本資料について

- 本資料は下記の論文を基にして作成されたものです．文章の内容の正確さは保障できないため，正確な知識を求める方は原文を参照してください．
- 著者：小池　竜一，中谷　直司，厚井　祐司
- 論文名：未知コンピュータウイルスを駆除するUSBフラッシュメモリの開発
- 出展：情報処理学会論文誌 Vol.48 No.4
- 発表日：2007年4月

未知コンピュータウイルスを駆除するUSBフラッシュメモリの開発

名城大学　理工学部　情報工学科
渡邊研究室
040427180　三根　健司

はじめに（1）

- インターネットをはじめとするネットワークが急速に発展し，コンピュータウイルスによる被害が年々深刻化
- ウイルスを検出・駆除するためアンチウイルスの利用が原則
- アンチウイルスはウイルスの特徴を収めたシグネチャと，対象となるファイルをパターンマッチングすることでウイルスを検出（パターンマッチング方式）
- 新種の未知ウイルスが1日に約30種類発生
- アンチウイルスメーカがシグネチャ生成に平均10時間が必要

はじめに（2）

- シグネチャに含まれない未知ウイルスを検出するための様々な研究が行われてきた
- 完璧な未知ウイルス検出手法というものは存在せず，ユーザの一定数が未知ウイルスに感染してしまう事態は避けることができない
- 未知ウイルスに感染済みのPCに挿入することで，人手を介することなく自動的に未知ウイルスを駆除するUSBフラッシュメモリを開発

コンピュータウイルス

- 経済産業省の定義
  第三者のプログラムやデータベースに対して意図的に何らかの被害を及ぼすように作られたプログラムで，自己伝染機能，潜伏機能，発病機能のうち1つ以上を有するもの
- ワームやトロイの木馬を含めた，不利益をもたらす不正プログラム全体をウイルスと呼んでいる
- 実際のウイルスのほとんどがWindowsを対象としていることを考慮して，本システムでもWindows上で動作する実行ファイル形式のウイルスのみ対象

未知ウイルス駆除ソフトウェア自動生成システム

- 何らかの原因で未知ウイルスに感染してしまったPCからウイルスを検出し，さらにそのウイルスの駆除ソフトウェアを自動生成するためのシステム
  - システムはUSBフラッシュメモリ内に格納され，これを用いて感染済みPCをブートさせ独自の検証環境を構築
  - 感染済みPC内からウイルス候補ファイルを探し出し，それらを順次検証環境内で実行
  - その際の振舞いからウイルスを検出し，ワクチンの自動生成およびウイルスを駆除
- この一連の動作を，検証環境構築，ウイルス候補ファイル検索，未知ウイルス検出，ワクチン自動生成と定義

検証環境構築(1)

- 検証対象のPCをLinuxが格納されたUSBフラッシュメモリを用いて起動させ，LinuxをHost OSとしてVMwareを実行
- ターゲットである未知ウイルスは最新のWindows環境を対象としていると考えられるので，VMware上にはGuest OSとしてWindows XP SP2を標準インストール
- VMwareのスナップショット機能により，ウイルスに破壊された環境を即座に復元できるように設定

検証環境構築(2)

- VMware上のWindows XPはHost OSのみと通信可能とし，閉じたネットワークを構成
- ウイルスのネットワークを使った感染活動が阻害されないように，Host側にDNSを用意し，あらゆる名前解決に対してHost OSのIPアドレスを返すように設定
- ウイルスのメールによる感染活動を阻害されないように，Host OS側にメールサーバを用意し，あらゆるあて先のメールもローカルに配信するように設定

検証環境構築(3)

