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

2003－ＣＳＥＣ－22　(16)
2003／7／17

仮想ネットワークを使った未知ウィルス検知システム

神薗　雅紀†　　白石　善明††　　森井　昌克†

† 徳島大学 工学部 知能情報工学科　〒770–8506 徳島県徳島市南常三島 2–1
†† 近畿大学 理工学部 情報学科　〒577–8502 東大阪市小若江 3–4–1
E-mail: †{kamizono,morii}@is.tokushima-u.ac.jp, ††zenmei@info.kindai.ac.jp

あらまし　電子メールを利用して感染するウイルスが社会問題となっている．特に，一般的に利用されているパターンマッチング手法に基づくウイルス検知技術では検知できない未知ウイルスは，その対策がなされるまで感染を続け，被害が拡大する傾向にある．近年最も注目されているメタモーフィック型と呼ばれる感染する度に自身を変化させるウイルスも，パターンマッチング手法では検知ができないためにその対策が必要とされている．未知ウイルスやメタモーフィック型ウイルスの検知に有効な手法としてダイナミックヒューリスティック方式に基づくものが挙げられる．これは実際に検査対象ファイルを実行し，その行動によりウイルスであるかどうか判断する方法である．本稿では仮想マシンとその閉じた仮想ネットワーク内でウイルスを実行し，その行動パターンを観測することによって未知ウイルスの検知を行うシステムを提案し，その評価を行う．
キーワード　コンピュータウイルス，未知ウイルス，ウイルス検知，仮想ネットワーク，ダイナミックヒューリスティック方式

Unknown Virus Detection System using Virtual Network

Masaki KAMIZONO†, Yoshiaki SHIRAISHI††, and Masakatu MORII†

† Department of Information Science and Intelligent Systems, The University of Tokushima
Minamijosanjima-cho 2–1, Tokushima, 770–8506 Japan
†† Department of Informatics, School of Science and Engineering, Kinki University
Kowakae 3–4–1, Higashi-Osaka, 557–8502 Japan
E-mail: †{kamizono,morii}@is.tokushima-u.ac.jp, ††zenmei@info.kindai.ac.jp

Abstract　The spread of computer virus by E-mail is a social problem. In particular, unknown virus which can not be detected by a general virus detection scheme based on pattern matching tends to expand the damage. It is also necessary to find out a countermeasure against a metamorphic virus, which changes itself whenever it infects a computer, because a pattern matching-based virus detection scheme can not detect the virus. It is known that a dynamic heuristic scheme is effective to detect unknown or metamorphic viruses. In the scheme, after a doubtful target file is actually run on a computer and its behavior on the computer is monitored, we judge whether the file is a virus or not. In this paper, we propose a dynamic heuristic scheme-based system which runs a target file attached to E-mail on a virtual machine and a virtual network, and monitored its behavior pattern in the virtual environment. We describe an implementation of proposed system, and show some evaluation results.
Key words　computer virus, unknown virus, virus detection, virtual network, dynamic heuristic scheme

1. まえがき

近年のインターネットの普及に伴いネットワーク被害が急増している．特にコンピュータウイルスは増殖能力を有し，不特定多数のユーザを標的にするため被害が拡大する傾向にある [8]．

最も一般的なウイルス検知手法であるパターンマッチング方式は既知のウイルスコードの特徴 (以下シグネチャと称する) をあらかじめデータベース化しておく．そして現在検査対象のファイルとシグネチャをマッチングして検査する．このようなパターンマッチング方式は，既知のウイルスに対しては高速に動作するなど有効な方法であるが，未知のウイルスを検知することができないという問題がある．近年最も注目されているメタモーフィック型ウイルス [5] などの感染する度に自身を変化させるウイルスも，パターンマッチング手法では検知することができないためその対策が必要とされている．運用の面では利用者が継続的に最新のシグネチャを更新する必要があり，利用者に負担がかかる．また，シグネチャの更新が遅れるとその間に出現したウイルスの被害を被る可能性があり，シグネチャを発行するベンダーに未知ウイルスの被害通知が届く前にウイルスが急激に拡大する場合もある．

未知ウイルスへの有効な検知手法が著者らによって提案されいる[1]．しかしながら，この手法はウイルスの自己増殖活動と感染活動を検知することができるが，そのウイルスの自己増殖活動による二次的なDoS攻撃等の被害を検知できないという問題があった．また，検査対象の添付ファイルを一つずつ検査するため，実用的な時間で処理が行えないなどの問題もある．

本稿では仮想マシンとその閉じたネットワークを利用してウイルスの自己増殖活動の検知を行うと同時に，先に提案しているシステムの問題点を解決する未知ウィルス検知システムを提案する．提案システムでは複数の検査対象の添付ファイルを同時に検査することで，実環境に適した処理時間で検査ができるという特徴も有する．

2. 従来手法

本節では著者らが既に提案している仮想サーバを使った未知ウイルス検知システム[1]の概要を述べる．以下ではこれを従来システムと呼ぶ．従来手法のシステム構成を図1に示す．

ダイナミックヒューリスティック方式[5]の一つである従来手法は，VMware[9]を使った仮想環境を用いて実現する．VMwareではVMwareをインストールするOSをホストOS，VMware内にインストールするOSをゲストOSと呼ぶ．従来手法ではゲストOSは2つ用意する．各ゲストOSは通常のOSをインストールし，これをホストOS内で仮想マシンとして動作させる．

ホストOSでは二つのゲストOSを制御する機能を持たせ，それぞれのゲストOS内には実際にウイルスに感染させるウイルス実行ホストと，それを監視する監視サーバを用意する．3.以降で述べる提案システムと同様に従来システムにおいてもウイルス検知は実際にゲストOSのウイルス実行ホストをウイルスに感染させ，その感染したホストの状態を見ることによりウイルスを検知する．そこで，ウイルス実行ホストのOSとして感染するウイルスが最も多いWindowsを使用する．VMwareの動作モードを適切に設定することで監視サーバはウイルスに感染する可能性はないが，万全を期すためにWindowsを感染対象としているウイルスに比較的耐性のあるLinuxを導入する．

