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

2007－DPS－130 (67)
2007－CSEC－ 36 (67)
2007／3／2

Outbreak型ワームを対象とした被害拡散防止システムの実現

豊国明子　原田道明　北澤繁樹　鈴木清彦　時庭康久

三菱電機（株）情報技術総合研究所
〒247-8501 神奈川県鎌倉市大船 5-1-1
Email: {Toyokuni.Akiko@bk, Harada.Michiaki@eb, Kitazawa.Shigeki@dn,
Suzuki.Kiyohiko@bc, Tokiniwa.Yasuhisa@aj} .MitsubishiElectric.co.jp

あらまし

近年ワームの手口は巧妙になってきており、ワームの侵入を完全に阻止することは非常に難しい。このため、仮にワームに感染しても、被害を最小限に食い止める適切で迅速な措置が必要である。本稿では、爆発的にパケットが増大するタイプのワームを対象に、ワームの被害拡散を早期に防止する方式を提案する。感染活動を抑制しながら、シグネチャを自動的に生成することにより、未知のワームに対しても速やかな対応を可能にする。提案方式のプロトタイプを実装し、即応性を検証する実験結果から明らかになった提案方式の実現可能性と課題について述べる。

Cyber Attack Containment System against the Outbreak of Worms

Akiko Toyokuni　Michiaki Harada　Shigeki Kitazawa
Kiyohiko Suzuki　Yasuhisa Tokiniwa

Information Technology R&D Center, Mitsubishi Electric Corporation
5-1-1 Ofuna, Kamakura, Kanagawa, 247-8501
Email: {Toyokuni.Akiko@bk, Harada.Michiaki@eb, Kitazawa.Shigeki@dn,
Suzuki.Kiyohiko@bc, Tokiniwa.Yasuhisa@aj} .MitsubishiElectric.co.jp

Abstract

As growing sophistication of worms, it is impossible to keep out them. Even if a worm intrudes, quick response and appropriate action that minimize damages should be required. In this paper, we propose a cyber attack containment system for early prevention against worm dissemination that outbreak on the network. This makes it possible to prevent an unknown worm by making infection slow and IPS signatures automatically. We build a prototype and conduct a readiness test. Also, the paper presents feasibility and future issues from the test results.

1. はじめに

これまで様々な方法でワームへの対策がなされてきたが、ワームの手口は巧妙になっており、万全な対策を施したつもりでも、ワームの侵入を完全に防止することは事実上困難である。ワーム対策には、事前の防御だけでなく、ワームに感染した場合に備え、適切かつ早急な事後対応（Incident Response）の確立が求められている。

従来のワームへの事後対応は、検出（警報発動）、対策設計、復旧作業というステップで行われてきた。しかし、未知のワームの場合、分析や対策設計に時間がかかり、その間にワームがネットワークに蔓延してしまうという問題があった。特に、爆発的に増殖するタイプ(Outbreak 型)のワームに遭遇したときは、対策までにかける時間を極力短縮し、感染被害を最小限に留める必要がある。

本稿では、Outbreak 型のワームに感染したケースを想定し、感染活動を抑制しながらシグネチャを自動的に生成・適用することにより、短時間にワームを封じ込め、被害の拡散を防止する方式を提案する。検出から対策までを自動化することによって、未知のワームに対しても、迅速に対応することが可能となる。次章では、提案方式の概要を説明し、3章以降で、提案方式におけるシグネチャ自動生成と管理に焦点を当て、実現方式と試作システムでの評価結果、今後の課題について述べる。

2. ワーム拡散防止システム

図1に、Outbreak 型ワームに感染した場合の感染端末の経過を示す。

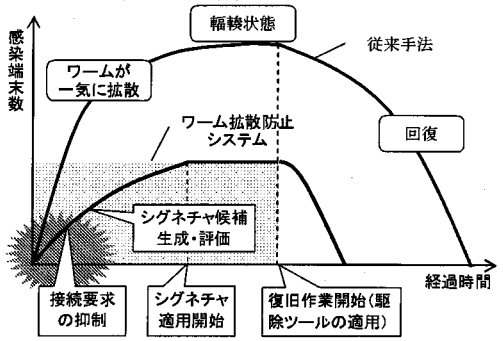


図1：Outbreak 型ワームの感染端末数の推移

ワームは急激に広がり、復旧作業が始まってから、ようやく感染端末数は減少する。従来の対策手法では、感染端末数は膨大となり、最悪の場合、ネットワークは輻輳状態に陥ることもあった。