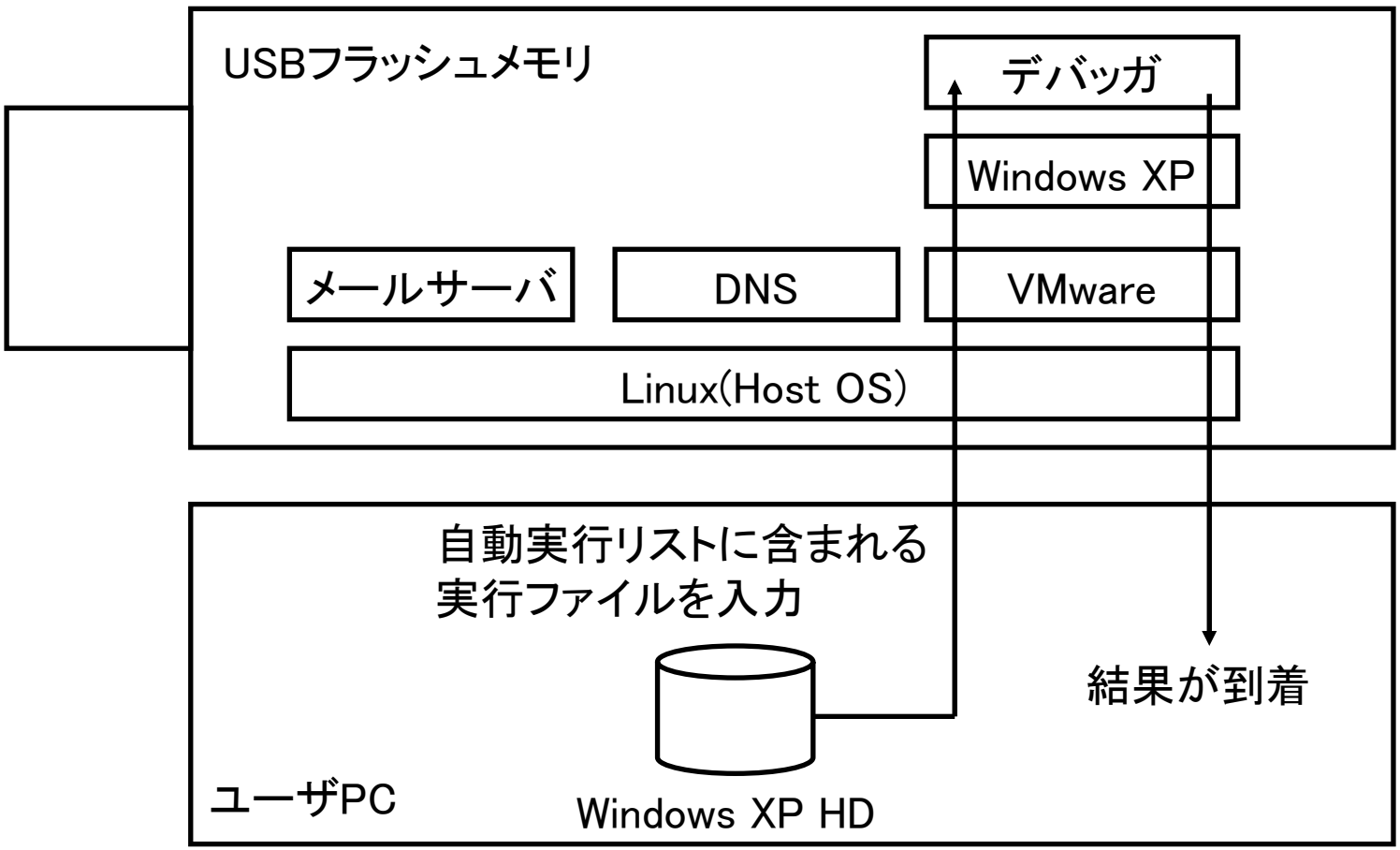

図1. 検証環境の構成

USBフラッシュメモリを選択した理由

- 本システムを利用可能なPCの数を最大化するため
- SDカードやメモリースティックの利用には専用のリーダが必要
- ウイルスの動作ログを記録するため，CD-R/DVD-R等の1度の書き込みが基本の記録媒体には不向き
- HDDは上記の記録媒体に比べ高価なので積極的に選択する必要はない

