自己ファイルREADの検出による未知ワームの検知方式

松本隆明[†] 鈴木功一[††] 高見知寛[††]
馬場達也[†] 前田秀介[†]
水野忠則[†††] 西垣正勝[†††]

ワームの感染は，ワーム自身を他のPCにネットワーク経由でコピーすることにほかならない．よってワームの感染行動は，OSのファイルシステム上では，自分自身のファイルをREAD（コピー）し，これを通信APIにWRITE（ペースト）するという動作として現れる．本論文では，この「ワームの自己ファイルREAD」を検出することにより，ワームを検知する方式を提案する．原理的にはワームは必ず自己ファイルREADを行うため，本方式によれば未知ワームや変異型ワームも検知可能であると考えられる．また本方式は，エンドユーザのPCにおける各プロセスのファイルアクセスを常時監視することにより実装可能であるため，ワームのリアルタイム検知も実現できる．本論文では本方式のコンセプトを示したうえで，ファイルアクセスを監視するモニタツールを用いて擬似的に本方式の未知ワーム検知能力を検証する．

An Unknown-worm Detection Based on Capturing Self-initiated READ Behavior

Takaaki Matsumoto,[†] Koichi Suzuki,[††] Tomohiro Takami,[††]
Tatsuya Baba,[†] Shusuke Maeda,[†] Tadanori Mizuno[†††]
and Masakatsu Nishigaki[†††]

Worm infection is just to copy the worm onto other PC by way of a network connection. Therefore, it is observed as the following behaviors; (1) COPY: read their own executable file, and (2) PASTE: write the file onto stream communication API. This paper proposes to use this type of worm's "self-initiated READ behavior" for unknown-worm detection. It is expected that the worm detection based on capturing self-READ behavior could be applicable to a variety of worms including mutated-worm since this behavior is basically found in most of them. Moreover, this scheme could achieve real-time worm detection because the self-READ behavior can be captured just by watching the file accesses of every process. In this paper, the conceptual design of the proposed scheme is described and its feasibility is investigated by using a tool kit to capture the file access in the OS.

1. はじめに

これまでに様々な未知ワーム検知手法が提案されてきており，その代表的なものが，プログラムの振舞いにおける「ワームらしさ」を検出するビヘイビアブロッキング法[1)]である．ワームらしい振舞いとしては，一般的に，レジストリの改ざん，システムファイルの書き換え，外部への感染活動などが規定される[1)]．ビヘイビアブロッキング法は，プロセスが発行するシステムコールなどを検査することによりコンピュータ上で動作しているプログラムの動きを監視し，ワームらしい振舞いをした場合に，それをワームとして検知する．

ビヘイビアブロッキング法はワームらしいプログラムをすべて検知することができるため，基本的には，亜種ワームやポリモーフィック/メタモーフィック型ワームを含むすべての未知ワームを検知することが可能である．しかし，レジストリの改ざんやシステムファイルの書き換えは，OSのアップデートや正常のアプリケーションプログラムのインストール時にも発生する振舞いであり，また，外部への感染活動と正常の通信を確実に見極めるには限界がある．このため，ビヘイビアブロッキング法には，ワームと類似した動きをす

† 株式会社NTTデータ技術開発本部
R&D Headquarters, NTT Data Corporation
†† 静岡大学大学院情報学研究科
Graduate School of Informatics, Shizuoka University
††† 静岡大学創造科学技術大学院
Graduate School of Science and Technology, Shizuoka University

る正常なプログラムを誤検知してしまうという大きな問題がある．すなわち，ビヘイビアブロッキング法の効果を高めるためには，真に「ワームらしい振舞い」を見極めることが重要となる．

そこで本論文では,「ワームは，感染行動の中で自分の複製を作成するにあたって，自分自身のファイルを READ する」というワーム特有の振舞いに着目し，プロセスのファイルアクセスをリアルタイムで監視することによりワームを検知する方法を提案する．原理的にはワームは感染行動の際に必ず自己ファイル READ を行うため，本方式によれば未知ワームや変異型ワームも検知可能である．