VMwareにはホストOSの接続された実ネットワークから切り離された仮想ネットワークを構築するための“Host-Only”モードと呼ばれるモードがある．ウイルスを実行した際に監視サーバやそのホストOSが接続されている実ネットワークに被害が及ばないようにするために，このHost-Onlyモードで二つのゲストOSは動作させる．また，ゲストOSの起動方法には実ハードディスクに直接影響を与えない“Nonpersistent”モードがVMwareに用意されている．このモードでゲストOSを立ち上げると，ゲストOS起動時にはあたかもハードディスクに処理を書き込んでいるように振舞うが，ウイルス実行ホストの停止時に全ての処理を破棄してハードディスクの内容に全く影響を与えないようになる．ウイルスを実行した際に実ハードディスクに被害が及ばないようにするため，実際にウイルスに感染させるウイルス実行ホストの入っているゲストOSをこの“Nonpersistent”モードで起動する．これにより，ウイルス実行ホストがウイルスに感染しても実環境に被害が及ぶこともなく，さらにウイルス実行ホストを終了させることでウイルス実行ホストを正常な元の状態に戻すことができる．

従来システムはウイルスの自己増殖活動検査と整合性検査の2つを用いて未知ウイルスを検知する．これらについて以下で説明する．

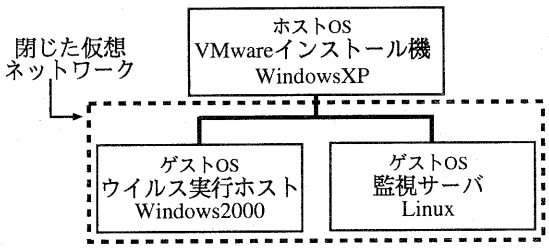

図1　従来システムの構成図 (Host-Only モード)

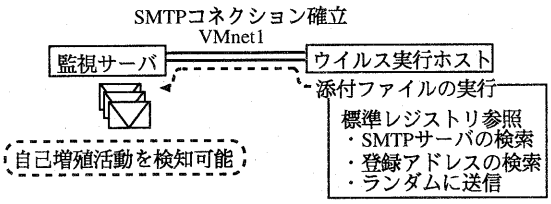

図2　SMTPのポート監視によるウイルスの自己増殖活動検知

2.1　ウイルスの自己増殖活動検知

ワームウイルスは自分自身をコピーし，メールに添付して不特定多数のユーザに送信することで増殖していく．したがって，検査対象の添付ファイルを実行している間にウイルス実行ホストからのメールを監視サーバが受信すれば，ウイルスの自己増殖活動である可能性が高いといえる．図2にSMTPポートの監視によるワームウイルスの自己増殖活動検知手法を示す．ウイルスの自己増殖活動を検知するため，監視サーバはウイルス実行ホストからのSMTPポートのアクセスを監視する．SMTPポートへのアクセスを検知すれば検査対象の添付ファイルがワームウイルスであり，自己増殖活動を行ったと判断する．

2.2　整合性検査によるウイルス検知

ウイルスが感染すると，システムファイルやレジストリの改ざん，削除を行い，そして悪性プログラムの追加などが行われる等の重大な被害を感染マシンは被る．この被害をシステムファイルやレジストリの整合性検査によって検出する．図3に整合性検査によるウイルス検知手法を示す．まず，事前に正常な状態のファイルやレジストリのハッシュ値Hを取得し格納しておく．次に添付ファイルをウイルス実行ホストで実行する．このとき，添付ファイルがウイルスであれば自身のコピーを行ったり，ファイルやレジストリの削除，改ざん，不要なプログラムの追加を行われる．この削除や改ざん，不要なプログラムの追加を検出するために，添付ファイルを開封／実行後にもう一度ファイルやレジストリのハッシュ値H'を取得する．そして最後にHと

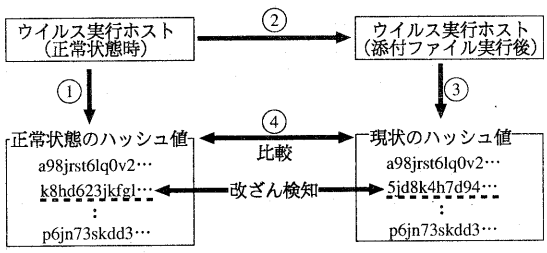

図3　整合性検査

H' を比較する．ハッシュ値が変わっていた場合にはファイルが改ざんされたことがわかり，そのホストはウイルスに感染したこと，つまりその電子メールはウイルスであることが検知される．この整合性検査には Tripwire [10] を利用する．

一般的なダイナミック・ヒューリスティック方式は仮想環境でウイルスを実行し，ウイルスが呼び出した OS のシステムコールを記録し検証することでウイルスであるかを判断する [2]．具体的にはシステムや OS から切り離された領域を確保し，特別な対話ルーチン上でウイルスを実行する．実行時のウイルス特有のファンクションコールを監視し，ウイルスであるかを判断する．このような手法はウイルスのファンクションコールを全て用意しておく必要があり，多用化するウイルスに対応することは困難である．またウイルスを悪意のある振舞いパターンを持つプログラムと捉え，その振舞いを検出することでウイルスを検出する手法も提案されている [3]．しかし，振舞いを隠す細工がしかけられている場合は検知が困難である．既存のシステムコールの監視に基づく方法ではこのように多様化するウイルスの検知は困難であるが，本手法では実際にウイルスに感染させた後に OS のシステムファイルやレジストリの整合性を検査する方法をとることで，ウイルスによる被害を取りこぼすことなく検出することができ，多様化したウイルスにも対応することができるという利点を持つ．

2.3 従来手法の問題点

従来手法はワームウイルスの自己増殖活動が成功した場合にはそのメールがウイルスであることを検知することができる．しかし，ワームウイルスの自己増殖先を探すためのネットワークスキャンや，SMTP ポート以外のポートへのアクセス，そして大量にパケットを送信するなどの被害を検知することができないという問題もある．