表1．本システムの構成

Linux	100MB
VMware	20MB
Windows XP	1700MB
合計	1820MB

（MB：メガバイト）

ウイルス候補ファイル検索

① Linuxはユーザ用Windowsのハードディスクをマウントし，レジストリから自動実行が行われる実行ファイルの検索を行い，それらをウイルス候補としてリスト化

② リスト中から1つ実行ファイルを選び，Linux側から検証用Windowsに渡す

③ 検証用Windows内で実行ファイルが実行され，未知ウイルス検出機能により活動を記録

④ 得られた動作情報を検証用Windows側からLinux側に渡し，次の検証に備えて検証用Windowsの環境を元の状態に復元

⑤ Linux側において，渡された動作の記録を解析し，ウイルスかどうか判定

⑥ 操作2.へ戻り，リスト中の全ての実行ファイルに対して繰り返す

未知ウイルス検出

- 未知ウイルスの検出にはパターンマッチングは使えないため，本システムではヒューリスティック手法を用いる
- ヒューリスティック手法は2つに分類される
  - 静的ヒューリスティック手法

    チェック対象のファイルのコード内の命令を解析して，実行するとどのような行動をするのか予測し，その動作がウイルスの動作と認められる場合にウイルスと判定
  - 動的ヒューリスティック手法

    特殊な環境下で実際に実行ファイルを動作させ，その動作がウイルスの動作と認められる場合にウイルスと判定
- ワクチン生成に利用可能なウイルスの動作情報を得るのに都合がよい動的ヒューリスティック手法をデバッグ機能を用いて実装

デバッガを用いた
動的ヒューリスティック手法（1）

- API（Apprication Programing Interface）
  Windowsで使用される関数や構造体，マクロなどの集合体で，DLL（Dynamic Link Library）が提供
- 基本的にWindowsの実行ファイルはAPIを経由してすべての処理を行っているので，API関数の呼び出しを取得することができれば，検証対象の動作をすべて検出することが可能
- API関数の呼び出しを横取りし，任意の動作を行わせる技術であるAPIフックを使用
- 本システムではWindowsが用意しているデバッグ機能を利用したAPIフックの手法を実装

デバッガを用いた
動的ヒューリスティック手法（2）

① 監視するAPI関数のブレークポイント設定に使用するアドレス情報を取得
② デバッグ対象の実行ファイルをデバッグ可能なフラグを立てて実行
③ 監視するAPI関数を含むDLLがロードされたら，取得したアドレス情報を用いて該当するAPI関数にブレークポイントを設定
④ ブレークポイントを検出したら，該当するAPI関数の引数を取得
⑤ 操作③，操作④を実行ファイルが終了，もしくは3分経過するまで繰り返す

監視対象API関数

表2. 監視する動作とAPI関数

監視する動作	提供するDLL	API関数	
ファイルの作成	Kernel32.dll	CreateFileA	CreateFileW
		CopyFileA	CopyFileW
		CopyFileExA	CopyFileExW
		WriteFile	
		WriteFileEx	
		CreateDirectoryA	CreateDirectoryW
		CreateDirectoryExA	CreateDirectoryExW
レジストリの変更	ADVAPI32.dll	RegOpenKeyA	RegOpenKeyW
		RegOpenKeyExA	RegOpenKeyExB
		RegCreateKeyA	RegCreateKeyW
		RegCreateKeyExA	RegCreateKeyExW
		RegSetValueA	RegSetValueW
		RegSetValueExA	RegSetValueExW

ワクチン自動生成

- 基本的には，ウイルスの動作を逆にすればよく，作成されたファイルを削除し，変更されたレジストリを元に戻せばウイルスは駆除可能
- ウイルスの動作にランダムな要素が含まれる場合，単純にウイルスの動作の逆を行うことができない
- ウイルスを識別する何らかのシグネチャを自動生成することが必要

シグネチャの自動生成

- 現在のウイルスの多くが共通して持っている固有な情報を抽出し，それをもとにシグネチャを生成することが必要
- Windowsの実行ファイルのヘッダ情報からシグネチャを自動生成する手法を提案
- Windowsの実行ファイルはPE（Portable Executable）形式と呼ばれるフォーマットに従って構成されている

