通信端点における侵入検知システムのカーネルモジュールを用いた実装手法

松井卓† 前田敦司† 山口喜教†

1. はじめに

ネットワーク経由での攻撃に対処するため，近年，侵入検知システム(IDS)の導入が進んでいる．しかし従来の IDS は，高速なネットワークに対応できない，内部から内部への攻撃は検知不能等の問題が指摘されている．

ここでは IDS の持つ諸問題への対策として，通信端点となるサーバ等に軽量な検知モジュールを導入することの利点を述べるとともに，Linux カーネルモジュールを用いて実装することでプロトコルに寄らない検知が可能になり，またスループットの向上が見込まれるということを示す．

2. 侵入検知システムとその問題点

2.1 ネットワーク型侵入検知システム

ネットワーク型 IDS(NIDS)は侵入や攻撃のパケットが対象のネットワークに流れていないか監視するシステムで，侵入やその兆候を検出した場合，管理者への通知やファイアーウォールと連携してパケットの遮断等を行うものである．一般的に NIDS は，外部ネットワークとの接続点付近に専用のマシンを設置して，外部から来るパケットを検査する．

フリーの NIDS の代表例として Snort[1]がある．これはネットワークを流れるパケットを検査し，攻撃特有の文字列がパケットに含まれていないか検査するものである．この検知方法をシグネチャ型と言い，検知に用いられる検知ルールは 2007/07 時点で約 8300 個ある．

2.2 NIDS の問題点

ネットワーク型 IDS は外部ネットワークとの接続点付近に設置されるという性質上，いくつかの問題を持っている．

TCP/IP の検査をする場合セッションの再構成が必要であり，NIDS はそのために大量のメモリが必要であり，NIDS はそのために大量のメモリ空間を使用する．扱うセッション数が膨大になると再構成に必要なメモリ量も膨大になり，IDS の処理性能を超える通信量が発生した場合，侵入を検知できなくなってしまう．

また検査を行うのはネットワークの接続点付近など局所的であり，内部から内部への攻撃など，IDS を通らない経路で攻撃される可能性が存在する．

3. カーネルモジュールを用いた実装と効果

3.1 概要

IDS の持つ諸問題を解決するために，外部ネットワークとの接続点付近に専用のマシンを設置するのではなく，通信端点となるサーバやクライアントに軽量な検知モジュールを配置することで負荷を分散させる手法を筆者らが提案してきた[3]．通信端点であれば TCP セッションの再構成が行われた後にパケットを取得すればよく，IDS が独自にパケットを再構成しなくても良いので IDS を軽量に実装できる．また内部から内部への攻撃にも対応できる．

本研究では実装にカーネルモジュールを用いることで，バッファに直接アクセスしメモリアクセスのオーバーヘッドを抑えることでパターンマッチングの速度の向上を図るとともに，UDP や ICMP といったユーザプロセスレベルでは効率のよい検知が困難であったプロトコルでの検知が可能になると考えられる．なお本論文では，ユーザプロセスのため筆者らが従来提案していた手法では検知の困難であった UDP や ICMP の検知と性能の向上を目的とする．

3.2 設計と実装

検知モジュールは，パケットのパターンマッチングを行うマッチャーと検知ルールのセットで構成されている．そしてこれらはカーネルモジュールとしてカーネルにロードされる．マッチャーでは検知ルールをメモリ効率のよい DFA 状態遷移表として保持する．今回は Aho-Corasick オートマトンをダブル配列表現で表しレベル 1

Implementation methods for Intrusion Detection Module at Network Endpoint using Kernel modules.