整合性検査では改ざんされるとシステムに影響を与えるシステムファイルやレジストリの全てを検査する．整合性を検査するファイルを開き，MD5 などの一方向性ハッシュ関数を用いてその内容のハッシュ値を求める．このとき，整合性を検査する全てのファイルを順にメモリ上に格納していくために，多くのファイルアクセスが必要となり，整合性検査処理時間も長くなる．ここで各マシンの実験環境を表 1 に示し，検査処理時間について述べる．整合性検査は提案システムと従来システムは同じであるため，提案システムの環境での処理時間を示す．この環境下での整合性検査処理時間は約 3 分 10 秒であった．多数のユーザを抱えるインターネットサービスプロバイダ (ISP) や組織のように検査対象メールが多数存在する場合，検査対象メール 1 通ずつにこの整合性検査を行う従来手法では検査処理が追い付かないことがわかる．

本稿では従来手法のこれら 2 つの問題を解決する手法とそのシステムを提案する．ワームウイルスの自己増殖活動検知では SMTP ポートの監視だけでなく，仮想ネットワークを監視することで，ワームウイルスのネットワークスキャンや SMTP ポート以外のポートへの何らかの要求を検知する新たな方法を **3.2** 節で述べる．検査処理時間は検査対象メールの一括検査によって検査処理回数を削減し，検査処理時間の短縮を行う．この方法は **3.5** 節で述べる．

3. 提案手法

3.1 仮想マシンのシステム構成

提案手法のシステム構成を図 4 に示す．提案手法は従来手法と異なり，用いるゲスト OS は一つとする．VMware をインストールするホスト OS を監視サーバとし，ゲスト OS であるウイルス実行ホストを操作する．ホスト OS 内にある監視サーバは感染するウイルスが多くある Windows ではなく，比較的ウイルスに耐性のある Linux を使用する．

ホスト OS とゲスト OS の間の仮想ネットワークは従来手法と同様に実環境に影響を及ぼさない “Host-Only” モードで実現する．ゲスト OS の起動も従来手法と同様に “Nonpersistent” モードで起動し，ハードディスクの保存内容に影響を与えないようにする．

提案手法では未知ウイルスの検知は仮想ネットワーク監視，監視サーバでのウイルスメール取得，そして整合性検査の 3 つによって実現する．監視サーバでのウイルスメール取得手法は直接ウイルスメールの検知を行うわけではないが，**6.3** 節で述べるウイルスメール情報の学習による検査処理回数の削減手法で利用する．この手法の内で 1 つでも何らかの活動を検知すれば，ウイルスを検知したと判断する．整合性検査は **2.2** 節で述べた方法と同様に行う．以下では他の 2 つの手法について述べる．

3.2 仮想ネットワーク監視によるウイルス検知

VMware の仮想ネットワーク (ここでは VMnet1) を監視し，ウイルスの自己増殖活動とそれによる被害の検知手法を述べる．ワームウイルスにシステムが感染した際には，感染マシンから必ず何らかのネットワーク要求が行われる．ワームウイルスが自身をコピーするという自己増殖活動がそれに当たる．そこで，

表 1 実験環境

ホスト OS 監視サーバ	RedHat Linux7.3
CPU	Pentium III 1GHz
Memory	512 Mbyte (160Mbyte はウイルス実行ホストで使用)
HD	Ultra160 SCSI
仮想マシン	VMware Workstation3.2
ゲスト OS ウイルス実行ホスト	Windows2000 Professional
仮想マシン用に割り当てた Memory	160 Mbyte

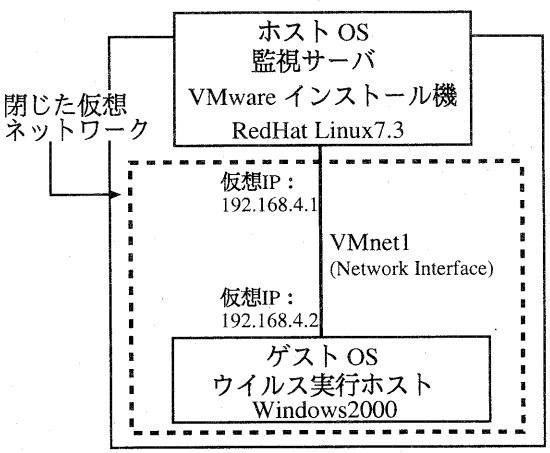

図 4 閉じたネットワーク構成図 (Host-Only モード)

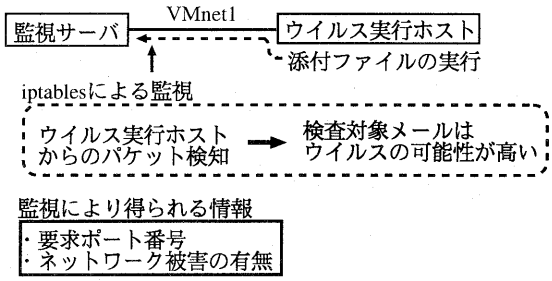

図 5　仮想ネットワーク監視によるウイルス検知

ウイルス実行ホストから何らかのネットワークアクセスを検知すればウイルスであると判断することができる．

図 5 に仮想ネットワーク監視手法を示す．まず，ウイルス実行ホストで検査対象メールの添付ファイルを実行する．この際，実環境上の安全な領域に存在する監視サーバから VMware の仮想ネットワークを監視する．監視方法はホスト OS にパケット・フィルタリングツールである iptables [11] を導入して実現する．仮想ネットワークを監視するためには iptables が監視するネットワークインタフェイスを VMware が提供する VMnet1 とする必要がある．そして，iptables の log を解析しウイルス実行ホストからのネットワーク要求があったか判断する．従来手法では SMTP ポートしか監視できなかったが，iptables を導入することで SMTP ポート以外のポートへ攻撃をするウイルスも検知できる．このように SMTP ポートに限らずウイルス実行ホストから何らかのネットワークアクセスを検知すれば，検査した添付ファイルがウイルスである可能性が極めて高いと判断する．この情報を未知ウイルス判断の要素の一つに利用する．