これに対し、ワーム拡散防止システムでは、業務を継続しつつ、できるだけ早い段階でワームを感染端末が属するセグメント内に封じ込め、他のセグメントへの被害拡大を阻止することを目的とする。

具体的には、以下の2つの機能を提供する。

- Connection Throttling（CT）

感染端末が発信する接続要求パケットを制限することによって、ワームの増殖スピードを落とし、感染端末が大量に発生するのを防ぐ。

- Automated Signature Generation（ASG）

侵入を検知する判定情報であるシグネチャを自動的に生成・適用することによって、ワームの拡散を早期に食い止める。

これらの2つの機能は、ワーム拡散の兆候を検知した直後から並行して実行される。

2.1 構成

本システムは、セキュアゲート装置（SGW）、端末エージェント(Agent)、管理サーバによって構成される。図2にワーム拡散防止システムの構成を示す。

- SGW

ネットワークの拠点やセグメント単位に配置される通信装置。内部コンポーネントに、シグネチャの候補を生成する ASG Maker、バースト的な不正コネクションの接続要求を制限する CT、侵入検知・遮断する IPS、トラヒック情報収集部を持つ。

- Agent

各端末にインストールするソフトウェア。端末内のワーム感染挙動を監視・通知する。

- 管理サーバ

SGW や Agent を管理し、内部コンポーネントに、SGW で生成した候補シグネチャを評価する ASG Mgr と端末の感染を判定する挙動監視アプリケーション(HIA) [1]を持つ。

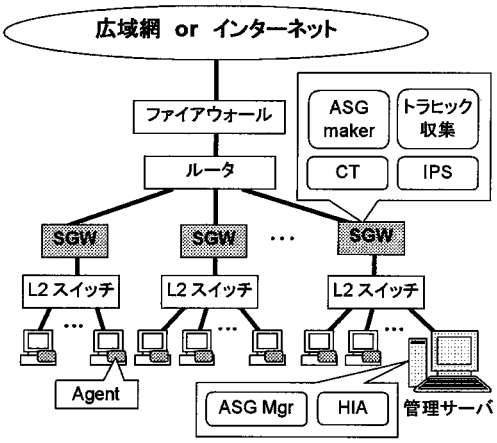


図2：ワーム拡散防止システム

2.2 Connection Throttling

ワーム拡散速度の抑制は、SGW の CT が単独で実施する。正常ホストのコネクションレートを低下させることなく、ワーム拡散に関わる異常パケットのコネクションを確実に排除する[2]。

ワームに感染した端末は、短時間に不特定多数の宛先に対してコネクション接続要求を実施し、偵察活動を行うことが知られている。このとき、単位時間当たりのコネクション接続失敗数が極めて増加する。この特性を着目し、CT では、短時間に大量にコネクション接続要求を送信かつ失敗している端末を見つけると、その端末が送信するコネクション接続要求の中継数に制限をかける。

2.3 Automated Signature Generation（ASG）

ASG 機能には、シグネチャデータの生成から評価、適用までを含み、SGW と管理サーバ、Agent が連携して動作する。

本システムで扱うシグネチャには、2 つの世代が存在する。

- 候補シグネチャ

SGW の ASGMaker が自動的に生成するシグネチャ形式のフィルタリング情報。管理サーバによって他の SGW にも配布される。

- 選定シグネチャ

候補シグネチャのうち、各 SGW での試用期間を経て、管理サーバの ASGMgr で評価した結果、シグネチャとしての適用基準を満たすと判定されたもの。選定シグネチャに至らなかった候補シグネチャは破棄する。

各 SGW には、候補シグネチャと選定シグネチャの両方が適用されることになるが、該当するパケットがきたときのアクションの違いで区別する。候補シグネチャの段階ではパケットの廃棄は行わず、アラート情報を挙げるためのフィルタの役割を果たす。