† 筑波大学
University of Tsukuba

エッジを除去したものを用いる．検知ルールから生成された DFA 状態遷移表をあらかじめ C 言語の配列形式で保存しておき，コンパイル時に検知モジュールにリンクする．検知ルールは UDP/IP の場合・HTTP の場合等に分割してカスタマイズすることが可能で，対象となるサーバに必要なルールのみを抽出して読み込むことで，省メモリ化を果たしている．

ロードされると Linux のプロトコル層にフック処理を登録する機能を用いて，IP 層でフックして検知処理を起動するように登録する．フックは NF_IP_LOCAL_IN (自分宛のパケットと判断後で IP 層から上位プロトコルに渡される前)の位置に登録される．マシンに何らかのパケットが到着するとフックされ，検知モジュールにパケットのバッファが渡される．パケットのペイロードをマッチャーに渡し検査を行い，攻撃を検知した場合はデバイスドライバを通して警告を行う．

4. 評価

評価環境は CPU:Pentium 4 (2.20GHz), RAM:512MB, kernel: 2.6.18-8.1.15.el5, OS:CentOS5, GCC 4.1.2 で，コンパイラオプションは-O2 で行った．検知ルールは Snort のルールのうち，UDP と IP に関するルール(670 個)で現れる文字列を使用している．

4.1 スループット

UDP/IP を用いて文字列を送受信するサーバとクライアントをそれぞれ用意し，10 秒間で送信できるデータ量からスループットを算出した．

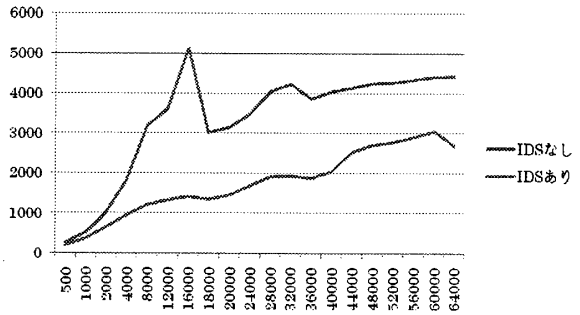

図 1.パケットサイズによるスループットの変化.

図 1 は縦軸が速度(Mbps)，横軸が 1 パケットあたりのパケットのサイズを表している．最初，パケットサイズが大きくなるにつれて差が大きくなっている．これはパケット数が少なくなった分オーバーヘッドが減少し，パターンマッチングの負荷が顕著に表れたためと考えられる．16000byte や 32000byte 付近でのスループットの周期的な悪化はバッファサイズとの関連である．

表 1.スループットと相対速度

パケットのサイズ (byte)	スループット (Mbps)	相対速度
1000	399.99	0.76
60000	3070.71	0.69

表 1 は，図 1 の値を抜き出したものである．1 パケットのサイズを 1000byte にした時の相対速度は 76%であった．なお，パターンマッチングを行わずフックのみの場合の性能の低下は 1％程度である．

検知モジュールのサイズは 180396byte(内検知ルールは 226465byte)であり，現在のサーバであれば十分に軽量と言える．

5. おわりに

本研究では，通信端点において侵入検知を行う軽量な検知モジュールの，カーネルモジュールを用いた実装手法の提案を行った．本手法を用いることで，許容し得るレベルのオーバーヘッドで個々のサーバに検知モジュールを導入できることが示せたと考える．

今後さらに効率を向上させ，オーバーヘッドの削減を目指す．

6. 謝辞

本研究の一部は，科学研究費補助金特定領域研究「情報爆発 IT 基盤」(領域番号 456)「通信端点における分散検知モジュールによる侵入防止機構」(課題番号 19024008)をうけて行われた．

7. 参考文献

[1] M. Roesch: “Snort - lightweight intrusion detection for networks”, Proc. 13th USENIX conf. on System Administration, pp. 229-238 (1999).

[2] A. V. Aho and M. J. Corasick: “Efficient string matching:an aid to bibliographicsearch”, CACM 18(6), pp. 333-340 (1975)

[3] 前田敦司，渡辺祐介，西孝王，山口喜教: “通信端点における軽量侵入検知モジュールの試作” : 信学技報，Vol. 106, No. 436, pp. 13-18 (2006)

[4] 高橋浩和，小田逸郎，山幡為佐久: “Linux カーネル 2.6 解読室” :ソフトバンククリエイティブ(2006)