3.3　監視サーバでのウイルスメールの取得

本手法では検査対象メール全てをウイルス実行ホストで実行する．自己増殖活動を行うウイルスはメールの添付ファイルに自身をコピーして増殖する．このため，検査対象メール内に自己増殖活動を行うウイルスが混入している場合，そのウイルスを実行するとウイルスメールがウイルス実行ホストから SMTP サーバへ向けて送信されるはずである．この SMTP サーバを監視サーバに設定することで，ウイルス実行ホストから送信されるウイルスメールを取得できる．

ここで，一般の MTA を使ってウイルス実行ホストからメールを受信することを考える．一般の MTA はメールを受信した場合，管理しているユーザ宛でなければ他の SMTP サーバへそのメールを転送する．ウイルス実行ホストは仮想ネットワーク内に，そして監視サーバは実環境上に存在するので，ウイルスメールを実ネットワークにある SMTP サーバに転送してしまわないように対策を施す必要がある．そこで，外部の SMTP サーバに転送してしまわないようにメールを受信するだけの MTA を用意し，ウイルス実行ホスト内の Windows の標準アドレス帳にはこの MTA が管理するユーザのメールアドレスを記述しておき，監視サーバの MTA にウイルスメールを誘導する．このような MTA を用いて得たウイルスメールは **6.3** 節で述べるウイルスメール情報の学習による検査処理回数の削減に利用する．

3.4　ウイルス被害の予測と監視サーバの保護

ウイルスの中にはネットワークを輻輳状態にする DoS 攻撃を発生させるものが存在する [4]．近年被害が拡大しているワームウイルスは必ずしも DoS 攻撃を行うとは限らないが，自己増殖機能を有するウイルスは “新たな感染先を検出するためのネットワークスキャン” や “可能な限りの自己増殖活動” を行うことでそのネットワークの輻輳状態を引き起こし，結果的に DoS 攻撃となる場合がある．このようなネットワークの輻輳状態などの被害を検知できれば，検査対象のファイルがウイルスであると考えられる．

監視サーバは仮想ネットワークから切り離された安全な領域に存在するためウイルスに感染することはない．しかしながら，仮想ネットワークの監視は iptabels の log ファイルを解析しているために，ウイルスによるネットワークを輻輳状態にする被害を受けると iptables の log 解析処理に時間がかかり，また監視サーバの処理全般が不能となる恐れがある．このようなワームウイルスのネットワークを輻輳状態にする被害の対策と監視サーバを保護するため，iptables の limit-burst 機能を利用する．

limit-burst 機能とはあらかじめバースト値とパケット受信間隔を設定することで SYN Flood 攻撃や Ping Flood 攻撃のような，大量のパケットを送りつけてくる DoS 攻撃の被害に対処できる機能である．ここでいう DoS 攻撃の被害とは log ファイルの容量の拡大や，ネットワーク帯域の浪費である．バースト値とはパケットを受信する度に減少する値であり，0 になるとあらかじめ用意しておいた条件を実行することができる．例えば，あらかじめ用意する条件をパケット破棄とすることで，バースト値が 0 になるまでの間に受信したパケットのみの log を残すことができ，それ以外は全て破棄することができる．また別のパラメータとして設定しておいたパケット受信間隔以上パケットを受信しなければバースト値は増加され，再度パケットを受信することができるようになるため常に新しい情報を得ることができる．したがって，大量の無駄な log ファイルを解析することなくウイルスの二次的な DoS 攻撃を検知でき，かつ監視サーバを二次的な DoS 攻撃から守ることも可能となる．

バースト値が一度でも 0 になった場合，ウイルス実行ホストから短時間内に大量のパケットを送り付けてられていることがわかる．したがって，ウイルスが攻撃対象となるホストへの DoS 攻撃や監視サーバへの二次的な DoS 攻撃を行ったと判断し，増殖能力や被害の規模が高く危険なワームウイルスであると判断する．バースト値が 0 まで減少しないが少しでも減少した場合

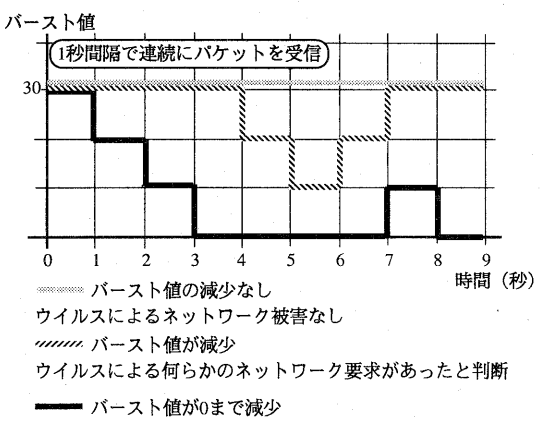

図 6　仮想ネットワーク監視によるウイルス検知

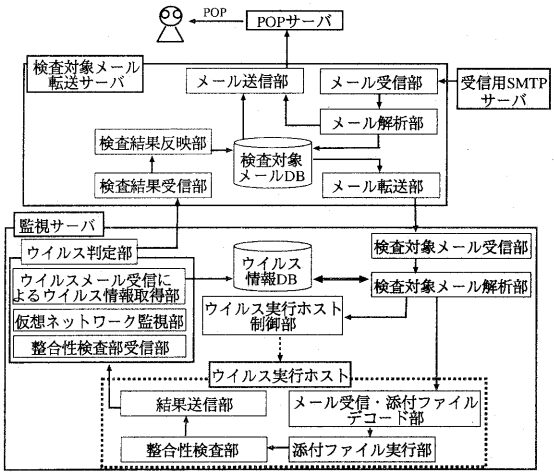

図 7　提案システムの構成図

はウイルスが何らかのネットワーク要求を行ったと判断し，増殖能力や被害の規模がそれほど高くはないワームウイルスであると判断する．このように，ウイルスの二次的な DoS 攻撃や自己増殖活動の程度を予測することが可能となる．