図 3 に、管理サーバからみたイベントとシグネチャのライフサイクルの関係を示す。

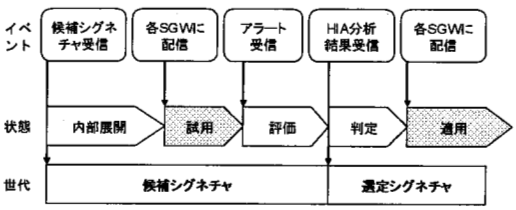


図 3：シグネチャのライフサイクルモデル

シグネチャの状態は管理サーバで管理する。ASG 機能の処理の流れと実装方式については、3 章で述べる。

3. シグネチャの自動生成と評価

3.1 対策の流れ

(1) 情報収集

SGW は常時、中継パケットのトラヒックを検査し、Agent は端末の挙動を監視している。管理サーバの HIA は、定期的にトラヒックの統計情報を SGW から収集する。

(2) 感染の検知と分析の開始

Agent が HIA に異常イベントを通知する。HIA は、トラヒックの統計情報や端末の異常イベントから感染状況の分析を行う[1]。

(3) シグネチャの自動生成と評価

SGW の ASGMaker がトラヒックの傾向から、候補シグネチャを自動生成し、管理サーバの ASG Mgr に通知する。

ASG Mgr は、候補シグネチャを全 SGW に展開し、試用させる。候補シグネチャの内容が適切であれば、各 SGW では、配布されたシグネチャによる IPS アラートが発生するはずである。

ASGMgr は、各 SGW からのアラートと HIA の分析結果からシグネチャを評価・選定する。

(4) シグネチャの適用

管理サーバから、各 SGW に選定シグネチャを適用する。シグネチャにより感染端末からのパケットが廃棄される。

3.2 シグネチャの自動生成

ワームには、感染のたびに変態を繰り返すものが存在するが、端末上のサービスのセキュリティホールを突いて乗っ取りを成立させる「exploit コード」は変態により変化しない部分であり、感染パケットに共通する数 100 バイト程度の固定的な特徴文字列の存在が期待される。従って、以下の特徴を満たす文字列パターンをシグネチャ候補と捉えることができる。

- トラヒックにおける当該文字列の出現頻度が高い
- 当該文字列のアドレス分散度（送受信ホスト数／単位時間）が多い

このような文字列パターンをリアルタイムに抽出し、候補シグネチャを生成する方式を考案した[3]。

(1) 候補シグネチャの作成

SGW の ASG maker による候補シグネチャの作成手順の概要を以下に示す。

1. パケットの組立てとパージング

プロトコルの構文精査、パケットのヘッダ情報の抽出を行う。

2. コンテンツ頻度の測定

文字列の切り出しと、採取した文字列の出現

頻度の計算を行う。文字列検索には、高速にスライド演算が可能な Rabin-Karp アルゴリズム[4]を使用する。また、Multi-state Filter[5]を利用したハッシュ近似計算により、省メモリでのコンテンツ頻度を測定する。

3. アドレス分散度の分析

一定以上の頻度を観測した文字列において、送信元 IP アドレス、宛先 IP アドレスのハッシュ値をビットマップ空間にマッピングし、分散度を近似的に計算する。閾値以上の分散度を持つ文字列を抽出する。

4. ホワイトリストフィルタ

プロトコルの種類により、高頻度で出現するコンテンツにも無害な既知のパターンが多く存在する。このようなパターンを事前にホワイトリストとして登録することで生成対象から除外し、無駄な候補シグネチャが出力されるのを防ぐ。

以上の手順により、最終的に残ったパターンを候補シグネチャとして ASGMgr に送信する。

(2) CT との関係

2 章で、CT によるコネクション要求の抑制と ASGMaker による候補シグネチャの作成は、並行して行われると述べた。しかし、CT で感染速度を抑えたために、ASGMaker が候補シグネチャの作成を始める基準になかなか到達せず、結果としてシグネチャ適用までの時間が延びてしまうという可能性がある。