シグネチャの自動生成

- すべての実行ファイルに存在するヘッダはCOFF（Common Object File Format）ヘッダとそれに続くオプション・ヘッダのみ
- この2つのヘッダ部分の各項目ごとに値が一致する確率を調査した結果，オプション・ヘッダ中の“Import Table”の値が最も一致しなかった（一致率0．02%）
- Import Tableまでのファイルの先頭からのオフセットは，MS-DOSヘッダの最後の部分に記述されている「PE＼0＼0」までのオフセット（可変：αバイトとする）と「PE＼0＼0」の先頭からImport Tableまでのオフセット128バイトの和となり，（128＋α）バイトとなる
- Import Tableのサイズは8バイトなので，生成されるシグネチャ（ITシグネチャ）は「ファイルの先頭から（128＋α）バイト目に続く8バイトがウイルスの同部分と等しい」となる

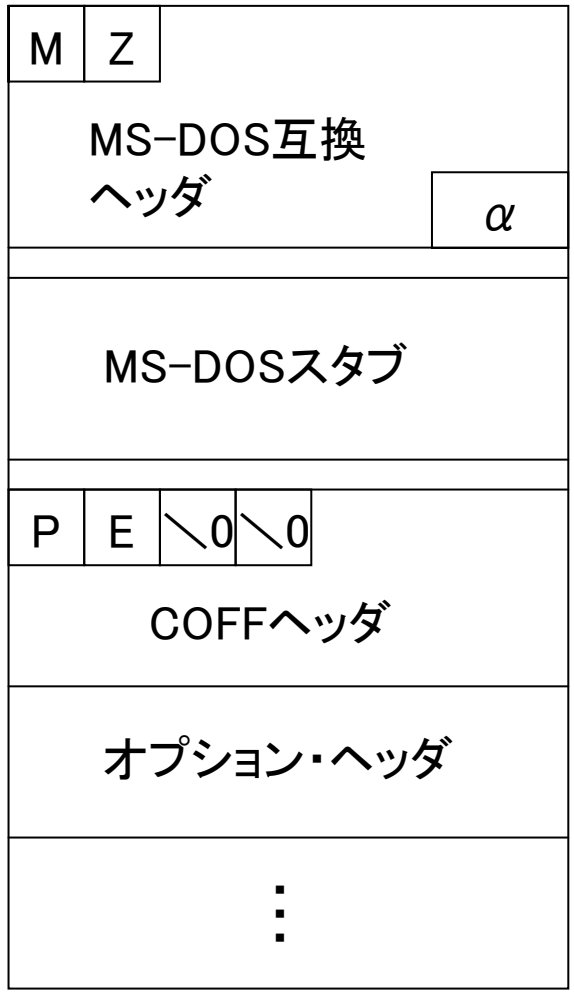

図2．PE形式の構造

ワクチンの構造

- 本システムではLinux上で動作するワクチンとWindows上で動作するワクチンとが同時に生成
  - 前者はユーザ用Windowsで発見された未知ウイルスをLinuxで動作する本システム側から即削除を行うためのもので，後者は他のPC向けに配布するためのもの
- ワクチンはファイルを削除しレジストリを元に戻す実行処理部分と動作を制御する定義部分とに分けて構成
- 定義部分が持つ内容
  - ITシグネチャ

    ヘッダ内のImport Tableをベースとしたシグネチャ
  - 削除すべきレジストリ

    ウイルスにより追加されたWindows起動時の自動実行設定等
  - 削除すべきファイルのあるフォルダ

    ウイルスにより追加されたファイルがあるフォルダを示す．ファイル名がランダムである場合が考えられるため，このフォルダ内をITシグネチャを用いたパターンマッチングで検索し，実際の削除すべきファイルを特定