3.5　複数の検査対象添付ファイルの一括検査

実環境では未知のウイルスは既知のウイルス数と比べ非常に少ないと考えても問題はない．このため検査対象添付ファイルの一括検査を行う方が効率が良いと考えられる．添付ファイルを一つずつ検査していたために現実的な時間で検査処理を終えることができないという従来手法の問題を一括検査の方法を取り入れることで解決する．

ウイルス実行ホストで全ての検査対象の添付ファイルを実行し，その間に仮想ネットワークの監視と，監視サーバでウイルスメール受信によるウイルス情報の取得を行う．全ての添付ファイルを実行後，整合性検査を行う．各検査結果全てにおいてウイルスを検知しなければ，検査した添付ファイルは全て正常なファイルであると判断する．つまり複数の検査対象に対する検査を一度の処理で実現できるので，検査対象を一つずつ検査するより効率が良いことがわかる．実環境のような未知のウイルスが既知ウイルスと比べ非常に少ない場合には有効であることがわかる．ただし，各検査のうちで 1 つでもウイルスの活動を検知した場合，複数の検査対象の中に少なくとも 1 つのウイルスが混入されていることがわかる．この場合にはウイルスファイルを特定する処理が必要となる．

複数の検査対象ファイルの中のウイルスファイルを特定する方法は，二分探索法の要領で検査対象メールを分割して検査を行っていく．全検査対象を二分割し，各々をもう一度検査する．ウイルス活動を検知した場合はさらに分割して検査を続ける．ウイルス活動を検知しなければ，そのファイルは検査対象外とし，残りの検査対象ファイルを検査する．このように検査処理を繰り返し，ウイルスファイルを特定する．

4.　システムの実装

提案システムの構成図を図 7 に示す．検査対象メール転送サーバ，監視サーバ，そしてウイルス実行ホストの各処理を説明する．

検査対象メール転送サーバ

検査対象メール転送サーバはウイルスを検知する監視サーバとは切り離されたマシン上にある．検査対象メール転送サーバはメール受信部と検査結果受信部に分かれる．

メール受信部では受信用 SMTP サーバからメールを常に受信し，検査対象のメールであるか判別する．検査対象メールとは添付ファイルの拡張子が実行形式のものである．受信したメールが検査対象である否かは，メールの MIME ヘッダを解析することで実現する．添付ファイルが無いメールはウイルスの可能性が無いため，そのまま POP サーバへ転送する．検査が必要と判断した検査対象メールは，一度指定ディレクトリにスプールしておく．検査対象メール転送サーバは検査対象メールをスプールすると定期的に監視サーバへアクセスし，監視サーバの状態を調べる．監視サーバが受信可能状態であれば，スプールしている検査対象メールを監視サーバへ転送し，それ以外は待機する．

検査結果受信部では監視サーバから検査結果を受信し，検査結果にウイルスがあればスプールしている検査対象メールからそのメールの添付ファイルを削除し，そのウイルス情報に置換する．そして，それ以外のスプールしている検査済メールとウイルス情報を置換したメールを POP サーバへ送信する．

監視サーバ

監視サーバは VMware をインストールするホスト OS であり，この監視サーバ内にゲスト OS であるウイルス実行ホストがある．監視サーバは待機状態と検査処理状態の二つに分かれる．そして，モジュールは大きく状態返答部，メール管理部，ウイルス実行ホスト制御部，自己増殖活動検査部，そしてウイルス特定部に分かれる．

監視サーバは待機状態時は何も処理を行わず，検査処理状態時は各検査を行いウイルスを検査している．

状態返答部は検査対象メール転送サーバから状態問い合わせを受けると，現在の状態を返答する．検査処理状態中の場合は “受信不可” を，待機状態であれば “受信可能” を返す．“受信可能” を返答すると，検査対象メールを受け付ける．

メール管理部では最初に検査対象メール転送サーバから検査対象メールを受信する．そしてウイルス実行ホストに転送する．

ウイルス実行ホスト管理部はウイルス実行ホストの起動，停止を行う．ウイルス実行ホストは “Nonpersistent” モードで起動し，停止はウイルス実行ホストのプロセスを停止させる．

ウイルス特定部では **3.5** 節で述べた一括処理したメールのうち，どのメールがウイルスメールか特定を行う．図 8 にウイルスメールの特定の流れを示す．まず，ウイルス実行ホスト管理部へウイルス実行ホスト起動要求を行う．ウイルス実行ホスト起動後，全検査対象メールをウイルス実行ホストへ転送する．ウイルス実行ホストがウイルス検査を実行中，**3.2** 節で述べたネットワーク監視検査，そして **3.3** 節で述べた監視サーバでのウイルスメール取得手法でウイルスメールを取得し，ウイルスメールを保存する．ウイルス実行ホストより整合性検査結果を受信後，ネットワーク監視結果と合わせ，ウイルス活動を検知したか判断する．そして，保存したウイルスメールを解析しウイルス情報 DB に登録する．このウイルス情報と検査対象メールと比較して，自己増殖を行ったウイルスメールを特定する．ウイルスメールを特定できれば，そのメールは検査対象外とする．この手法は **6.3** 節で詳しく述べる．各ウイルス検知手法でウイルス活動を検知しなければ，検査終了となる．ウイルス活動を検知した場合はウイルスメールの特定処理を行う．ウイルス実行ホストをウイルスが感染する前の正常状態に戻すため，ウイルス

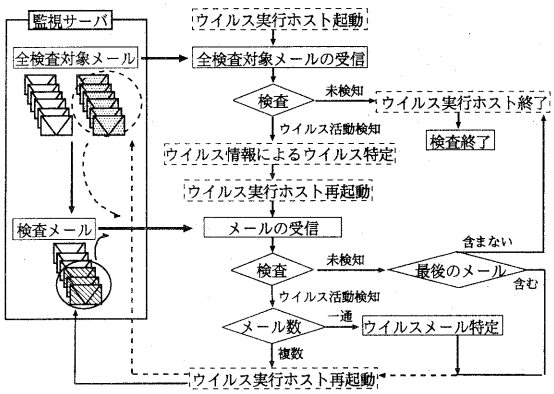

図 8　ウイルスメールの特定処理

```
FROM: VCcp@report
TO: dummy@is.monitoring-server
SUBJECT: Important
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="Boundary-a8dfidaoRadvfuck

--Boundary-a8dfidaoRadvfuck
Content-Type: text/plain; charset=iso-2022-jp
Content-Transfer-Encoding: 7bit
Content-Description: Mail message body

--Boundary-a8dfidaoRadvfuck
Content-Type: application/x-msdownload; name="patch.exe"
Content-Disposition: attachment; filename="patch.exe"
Content-Transfer-Encoding: BASE64

TVpQAAIAAAAEAA8A//8AALgAAAAAAAAAQAAaAAAAAAAA·
--Boundary-a8dfidaoRadvfuck--
```