そこで、本システムでは、CT と ASGMaker を連動させ、適切なタイミングで候補シグネチャを作成させる工夫を施した。CT で抑制されている端末はワーム感染端末の可能性が高いという前提のもと、候補シグネチャに危険度の重み付けを行い、CT で抑制対象となっている端末の送信パケットの危険度を通常より高くすることで ASGMaker の候補シグネチャ作成を促す。

3.3 シグネチャの評価

候補シグネチャは、中継パケットのコンテンツ頻度を元に抽出されている。このため、ホワイトリストによるフィルタだけでは誤検出となるものが含まれることが予想される。シグネチャとして正式に採用するには、候補シグネチャの評価フェーズを設ける必要がある。

ASGMgr は、候補シグネチャを一旦全 SGW に展開・試用させた上で、候補シグネチャの妥当性を評価し、候補シグネチャを選定する。

(1) 評価の基準

シグネチャ評価の考え方として、アラート端末の分布と感染端末の分布が十分一致していることがシグネチャの必要条件であると言える。この考え方を基づき、信憑性の高い候補シグネチャの条件を以下のように仮定した。

- そのシグネチャに関連するアラートが広範囲に発生している
- そのシグネチャに関連するアラートは、感染兆候が認められる端末から発信される

ASGMgr は、これらの条件と相関性の高いシグネチャを選出する。相関性を定量的に評価するため、個々のシグネチャに関する 2 つの指標を定義した。

指標 1 ：アラート発生端末の台数
指標 2 ：ホスト感染率

ASGMgr は、これらの指標をアラート情報や HIA のホスト感染分析結果から算出し、算出した指標が閾値以上の候補シグネチャを選定シグネチャとして採用する。

(2) 評価の手順

図 4 に、シグネチャの評価に必要な参照テーブルの関係を示す。

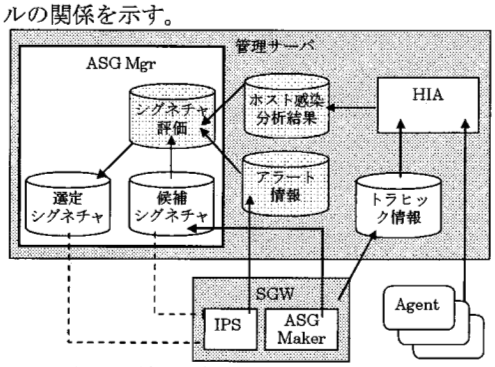


図 4 : 評価の流れとテーブル参照関係

ホスト感染分析結果テーブルは、HIA が出力する時間毎のホスト別感染度情報である。HIA は、確率分布に従った推論を行っており、項目に感染の有無に付随した出力結果の確信度を含む。

指標の算出と閾値比較による判定は、以下の手順で、ASGMgr が周期的に行う。

1. アラート情報の集約

アラート情報テーブルには、随時報告される SGW からのアラート情報が格納されている。個々のアラート情報には、発信端末 IP や詳細情報の他、シグネチャ ID（SID）が含まれ、どのシグネチャに関するアラートなのかがわかる。

ASGMgr は、アラート情報を、SID とアラート発信元 IP 別に集計し、アラート初回発生日時、アラート最終発生日、アラート受信数をシグネチャ管理テーブルに登録する。

2. ホスト感染分析結果の集約

次に ASGMgr は、シグネチャ評価テーブルに存在するアラート発信元 IP ごとに、ホスト感染

分析結果テーブルの判定対象端末 IP を検索する。該当する IP が見つかれば、その端末に関する結果の確信度から可能性（感染度）を計算し、シグネチャ評価テーブルに保存する。

- 検索範囲

判定時刻がアラート初回発生日時の T 時間前から、アラート最終受信時刻までに出力された時刻までの期間の全エントリ（Tは、ASGMgr で事前に設定して、使用環境に応じて調整する）

- 感染有無の確認と感染度の計算

検索範囲に存在する全エントリの感染有無を判定対象ホスト IP 別に調べる。ある判定対象ホスト IP に関して、検索範囲に含まれる該当 IP の全エントリが「感染なし」のとき、感染の疑いはないものと判断し、感染度 0 とする。検索範囲に含まれる該当 IP のエントリの中に一つでも「感染あり」が含まれるとき、次項の式に基づいて、アラート発信元端末の感染度を計算する。

$$\text{感染度} = \frac{\text{感染ありエントリの確信度の合計} + \text{感染なしエントリの(1-確信度)の合計}}{\text{エントリの数}}$$