2. 自己ファイル READ の検出によるワーム検知

ワームの感染は，ワーム自身を他の PC にネットワーク経由でコピーすることにほかならない．ファイルのネットワーク経由のコピーを，コピー元の PC における OS（Windows）の観点で見ると，その処理は，1) コピー元のファイルを読み出し（READ），2) これを通信 API である WINSOCK に引き渡す（WRITE），という動作により実行される．よってワームが感染活動を行うにあたっては，メモリにて稼動しているワームのプロセスがファイルシステム上にある自分自身の実行ファイル（ワーム本体のファイル）を READ するという動作が発生する☆．以降，本論文では上記のようなワームが自分自身のファイルを READ するという動作を「自己ファイル READ」と呼ぶことにする．

本論文では，この「ワームの自己ファイル READ」を，ビヘイビアブロッキング法におけるワームらしい振舞いとして利用する．具体的には，OS のファイルシステムをフックすることにより，エンドユーザの PC における全プロセスのファイルアクセスを常時監視し，自己ファイル READ を行ったプロセスをリアルタイム検知する．よって，ワームが感染活動を開始した瞬間にこれを発見することが可能である．

通常，実行プログラムはその一部分が欠落するだけで正常に動作をしなくなるため，ワームが感染するにあたっては基本的に自分自身をそっくりコピーすることになる．そこで本方式では，PC 内で実行中の全プロセスのファイルアクセスを監視し，自分自身の本体のファイルのヘッダ領域を除くプログラム（以下，プログラム領域）のすべてを READ したプロセスが検出された時点でアラートを上げる．

ファイルシステムのフックを行うにあたっては，フィルタドライバを改造することで比較的容易に実装が可能である．また，ファイルシステムのフックは，動的ヒューリスティック法[2] のように PC 上で常駐的に仮想環境を用いるような方法より低負荷であるため，エンドユーザの PC の性能に鑑みてもリアルタイム検知が十分可能である．

また，本方式の特長として，本方式は従来の方法では検知が困難であった変異型ワームに対しても有効に検出が可能である点があげられる．変異型ワームは感染のたびに自分自身を暗号化（ポリモーフィック型）または難読化（メタモーフィック型）により自己改変するが，これらの処理は，1) 自分自身のコードを READ してメモリに展開し，2) コードの自己改変を行い，3) これを他の PC に送信する，という手順で行われる．すなわち，変異型ワームも必ず自己ファイル READ を行う．

3. 検知能力に関する検証

3.1 検証の目的

ビヘイビアブロッキング法によるワーム検知方式の要件として以下のものがあげられる．

【機能要件】

1. 変異型ワームを含む，未知ワーム検知が可能である．
2. 正規プロセスを誤検知しない．

【性能要件】

1. 検知速度が速い（リアルタイム検知）．
2. オーバヘッドが少ない．

本論文は，基本設計仕様の提案と試実験による本方式の有効性の報告に注力しているため，上記要件のうち，機能要件に焦点を当て，検証を行う．

3.2 検 証 方 法

機能要件の検証をするにあたっては実際にシステムを実装する必要はないため，ファイルシステムのフックの代わりに OS のファイルアクセスをリアルタイムで監視可能なモニタツールである FileMon[3] を用いて自己 READ の検出を行うことにより，検証を行った．具体的には，FileMon により PC 内で発生したすべてのファイルアクセスを記録し，プロセスが自分自身の本体のファイルのプログラム領域のすべてを READ するかどうかをチェックする．すなわち，パス名 A のファイルを実行することによって生起されたプロセス

☆ 非常に多くのワームに共通して見られる「OS のシステム管理下のフォルダに自身のコピーの書き込む」という挙動も，コピー元のファイル（ワーム自身の実行ファイル）を読み出してコピー先（システムフォルダ）に書き込むという動作によって実行されるため，やはり，自己ファイル READ が検出されることになる．

が，パス名 A のファイルのプログラム領域のすべてを READ した場合に，ワームの疑いありと判断する．

3.3 正検知実験

本方式によって実際に未知ワーム検知が可能であるかどうかの実験を行った．ただし，未知ワームの入手が困難であるため，ここでは代表的な既知ワームのファイルアクセスを見ることで仮想的に未知ワーム検知が可能であるかを確認することとした．**表 1** に本実験で用いたワームの種類を示す．