図 9　受信したウイルスの自己増殖活動メール

実行ホスト管理部へ再起動要求を行う．そして，検査対象メールを二分割しその片方をウイルス実行ホストへ転送する．同様に各検査によりウイルスメールが存在するか判断していく．ウイルス活動を検知すれば，現在検査対象メールを更に二分割して検査していく．最終的に検査対象メールを一通検査し，ウイルス活動を検知すれば，そのメールがウイルスであると判断する．ウイルスメールを特定後，検査結果を検査対象メール転送サーバへ転送する．この検査結果内容はウイルスであるメール名，自己増殖活動の有無や自己増殖による被害状況，整合性検査結果である．

ウイルス実行ホスト

ウイルス実行ホストは VMware によるゲスト OS であり，VMware インストール機である監視サーバ内にある．ウイルス実行ホストは監視サーバからメールを受信し，添付ファイルをデコードする．そしてデコードした全ての添付ファイルを実行する．その後 **2.2** 節で述べた整合性検査を行い，その結果を監視サーバへ通知する．

5. 実　　験

5.1 実験内容と結果

本システムは未知のウイルスの検知を目的としているが，サンプルがないため実際に未知のウイルスを検査することはできない．そこで，今回の実験では近年被害が拡大し，社会問題を引き起こしたウイルスである W32.FBound.exe [6] と W32.Klez.H.scr [7] を検査対象とする．W32.FBound.exe は自分自身のコピーをメールに添付して増殖する単純なワームウイルスであり，W32.Klez.H.scr は増殖と感染活動を行う危険度が高いワームウイルスである．各マシンの実験環境は **2.3** 節で述べた環境と同じである．

実験結果を表 2 に示す．W32.FBound.exe は SMTP ポートを用いて自己増殖を行うため，ネットワーク監視検査でウイルス活動を検知することができた．ならびに監視サーバでのウイルスメールを受信することができた．検査で検知することができた．W32.Klez.H.scr は整合性検査によって検知することができた．

5.2 実験結果に対する考察

W32.FBound.exe は大量メール送信型ワームである．Windows の標準アドレス帳に登録されている全アドレス宛に，感染マシン上のレジストリに登録されている標準の SMTP サーバを使用して自分自身を送信する．

W32.FBound.exe は実行と同時に自己増殖活動を行うため，**3.2** 節で述べた仮想ネットワーク監視手法でネットワークアクセスを検知することができた．また，あらかじめウイルス実行ホストの標準 SMTP サーバを監視サーバに設定することで，**3.3** 節で述べた監視サーバでウイルスメールの取得手法で自己増殖活動であるウイルスメールを受信することができた．受信したウイルスの自己増殖活動であるウイルスメールの一部を図 9 に示す．図 9 より “dummy@is.monitoring-server” というメールアドレスからメールが送信されていることがわかる．“dummy@is.monitoring-server” はあらかじめウイルス実行ホスト内の標準アドレス帳に登録しておいた疑似アドレスであり，ウイルスが標準アドレスを参照して自己増殖活動を行ったことがわかる．W32.FBound.exe はファイル改ざんなどのファイル感染を行わないため，ファイルの整合性検査では検知することはできなかったが，このようにネットワーク監視手法と SMTP ポート監視手法の両手法でウイルス検知に成功した．

W32.Klez.H.scr は大量メールを送信し，感染マシンの秘密情報を漏洩するワームウイルスである．まず，自分自身を Windows のシステムフォルダ内にコピーする．続いてレジストリに値を追加，新たなレジストリキーを作成して挿入することで，Windows が起動される度に毎回ワームが実行されるように設定する．

W32.Klez.H.scr はレジストリの追加や作成が行われるため，整合性検査によってウイルス被害を検知することができた．実験では，“HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Winkpt” というレジストリキーの作成を検知した．

W32.Klez.H.scr は大量メールを送信するが，**3.3** 節で述べた監視サーバでのウイルスメール取得手法でウイルスメールを受信することができなかった．また，仮想ネットワーク監視手法でもウイルス実行ホストからのポートアクセスを検知することができなかった．したがって，自己増殖活動が実行されるための環境が整っていないと考えられる．自己増殖活動の検知に失敗したが整合性検査手法でウイルスを検知でき，結果としてウイルスの検知に成功した．

表 2　実験結果

	ネットワーク監視	ウイルスメール取得	整合性検査
FBound.exe	○	○	×
Klez.H.scr	×	×	○

6. 諸 考 察

6.1 処理時間

実環境において未知ウイルスは既知ウイルスと比較し非常に少ないものとして提案システムでは一括処理をしているが，一括処理するメール数が多くなると，未知ウイルスが検知されたときにはその中からウイルスメールを特定するのに時間がかかってしまうと予想される．そこで，ウイルスが一通のみ混入されているときの提案システムの処理時間について考察する．

検査処理時間はウイルス実行ホストの正常状態への移行処理時間，ウイルス実行ホストの検査対象メール受信・添付ファイルのデコード処理時間，添付ファイルの実行時間，そして整合性検査処理時間である．ウイルス実行ホストの正常状態への移行処理時間とは，ウイルス実行ホストをウイルスに感染する前の正常な状態に戻すために必要な処理時間である．つまり，一度ウイルス実行ホストを停止させ，サスペンドしている正常な状態で起動させるまでに必要な時間である．この各処理時間のうち，検査対象メール受信・添付ファイルのデコード処理，そして添付ファイルの実行処理時間はメール数や添付ファイルの容量によって変わる．表 3 に各処理に必要な時間を示す．メール受信・添付ファイルのデコード処理時間は 12KB の添付ファイル付きメールの 1 通の場合の処理時間を表示する．