表 3 において、エントリ No.1 の感染度は、表 2 の確信度から(0.7+0.9+0.8)／3 = 0.8 と計算される。エントリ No.2 は、表 2 において感染可能性なしに該当するため、感染度 0 である。

また、検索範囲に該当 IP のエントリが含まれないとき、分析結果が存在しないことからホスト感染の可能性が極めて低いと判断し、感染度 0 とする。表 3 において、エントリ No.3 に該当する結果は表 2 に存在しないため、感染度 0 である。このとき、ASGMgr は、HIA に対して存在しなかった IP についての分析要求イベントを通知する。

3. 指標の算出

指標 1 と指標 2 の計算を行う。

まず、指標 1 のシグネチャ評価テーブルの各 SID に関するアラート発生端末台数を計算する。表 1 と 3 の例では、SID=2, SID=3 のアラート発生台数はともに 2 台である。

次に、2.で計算した感染度から、個々のシグネチャに関して、指標 2 であるホスト感染率を計算する。具体的には、シグネチャ別に、アラート発生回数を重み付けしたアラート発信元端末のワーム感染度を平均する。表 3 の例では、SID = 2 の端末感染率は、(0.8×3 + 0×1)／(3+1) = 0.6 と計算できる。

4. 判定

指標 1 と指標 2 を閾値と比較し、両方とも閾値を超えた SID を持つ候補シグネチャを選定シグネチャとして採用する。

各指標の閾値はあらかじめ管理サーバに設定し、調整可能なものとする。なお、指標 1 に関しては、制限時間を設け、閾値を満たすことなく一定上の時間が経過した候補シグネチャは不採用とする。

この例の場合、説明のためデータを簡略化するため、指標 1 = 2、指標 2 = 0.6 という結果が出ているが、実際の環境では、指標 1 は数百台以上、指標 2 は 0.95 以上の精度が求められる。

4. 実験

本システムでは、シグネチャ検知精度と即応性の両面から評価を行う必要がある。まず、即応性の観点から評価を進めているが、現時点では、システム全体の評価は完了しておらず、個々の機能に関わる性能を検証している段階である。

本実験では、シグネチャ自動生成の高速性を検証する足がかりとして、ASGMgr の候補シグネチャの収集と配信の処理時間を測定した。

(1) 測定環境

ASGMgr を稼動させた Windows 管理サーバの動作環境は、以下のとおりである。

CPU：PentiumM 1.73GHz

メモリ：2GB

また、管理サーバの通信先となる SGW には、多数の SGW を模擬できる SGW シミュレータを用意した。

(2) 測定結果

SGW の台数を 1 台、10 台、50 台、シグネチャの数を 1、10、100、200 と増やし、それぞれの組合せで候補シグネチャの収集と配送に要した処理時間を表 4, 5 に示す。

グラフからシグネチャの数が変化しても収集にかかる性能に大きな変化はないことがわかる。一方、SGW の台数の増加は、処理時間に大きな影響を与えている。また、シグネチャの配送は、10 台まではシグネチャの数が増えてもあまり変化が見られないが、SGW の台数が 50 台に増えると、シグネチャの数の増加に伴う処理時間が大幅に長くなることがわかった。

(3) 考察

実際に生成される候補シグネチャの数は 10 程度と予測しているので、収集については実環境に耐えられるスペックと判断した。

しかし、配送には分単位の時間がかかっており、ワームの非常に早い拡散スピードを凌駕できるか疑問が残る。この点については、現在 Oracle で保持している情報をデータベースに介さず、すべてメモリに蓄える打開策がある。

表 1 :アラート情報テーブル

No	発生時刻	SID	発信元 IP	詳細情報
1	7:10:00	2	192.168.0.100	
2	7:10:02	2	192.168.0.100	
3	7:10:02	2	192.168.10.1	
4	7:10:15	2	192.168.0.100	
5	7:10:15	3	192.168.0.200	
6	7:10:15	3	192.168.0.100	