本実験は，物理的に隔離されたネットワーク上で行った．隔離されたネットワーク上の PC でワームを実行し，FileMon でワームのファイルアクセスを観測する．なお，実験に使用した PC の OS はセキュリティパッチの当たっていない Windows2000 である．表 1 には，実験の結果も併記してある．○ は検知を表し，× は検知されなかったことを表す．シーケンシャル READ，ブロック READ のいずれかが ○ になっていれば本方式による検知を表す．

表 1 検査対象ワームおよび検知結果
Table 1 Worm detection by the proposed scheme.

ワーム	型	シーケンシャル READ	ブロック READ
Sasser.C	脆弱性悪用型	○	×
Blaster.C	脆弱性悪用型	○	×
Beagle.X	メール送信型	○	×
Netsky.B	メール送信型	×	○
Netsky.D	メール送信型	○	×
Netsky.Z	メール送信型 Zip 圧縮型	×	○
Beagle.AG	メール送信型 鍵付き Zip 圧縮型	○	×
Mimail.Q	メール送信型 自己変異型	○	○

ワームのファイルアクセスを観測した結果を解析したところ，2 種類の方法で自己ファイル READ を行っていることが確認された．以下，それぞれについて説明する．

- シーケンシャル READ

  Beagle.X，NetSky.D，Blaster.C，Sasser.C，Mimail.Q においては，自分自身のファイル（ワーム本体）のファイルサイズなどを調べたうえで，ファイルの中身全体をシーケンシャルにコピーするという動作がとらえられた．「Open Sequential Access」オプション付きのファイル OPEN，ファイル CREATE，ファイル WRITE の処理の中で自分自身がシーケンシャルに READ されている．Filemon によるファイルアクセスの観測結果（の一部）を**図 1** に示す．今回の実験に用いたワームにおいては，シーケンシャル READ により，自分自身のファイルをシステム管理下のフォルダにコピーする挙動が見られた．

- ブロック READ

  NetSky.B，NetSky.Z，Mimail.Q においては，自分自身のファイル（ワーム本体）をブロック（たとえば NetSky.Z では 1,024 バイト）ごとに次々と READ していることが分かった．Filemon によるファイルアクセスの観測結果（の一部）を**図 2** に示す．今回の実験に用いたワームにおいては，ブロック READ により，自分自身のファイルをシ

OPEN	C:¥avserve2.exe	SUCCESS	Options: Open Sequential Access: All
QUERY INFORMATION	C:¥avserve2.exe	SUCCESS	FileAttributeTagInformation
QUERY INFORMATION	C:¥avserve2.exe	SUCCESS	Length: 15872
QUERY INFORMATION	C:¥avserve2.exe	SUCCESS	Attributes: RA
QUERY INFORMATION	C:¥avserve2.exe	SUCCESS	FileStreamInformation
QUERY INFORMATION	C:¥avserve2.exe	SUCCESS	Attributes: RA
QUERY INFORMATION	C:¥avserve2.exe	SUCCESS	FileEaInformation
CREATE	C:¥WINNT¥avserve2.exe	SUCCESS	Options: OverwriteIf Sequential Access: All
SET INFORMATION	C:¥WINNT¥avserve2.exe	SUCCESS	Length: 15872
QUERY INFORMATION	C:¥avserve2.exe	SUCCESS	Length: 15872
WRITE	C:¥WINNT¥avserve2.exe	SUCCESS	Offset: 0 Length: 15872
SET INFORMATION	C:¥WINNT¥avserve2.exe	SUCCESS	FileBasicInformation
CLOSE	C:¥avserve2.exe	SUCCESS	

図 1 シーケンシャル READ を行うワーム（Sasser.C）の観測結果の一部
Fig. 1 File access log for sequential-READ obtained with Sasser.C.

