異常の発生したホストの特定に成功した。検出誤りへの対策として誤り訂正を付加したところ、8.5%程度であった誤検知率を 5.0%まで改善することに成功した。

参考文献

[1] Cisco Systems NetFlow Services Export Version 9, RFC 3954, IETF
http://www.ietf.org/rfc/rfc3954.txt
[2] 榊原裕之,河内清人,北澤繁樹,藤井誠司 “正常域データの変動を考慮した不正アクセス分析システム” 21 回 CSS 情報処理学会 2008
[3] 今井 秀樹 “符号理論” コロナ社 ISBN 4-88552-090-8 1990
[4] Salem, B. and Karim, T. . “Context-based Profiling for Anomaly Intrusion Detection with Diagnosis” ARES vol.00. IEEE Computer Society, 2008
[5] 大河内一弥,力武健次,中尾康二 “自己組織化マップを用いたネットワークインシデント分析の研究” SCIS 2006 2e-2-3
[6] Takemori, K.; Miyake, Y.; Ishida, C.; Sasase, I., "A SOC Framework for ISP Federation and Attack Forecast by Learning Propagation Patterns," Intelligence and Security Informatics, IEEE , pp.172-179,2007