表 2:ホスト感染結果テーブル

No	発生時刻	判定対象 IP	感染可能性	確信度
1	7:09:45	192.168.0.100	あり	0.7
2	7:09:55	192.168.0.110	あり	0.6
3	7:10:00	192.168.0.100	あり	0.9
4	7:10:05	192.168.10.1	なし	1
5	7:10:15	192.168.0.100	あり	0.8

表 3 : シグネチャ管理テーブル

No	SID	発信元ホスト	Alert 初回発生時刻	Alert 最終受信時刻	Alert 回数	感染度
1	2	192.168.0.100	7:10:01	7:10:15	3	0.8
2	2	192.168.10.1	7:10:02	7:10:02	1	0
3	3	192.168.0.200	7:10:15	7:10:15	1	0
4	3	192.168.0.100	7:10:15	7:10:15	1	0.8

表 4: 候補シグネチャの収集時間

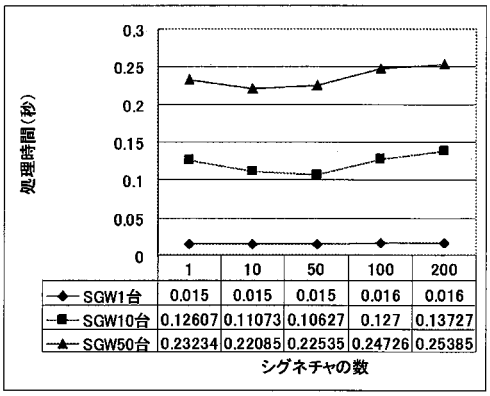
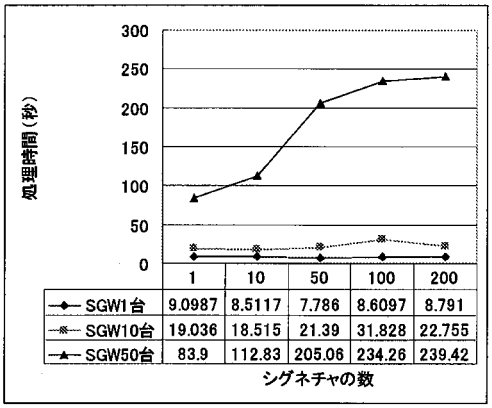


表 5 : シグネチャの配送時間



また、配送に関わる時間のほとんどが、配送前の配送データ生成処理に費やされているため、シグネチャデータの効率的な変換方法についても改善の予知がある。

さらに、各 SGW が連携する仕組みを作り、管理サーバと SGW や Agent の通信量を減らすことによって管理サーバに集中するオーバーヘッドをなくす仕組みが必要である。

5. まとめ

本稿では、ワーム侵入被害の現状を鑑み、万が一、ワームが侵入したとしても、拡散活動を最小限に抑える被害拡散防止システムを提案した。ワームの拡散を早期に封じ込めるため、自動的にシグネチャを生成・評価する方式を示し、実験において実現可能性と課題を述べた。

今後は、実環境への適用を目指し、実験で明らかになった課題の改良とシグネチャ検知精度の評価を行う。また、Outbreak 型以外のタイプのワームにも対応できる方式を検討する。

参考文献

[1] 北澤 繁樹, 樋口 毅, 原田 道明, 藤井 誠司, "マハラノビス距離を用いた判別分析による未知ワーム感染挙動特定方式", 情報処理学会シンポジウム DICOMO2006 予稿集 II-6E3, pp.733-736

[2] 杣信吾, 藤井照子, 馬場 義昌, "L2 ネットワークにおけるワーム拡散抑制のためのコネクション接続制限方式の提案" , 電子情報通信学会総合大会 2007 にて発表予定

[3] 時庭康久, 鈴木清彦, 原田道明, 後沢忍, "侵入検知システムにおけるシグニチャ自動生成方法の検討", 第 69 回 情報処理学会全国大会にて発表予定

[4] Rabin-Karp string search algorithm, http://en.wikipedia.org/wiki/Rabin-Karp_string_search_algorithm

[5] S.Singh,C.Estan,G.Varghese and S.Savage, " Automated Worm Fingerprinting", 2004 USENIX Operating Systems Symposium (OSDI '04), 2004.