READ	C:¥Informations.txt(大量のスペース)*.exe	SUCCESS	Offset: 0 Length: 1024
READ	C:¥Informations.txt(大量のスペース)*.exe	SUCCESS	Offset: 1024 Length: 1024
READ	C:¥Informations.txt(大量のスペース)*.exe	SUCCESS	Offset: 2048 Length: 1024
READ	C:¥Informations.txt(大量のスペース)*.exe	SUCCESS	Offset: 3072 Length: 1024
：	：	：	：
READ	C:¥Informations.txt(大量のスペース)*.exe	SUCCESS	Offset: 20480 Length: 1024
READ	C:¥Informations.txt(大量のスペース)*.exe	SUCCESS	Offset: 21504 Length: 1024
READ	C:¥Informations.txt(大量のスペース)*.exe	END OF FILE	Offset: 22016 Length: 1024

＊ NetSky.Z は，プログラム名に大量のスペースが含まれるが，図では（大量のスペース）と表した

図 2 ブロック READ を行うワーム（NetSky.Z）の観測結果の一部
Fig. 2 File access log for block-READ obtained with NetSky.Z.

ステム管理下のフォルダにコピーする挙動や，自分自身のファイルを（WINSOCK などへ引き渡すために）メモリ領域に読み出す挙動が見られた．

以上のように，今回の試実験では，すべてのワームにおいて，自分自身のファイルのプログラム部分のすべてが READ されていることが確認された．

NetSky.Z は感染の際に自分自身を Zip 圧縮したものを相手に送りつけるメール送信型ワームである．同じく，Beagle.AG は自分自身を鍵付き Zip 圧縮したものを送る（鍵はつねに変化する）．Mimail.Q は感染のたびに自分自身を変異させる．本実験を通じ，本方式がこれら変異型ワームを含む一般的なワームに対して有効であることが確認された．

3.4 誤検知実験

本方式によって正規のプロセスがワームとして誤検知されることがないかどうかを調べる実験を行った．**表 2** に本実験で用いた正規プロセスと実験結果を示す．× は本方式において誤検知が発生しなかったことを表す．

MS WORD，MS EXCEL においては，プロセスを起動させた後，しばらくの間のファイルアクセスをモニタリングしたが，自己ファイル READ は観測されなかった．

インストーラ（sinst1-4-7-0.exe [4]，memcl [5]，Norton Antivirus）においては，稼動（インストール実行）中に自分自身のファイルを READ するイベントが観測された．しかし，インストーラは，インストールされるプログラム群に関するデータは READ されるが，インストールを制御するプログラム部分については READ されない．よって，プログラム領域のすべてが READ されることはなかった．

Internet Explorer，Adobe Reader，SymantecClientFirewall においても，プロセスを起動させた直後に，自分自身のファイルの一部を READ するイベントが観測された．しかし，自己ファイルを READ する対象がファイルのプログラム領域の数%にしか満たないもの，また，ファイルのヘッダ領域の一部のみを READ しているもののみであった．

表 2 検査対象ワームおよび検知結果
Table 2 False positive of the proposed scheme.

正規プロセス	誤検知
MS WORD	×
MS EXCEL	×
Adobe Reader	×
Symantec Client Firewall	×
Internet Explorer	×
インストーラ (sinst1-4-7-0.exe)	×
インストーラ(memcl)	×
インストーラ (Norton AntiVirus)	×

以上より，自己ファイル READ が検知された際に，「ファイルのプログラム領域のすべて」を READ しているかどうかということをチェックすることにより，ワームとその他のプログラムを切り分けることが可能であることが確認された．

4. 考　　察

4.1 既存方式との比較

本方式の目的は，ビヘイビアブロッキング法の効果を高めるというものである．そこで，本方式と既存のビヘイビアブロッキング法について，検知漏れと誤検知の発生状況を比較し，考察を行った．評価に際し，用

意したアプリケーションは，常駐型アプリケーションのインストーラ（memcl[5]），メーラ（Edmax），FTPクライアント（Smart FTP），ブラウザ（Internet Explorer），Ethereal，アンチウイルスソフト☆（Norton AntiVirus）のインストーラである．また，ワームは，3.3 節の正検知実験で用いたワームを使用した．