検査対象メール L 通の中にウイルスが 1 通混入されている場合にウイルスメールを特定するために必要な検査処理回数を示す．提案手法では複数の検査対象メールの一括検査を行うために，その中のどのメールがウイルスであるか，一度の検査で特定することはできない．ウイルスメールを特定するために，**3.5** 節で述べた二分探索の要領で検査対象メールの半分を検査する．その中にウイルスが混入されていることが判明しても，残りのメールにもウイルスが混入されていないか検査する必要がある．したがって，検査対象メール内にウイルスが 1 通のみ混入されている場合は，二分探索処理のちょうど 2 倍の検査処理回数が必要となる．検査処理回数を式 (1) に示す．

$$\text{検査処理回数} = 2log_2 L + 1 \qquad (1)$$

ウイルスメールを特定するため，式 (1) に示す回数だけ各ウイルス検査を行う．

6.2 複数の監視サーバによる分散処理

提案システムは検査対象メール転送サーバによって検査が必要と判断されたメールを監視サーバに転送し，検査処理自体は各監視サーバで行う．したがって，監視サーバを複数台用意し，検査対象メールの分散検査処理が可能である．実環境を想定し，検査対象メール内に 1 通のウイルスメールが混入されている場合の複数の監視サーバによる分散処理を考慮したシステムのスループットの期待値を考える．監視サーバ数が 10 台，そして検査対象メール数が 1,000,000 通の場合の分散処理を導入したスループットの期待値は約 18,000(通/sec) となる．1 秒間に 18,000 通のメールを検査できるので，多数のユーザを抱えるインターネットサービスプロバイダ (ISP) などの検査対象メールが多数存在する環境でも提案システムは対応できることがわかる．

表 3 監視サーバのウイルス検査に要する各処理時間

ウイルス実行ホストの正常状態への移行処理時間	3 秒
整合性検査処理時間	3 分 10 秒
メール受信・添付ファイルのデコード処理時間 ＋ 添付ファイルの実行時間	0.252 秒

6.3 ウイルスメール情報の学習による検査処理回数の削減

本システムで最も時間を必要とする処理は **2.2** 節で述べた整合性検査処理時間である．この整合性検査回数を削減することで，システムのスループットを向上することができる．**3.3** 節で述べた監視サーバでのウイルスメール取得手法によって得られたウイルスメールを解析しその解析情報を学習することで，複数の検査対象メールの中からウイルスメールを特定するまでに必要な各検査の処理回数を削減する．自己増殖活動を行うウイルスは自身を添付ファイルにコピーし，それを送信することで増殖を繰り返す．したがって，ウイルス実行ホストで検査対象添付ファイルを実行した際にウイルスメールが検出されると，得られたウイルスメールの添付ファイルと同じ添付ファイルを持つメールが検査対象メール内に存在するはずである．得られたウイルスメールの添付ファイルと同じ添付ファイルを持つメールが，その自己増殖活動であるウイルスメールを送信したウイルスメールそのものである．監視サーバでは検査対象メールを保持しているため，添付ファイルと同じ値を持つウイルスを検知することができ，ウイルスメールを特定することができる．ウイルスメールが同じであるか否か判断する方法として，ハッシュ関数 MD5 に通して得られた添付ファイルのハッシュ値を用いる．このハッシュ値を学習し，同様に検査対象メールの添付ファイルのハッシュ値を求め，それらを比較してウイルスメールを特定し，検査処理回数を削減する．

メールの解析方法を図 10 に示す．監視サーバは検査対象メールを検査対象メール転送サーバから受け取ると，各メールの添付ファイル部分を抜きだし，一方向性ハッシュ関数である MD5 を使って添付ファイルのハッシュ値 H を保存する．そして，検査対象メールをウイルス実行ホストで実行させる．検査対象メールにワームウイルスが存在すれば，**3.3** 節より監視サーバは自己増殖活動によるウイルスメールを受信する．監視サーバは受信したウイルスメールの添付ファイル部分を抜きだし，同様にハッシュ値 H' を求める．先に保存していた各添付ファイルのハッシュ値 H と，ウイルスの自己増殖活動によって得られた添付ファイルのハッシュ値 H' を比較する．$H = H'$ であれば，H' が得られたメールはウイルスメールである．ウイルス情報の学習によるウイルスメールの特定方法を図 11 に示す．

ここで W32.FBound.exe の場合を例にあげて説明する．最初に，監視サーバが検査対象メール転送サーバから送られてきたウイルスメールのハッシュ値を求める．ハッシュ値は “2c580f3943d44ddcf82b26d813f83c59” となった．次にウイルス実行ホストで W32.FBound.exe を実行し，自己増殖活動によるウイルスメールを受信した．受信したメールの添付部分のハッシュ値を求めると “2c580f3943d44ddcf82b26d813f83c59” とな

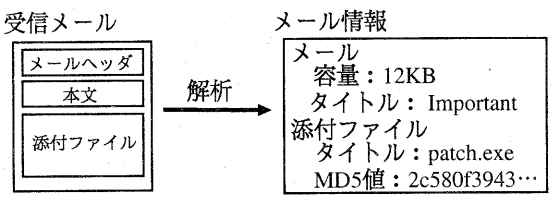

図 10 ウイルスメール情報の解析

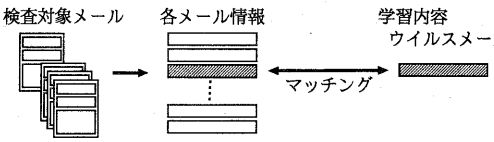

図 11　学習機能によるウイルス検知手法

り，あらかじめ求めていたハッシュ値と等しいことがわかる．つまり，ウイルスメールのハッシュ値と等しいハッシュ値を有するメールは，ウイルスメールであることが確認できる．したがって，監視サーバ内の検査対象メールのどのメールがウイルスであるか，一度ウイルス情報を取得するれば，自己増殖を行うウイルスに関しては添付ファイルのハッシュ値による比較のみでウイルスを検知することができる．これより，**3.5** 節で述べた検査対象メールの一括処理による，ウイルスメールを特定するまでの検査処理回数を削減することができる．仮に検査対象メールに混入されているウイルスが自己増殖を行うウイルスであり，かつそのウイルスメール情報を監視サーバで取得することができたと仮定する．この場合，一度だけウイルス実行ホスト内で検査対象メールを実行し，取得したウイルスメール情報で全てのウイルスを検知することが可能となる．