これらに対し，既存のビヘイビアブロッキング法で規定されている「ワームらしい振舞い」と本方式による検知を実際に行い，比較を行った．

本実験では，既存のビヘイビアブロッキング法で規定されている「ワームらしい振舞い」としては以下の 6 つを取り上げた．それぞれの説明と具体的な監視方法，および，誤検知，検知漏れの実験結果を以下に示す．

1. OS 起動時の自動実行[6]
   【振舞い】：ワームはできるだけ長い期間，クライアントに感染し続けようとするため，再起動後にも自身が自動実行されるようにする．
   【監視対象】：OS の自動実行に関わるレジストリや，スタートアップへの書き込み（タイプ 1）．
   【結果】：常駐型のプログラムは OS 起動時に自動実行されるようにするため，インストーラによるインストールの際にレジストリへの追記が行われた．よって，インストーラをワームであると誤検知した．
2. 起動直後のファイルコピー[7]
   【振舞い】：ワームは OS のシステムフォルダ以下に自身のコピーの書き込みを試みる．
   【監視対象】：システムフォルダ以下への書き込み（タイプ 2）．
   【結果】：インストーラは起動直後に OS のシステムフォルダ以下にファイルコピーを行った．よって，インストーラをワームであると誤検知した．
3. 別プロセスの起動[8]
   【振舞い】：ワームは別のプロセスを起動することによって自身の行動を隠そうとする．また，感染などの目的のために別プロセスを起動する．
   【監視対象】：ユーザが起動していないプロセスの追加（タイプ 3）．
   【結果】：ブラウザとアンチウイルスソフトのインストーラは別プロセスを起動した．よって，これらをワームであると誤検知した．一方，ワームの中に別プロセスを起動しないものが存在した．このようなワームに対しては検知漏れが生じた．
4. トラフィック量の上昇[9]
   【振舞い】：ワームは他の多数の PC への自己複製を行う．
   【監視対象】：PC の送信トラフィック量の上昇（タイプ 4）．
   【結果】：FTP クライアントでは，FTP 通信時にトラフィックが上昇し，誤検知が発生した．
   【考察】：ネットワークを利用するアプリケーションは，実行時に少なからずトラフィックが上昇する☆☆．このようなアプリケーションに対して，トラフィックがどれくらい増加したらワームであるかの閾値を適切に求めることは一般的に難しいため，閾値を低く設定してしまうと誤検知が，閾値を高く設定してしまうと検知漏れが発生する可能性がある．
5. CPU 利用率の上昇[10]
   【振舞い】：ワームは侵入，発病，感染の動作をなるべく速く行おうとする．
   【監視対象】：プロセスの CPU 利用率の上昇（タイプ 5）．
   【結果】：Ethereal の実行時には CPU 利用率が上昇し，誤検知が発生した．
   【考察】：多大な計算処理をともなうアプリケーションにおいては，CPU が寡占されることも多い．このようなアプリケーションに対して，CPU 利用率がどれくらい増加したらワームであるかの閾値を適切に求めることは一般的に難しいため，閾値を低く設定してしまうと誤検知が，閾値を高く設定してしまうと検知漏れが発生する可能性がある．
6. 送受信データ間の相関[11]
   【振舞い】：ワームは PC への感染後，他の PC へネットワークを介して自己複製（二次感染）を行う．
   【監視対象】：「外部から受信したデータ」と「外部へ送信するデータ」の相関の高さ（タイプ 6）．
   【結果】：ユーザが自分宛に届いたメールを友人などに転送する際に誤検知が発生した．
7. 本方式
   【振舞い】：ワームは自己複製のため，自分自身のファイルの READ を行う．
   【監視対象】：実行プログラムのパスとそのプログラムが稼働中に READ するファイルのパスの一

☆ アンチウイルスソフトも常駐型アプリケーションのカテゴリに含まれるが，ここでは特に個別に評価した．

☆☆ メーラにおいては，送受信を行ったメールの本数やファイルサイズが多大ではなかったため，誤検知には至らなかった．

表 3 誤検知および検知漏れの比較表
Table 3 Comparison results.

	ウイルスらしい振る舞い(監視対象)						
	タイプ 1	タイプ 2	タイプ 3	タイプ 4	タイプ 5	タイプ 6	**本方式**
インストーラ (常駐型)	誤検知	誤検知	×	×	×	×	×
メーラ	×	×	×	×	×	誤検知	×
FTP	×	×	×	誤検知	×	×	×
ブラウザ	×	×	誤検知	×	×	×	×
Ethereal	×	×	×	×	誤検知	×	×
インストーラ (アンチウイルス)	誤検知	誤検知	誤検知	誤検知	×	×	×
ワーム	○	○	検知漏	検知漏	検知漏	○	○
変異型ワーム	○	○	検知漏	検知漏	検知漏	×	○

致，および，当該ファイルのプログラム領域のすべての READ.

【結果】：インストーラやブラウザなど，自分自身のファイルの一部を READ する正規アプリケーションは存在するが，3.4 節で述べたように，プログラム領域のすべてを READ しているかどうかを検査することにより誤検知は回避できた．

以上の結果をまとめたものが**表 3** である．表中の ○ は検知を表す．× は，正規プロセスに対しては非検知を，ワームに関しては検知不能であることを表す．「誤検知」は，正規プロセスをワームだと検知したことを表している．「検知漏」は，一部，検知できないワームがあったことを表す．

表 3 より，それぞれの「ワームらしい振舞い」を個別に見た場合に，本方式は，他の方法と比べて，検知漏れが起こりにくく，かつ，誤検知が少ない方法であるということが分かる．特に，タイプ 1 ＋タイプ 6 の組合せ，またはタイプ 2 ＋タイプ 6 の組合せ（AND 条件）を用いたとしても，本方式の特徴である変異型ワームの検知能力を備えることはできないことに注意されたい．

4.2 本方式の適用範囲

本方式は，プロセスが自分自身の本体のファイルのプログラム領域のすべてを READ するかどうかをチェックするというものである．よって，本方式によって，検知可能なワームの要件として以下の 3 つがあげられる．

- 自分自身を READ する．
- プログラム領域をすべて READ する．
- READ の対象はファイルである．

よって，以下のようなワームは現時点では検知の対象とはなっていない．

1. 他のプロセスに自ファイルを READ させるワーム
2. プログラム領域の一部のみを READ するワーム
3. メモリ上にのみ存在するワーム

次項からはこれらのワームについて考察を行う．

4.2.1 他のプロセスに自ファイルを READ させるワーム

他のプロセスに自ファイルを READ させるワームとは，自らのファイルの伝播を他のプロセスに実行させるワームである．このタイプのワームは以下の 3 つに大別できる．

- 既存のプロセスの機能を用いて自ファイルの READ 依頼をするワーム

既存のプロセスの機能を用いて自ファイルの READ 依頼をするワームとは，たとえば Microsoft Outlook Express に自ファイルを READ させ，メール送信を行わせることにより自分を感染させるワームである．Microsoft Outlook Express は MAPI（Messaging Application Program Interface）をサポートしている．そこで本方式を拡張して，MAPI のメール送信 API である MAPISENDMAIL の添付ファイルを指定する部分を監視すれば，ワームから Microsoft Outlook Express への自己ファイル READ の依頼（ワームが Microsoft Outlook Express を介して自己ファイル READ を行う動作）をとらえることができる．汎用的な API を

通じて READ 依頼がなされるものであれば，同様の対応が可能である．

- 他のプロセスを起動し，自ファイルの READ 依頼をするワーム

他のプロセスを起動するワームとは，ワームが別の実行プロセスを起動し，起動した別プロセスに自分自身のファイルの READ を依頼するワームである．このようなワームに対しては，上記と同様に，本方式を拡張して，プロセスを起動する API である CreateProcess を監視し，CreateProcess によって起動されたプロセスが CreateProcess を発行したプロセスの実行ファイルを READ した場合にアラートをあげるようにすればよい．