同時にウイルスメールのヘッダなどを解析することにより，メールのタイトルや添付ファイルのサイズ，そして添付ファイル名も学習することができる．**3.3** 節で述べた監視サーバでのウイルスメール受信手法により受信したウイルスメールの解析法を図 10 に示す．この解析情報をウイルスメールの特徴とし，より広くウイルスメールを検知することに利用する．例えば subject が同一のものなどは，ウイルスの可能性が高いと考えられる．図 9 の W32.FBound.exe の自己増殖活動によるメールの場合，メールタイトルは “Important”，容量は “12KB”，添付ファイルタイトルは “patch.exe”，そして添付ファイルのハッシュ値は “2c580f3943d44ddcf82b26d813f83c59” というウイルスメールの情報を学習し，ウイルス検査に利用する．

以上より，ウイルスメールを解析しその特徴を学習することで，ウイルスメールと特定することができることがわかる．また，自分自身をそのままコピーするワームウイルスは，ウイルス実行ホストで最低一回実行し自己増殖活動によるウイルスメールを受信することができれば，後は学習機能で全てウイルスを検知することが可能となり，処理時間を削減することができる．

6.4　誤検知に対する対策

提案システムでは正常な添付ファイルをウイルスと誤検知する可能性がある．例として，正規の patch ファイルなどが挙げられる．patch ファイルはシステムファイルなどを変更するため，整合性検査ではファイルの改ざんがあったと判断してしまう．提案システムではウイルスと判断した添付ファイルを持つメールはその添付ファイルを削除し，ウイルス情報に置換してユーザにメールで通知する．この処理では正常なファイルを削除してしまう場合もでてくる．正常な添付ファイルの削除を防ぐために，ウイルスの可能性が高いと判断した添付ファイルを特定領域に保存し，その URL をメールでユーザに通知する機能を追加する必要がある．ユーザは通知された URL から添付ファイルを受信する．これにより正常なファイルを削除してしまう問題点を解決でき，またユーザは受信しようとしているファイルがウイルスである可能性が高いことがわかるため，不用意にファイルを展開してシステムに感染することを防ぐことが可能となる．

7. ま と め

本稿で提案したダイナミックヒューリスティック方式の一つであるウイルス検知システムは一般的に用いられているパターンマッチング方式では検知できない未知ウイルスの検知を実現した．

ダイナミックヒューリスティック方式における既存の手法はシステムや OS から切り離された領域を確保し，特別な対話ルーチン上でウイルスを実行する．そして，実行時のウイルス特有のファンクションコールを監視し，ウイルスであるか判断する．このような手法はウイルスのファンクションコールを全て用意しておく必要があり多用化するウイルスに対応することは難しく，また設定不備によってウイルスを検知できない恐れがある．特に未知ウイルスを検査対象とする場合は検知できない可能性が高いと言える．

本稿で提案したシステムでは仮想マシンと閉じたネットワークによる実環境に近い環境を用意し，この実環境に近い仮想環境上でウイルスを動作させ，その行動結果を観測することでウイルスを検知する．増殖するウイルスは必ずネットワークを用いることに着目し，ウイルスがどのように多用化しても，ネットワークの要求を監視することでウイルスを検知できると考えている．そして，従来手法では実現できなかったワームウイルスの自己増殖活動による二次被害の検知と，複数の検査対象メールを一括して検査する方法を実現した．システムを実装し，実環境を想定して評価を行い，既知のウイルスと比べて未知のウイルスが非常に少ない環境では，実用的な時間で検査することができることを示した．

謝辞

有益な御討論を頂いた中尾康二氏をはじめとする KDDI 研究所ネットワークセキュリティーグループの各位に感謝する．

文　献

[1] 三宅崇之，白石善明，森井昌克，“仮想サーバを使った未知ウイルス検知システムの提案，” 情報処理学会 コンピュータセキュリティ，No18，pp.45–52，Jul. 2002.

[2] 竹内大輔，千石靖，服部進実，“仮想マシンを用いた実行形式型ウイルスの検出，” コンピュータセキュリティシンポジウム 2001，pp.179–184，Oct. 2001.

[3] 松浦佐江子，加藤道子，小島量，“未知のコンピュータ・ウイルス検出プログラムの開発，” 情報処理学会 コンピュータセキュリティ，No20，pp.89–94，Feb. 2003.

[4] 三輪信介，滝澤修，大野浩之，“トラフィックジェネレータによる DDoS 攻撃の再現，” 情報処理学会 コンピュータセキュリティ，No20，pp.121–126，Feb. 2003.

[5] 安東一真，河井保博，“ウイルス対策最前線，” 日経 Internet Solutions，Vol.12，No65，pp68–81，2002.

[6] シマンテック，“Security Response W32.FBound.gen@mm” *available at http://www.symantec.com/region/jp/sarcj/data/w/w32.fbound.gen@mm.html*

[7] シマンテック，“Security Response W32.Klez.H@mm” *available at http://www.symantec.com/region/jp/sarcj/data/w/w32.klez.h@mm.html*

[8] IPA セキュリティセンター，*http://www.ipa.go.jp/security/*

[9] VMware, Inc., VMware, *http://www.networld.co.jp/products/vmware/*

[10] Tripwire, Inc., Tripwire, *http://www.tripwire.co.jp/*

[11] iptables, *http://www.netfilter.org/*