- 2 つのプロセスが互いのファイルを送信しあうワーム

ワーム自体が 2 つで構成されており，それぞれのプロセスが互いのファイルを READ し，送信を行うというワームが考えられる．このようなワームに対しては，それぞれが独立に活動するため，ワームが自ファイルの READ を依頼することも，他のプロセスを起動することもない．よって，上述の方法では対処できない．これに対しては，任意のプログラムが実行中の他のプログラムを READ することを制限するといったアクセス制御を行うなどの対策を併用する必要がある．

4.2.2 プログラム領域のすべてを READ しないワーム

プログラム領域のすべてを READ しないワームとは，自ファイルの一部のみを READ して全体を再構成するワームである．このタイプのワームは以下の 2 つに大別できる．

- ジャンクコードを付加するワーム

ある程度のサイズのジャンクコードをワーム本体に付加することにより，「ファイル全体の中の一部に本体が隠されている」というワームが作成可能である．このようなワームは，1) ファイル全体の中からワーム本体の部分のみを READ し，2) 新たなジャンクコードを生成したうえで，3) 1 のワーム本体と 2 のジャンクコードを合わせたものを他の PC に感染させるという動作をする．このようなワームは，プログラムのエントリポイントが READ された場合にアラートをあげるという本方式の変形方式によって対処可能である．

- データ領域のデータから自分自身を再構成するワーム

任意のワーム（ワーム A とする）に対し，1) そのワームを複製し（これをワーム B とする），2) ワーム A のデータ領域にワーム B を挿入する，という操作によって，データ領域中に自分自身を宿したワーム（ワーム C）が作成可能である．ワーム C は，データ領域の中のワーム B を READ するだけで，自分自身を再構成できる．このようなワームは自己ファイル READ（プログラム領域のデータをすべて READ したかどうか）を検査するだけでなく，READ するデータの内容を監視する必要がある．

4.2.3 メモリ上にのみ存在するワーム

PC のメモリ上でのみ動作し，ファイルシステムの中にワーム本体のデータを書き込むことがない CodeRed のようなワームは，自己ファイル READ のイベントが発生しないため検知不可能である．しかし，このようなワームも，感染活動を行う際，メモリ上の自分自身を READ しているはずであるため，自己ファイル READ のアイディアを自己メモリ READ という形に拡張することで対処可能となる．

5. ま と め

自分自身のファイルを再び READ して自己複製を行うというワーム特有の振舞いに着目し，プロセスのファイルアクセスを監視することにより未知ワームの検知を行う方法を提案した．本方式は変異型ワームの検知に対しても効果的であると考えられる．

OS のファイルアクセスを観測するモニタツールを用いた基礎実験から，本方式の可能性が確認できた．今後は本方式を実装したうえで各種のワームに対する実証実験を行うことにより，実際の検知率，誤検知率，リアルタイム検知を行うにあたってのオーバヘッドなどを計測していく．また，自己ファイル READ をワームらしい振舞いとして追加することにより，ビヘイビアブロッキング法の性能がどれくらい改善できるか確かめていきたい．

謝辞 京都大学高倉弘喜先生，静岡大学長谷川孝博先生，情報通信研究機構三輪信介様には，ワームの検体および計算機の実行環境に関してご協力をいただいた．NTT データ坂本弘章様，NTT 情報流通プラットフォーム研究所星敬一様，井上芳隆様，富士通研究所鳥居悟様，近畿大学白石善明先生には，ワームの検体に関する情報を提供していただいた．NTT データ東川淳紀様，情報通信研究機構安藤類央様には方式の実装に関する情報を提供していただいた．ここに深く謝意を表する．また，本研究は一部，(財) セコム科学技術振興財団の研究助成を受けた．ここに謝意を表する．

参 考 文 献

1) 情報処理推進機構：未知ウイルス検出技術に関する調査．http://www.ipa.go.jp/security/fy15/reports/uvd/index.html
2) Computer Associates/Taras Malivanchuk: The Win32 worms: classification and possibility of heuristic detection. http://www.virusbtn.com/conference/vb2002/abstracts/heuristic.xml
3) FileMon. http://www.sysinternals.com/
4) サクラエディタ．http://sakura_editor.at.infoseek.co.jp/
5) メモリの掃除屋さん．http://www6.plala.or.jp/amasoft/soft/soft1/memcl.html
6) シマンテック/Nachenberg, C.: Behavior Blocking: The Next Step in Anti-Virus Protection/ビヘイビアブロッキング．http://www.securityfocus.com/infocus/1557
7) クワンタム・リープ・イノヴェーションズ・インコーポレーテッド/シュヌラー：コンピュータ・ウィルス・トラップ装置，特表平 10-501354 (1998).
8) Japan Network Security Association Dynamic Defense Working Group: ホストベースの IDS の概要と適用について．http://www.jnsa.org/active/houkoku/IDSBasic.pdf
9) 株式会社東芝/高橋俊成：コンピュータウィルス発生検出装置，方法，およびプログラム，特開 2003-241989 (2003).
10) 東日本電信電話株式会社/鈴木 晃：電子メール中継システム及び電子メール中継方法，特開 2002-314614 (2002).
11) 松本隆明，杉村友幸，鈴木功一，前田秀介，馬場達也，水野忠則，西垣正勝：送受信データ間の相関に基づく未知ワーム検知を利用した蔓延防止手法の提案，情報処理学会論文誌，Vol.47, No.6, pp.1941–1953 (2006).
12) 内閣官房情報セキュリティセンター：電子政府におけるセキュリティに配慮した OS を活用した情報システム等に関する調査研究．http://www.bits.go.jp/inquiry/pdf/secure_os_2004.pdf

(平成 18 年 11 月 27 日受付)
(平成 19 年 6 月 5 日採録)

松本 隆明（正会員）

1978 年東京工業大学大学院電子物理工学専攻修士課程修了．同年日本電信電話公社（現 NTT）入社．2003 年株式会社 NTT データ技術開発本部長．オペレーティングシステム，コンピュータアーキテクチャ，情報セキュリティに関する研究に従事．現在，NTT データ先端技術株式会社勤務．静岡大学客員教授．

鈴木 功一

2005 年静岡大学情報学部情報科学科卒業．2007 年同大学大学院修士課程修了．同年東芝ソリューション株式会社入社．在学中，情報セキュリティに関する研究に従事．

高見 知寛

2006 年静岡大学情報学部情報科学科卒業．現在，同大学大学院修士課程．情報セキュリティに関する研究に従事．

馬場 達也（正会員）

995 年慶應義塾大学理工学部電気工学科卒業．同年 NTT データ通信株式会社（現，株式会社 NTT データ）入社．同社技術開発本部にてネットワークセキュリティに関する研究に従事．現在，同社ビジネスソリューション事業本部勤務．著書に『マスタリング IPsec』（オライリー・ジャパン）がある．IEEE 会員．

前田 秀介（正会員）

2004 年電気通信大学大学院電気通信学研究科情報工学専攻博士前期課程修了．同年株式会社 NTT データ入社．現在，同社技術開発本部にてネットワークセキュリティに関する研究に従事．

水野　忠則（フェロー）

1945 年生．1969 年名古屋工業大学経営工学科卒業．同年三菱電機（株）入社．1993 年静岡大学工学部情報知識工学科教授．1996 年情報学部情報科学科教授．2006 年より創造科学技術大学院長．工学博士．情報ネットワーク，モバイルコンピューティング，ユビキタスコンピューティングに関する研究に従事．著訳書としては『コンピュータネットワーク』（日経 BP），『モダンオペレーティングシステム』（ピアソン・エデュケーション）等がある．電子情報通信学会，IEEE，ACM 各会員．情報処理学会フェロー．

西垣　正勝（正会員）

1990 年静岡大学工学部光電機械工学科卒業．1992 年同大学大学院修士課程修了．1995 年同博士課程修了．日本学術振興会特別研究員（PD）を経て，1996 年静岡大学情報学部助手．1999 年同講師，2001 年同助教授．2006 年より同大創造科学技術大学院助教授．2007 年より准教授．博士（工学）．情報セキュリティ，ニューラルネットワーク，回路シミュレーション等に関する研究に従事．