ワクチンの動作

① 実行部分の直後に添付された定義部分から，駆除を行うウイルス固有の情報を読み出す

② レジストリの削除

(a)“削除すべきレジストリ”の情報をもとに，追加されたレジストリを削除する

(b)“削除すべきレジストリ”の情報がある間(②a)に戻り繰り返す

③ ファイルの削除

(a)“削除すべきファイルのあるフォルダ”内を“ITシグネチャ”を用いて検索し削除すべきファイルを特定する

(b)(③a)で特定されたファイルを使用しているプロセスを検索し，存在した場合はそのプロセスを停止する

(c)(③a)で特定されたファイルを削除する

(d)“ITシグネチャ”が一致するファイルがある間，(③a)に戻り繰り返す

④ “削除すべきファイルのあるフォルダ”の情報がある間③に戻り繰り返す

⑤ Windowsを再起動する

未知ウイルス検出実験（1）

- 対象とした実行ファイルは表のウイルスとWebサイトから無作為に選択したインストーラを含む50の通常の実行ファイル

表3．対象としたウイルス

ウイルス名	総数	ウイルス名	総数
Erkcz.B@mm	10	Mytob.V@mm	14
Mydoom.M@mm	52	Mytob.AF@mm	36
Mydoom.BO@mm	149	Mytob.AG@mm	75
Mydoom.BT@mm	16	Mytob.AH@mm	11
Mytob.B@mm	11	Mytob.AP@mm	136
Mytob.C@mm	69	Mytob.AS@mm	17
Mytob.M@mm	57	Mytob.CH@mm	53
Mytob.U@mm	245		

未知ウイルス検出実験(2)

表4. 未知ウイルス検出結果

ウイルス名	File	Registry	ウイルス判定	ウイルス名	File	Registry	ウイルス判定
Erkcz.B@mm	○	○	○	Mytob.V@mm	○	×	○
Mydoom.M@mm	○	○	○	Mytob.AF@mm	○	×	○
Mydoom.BO@mm	○	○	○	Mytob.AG@mm	○	×	○
Mydoom.BT@mm	×	○	○	Mytob.AH@mm	○	×	○
Mytob.B@mm	○	×	○	Mytob.AP@mm	○	×	○
Mytob.C@mm	×	×	×	Mytob.AS@mm	○	×	○
Mytob.M@mm	○	×	○	Mytob.CH@mm	○	○	○
Mytob.U@mm	○	×	○	通常の実行ファイル	×	×	×

未知ウイルス検出実験(3)

- Mytob.C@mmが検出できずウイルスと判定できず，他のMytob亜種でもMytob.CH@mm以外はレジストリの変更が検出できなかった
  - UPXをはじめとするPackerと呼ばれるソフトウェアによって圧縮され，アンチデバッグ機能が付加されたため
- 通常ファイル50個すべてに対して誤検出することはなかった

ワクチンの自動生成実験(1)

表5. ITシグネチャの生成結果

ウイルス名	総数	ITシグネチャ	MD5	ウイルス名	総数	ITシグネチャ	MD5
Erkcz.B@mm	10	1	7	Mytob.V@mm	14	1	2
Mydoom.M@mm	52	1	51	Mytob.AF@mm	36	2	4
Mydoom.BO@mm	149	1	1	Mytob.AG@mm	75	1	11
Mydoom.BT@mm	16	1	1	Mytob.AH@mm	11	4	4
Mytob.B@mm	11	1	1	Mytob.AP@mm	136	1	1
Mytob.C@mm	69	2	3	Mytob.AS@mm	17	2	2
Mytob.M@mm	57	1	3	Mytob.CH@mm	53	1	1
Mytob.U@mm	245	1	20				

ワクチンの自動生成実験(2)

表6. ITシグネチャによる誤検出ファイル数

ITシグネチャ		スキャン対象ウイルス	
		Mytob.M	Mytob.AF
Mytob.M@mm			1
Mytob.AF@mm	1	57	
	2		

ウイルス駆除実験

表7．ワクチンによる駆除判定

ウイルス名	File	Registry	駆除判定	ウイルス名	File	Registry	駆除判定
Erkcz.B@mm	○	○	○	Mytob.V@mm	○	－	○
Mydoom.M@mm	○	○	○	Mytob.AF@mm	○	－	○
Mydoom.BO@mm	○	○	○	Mytob.AG@mm	○	－	○
Mydoom.BT@mm	－	○	×	Mytob.AH@mm	○	－	○
Mytob.B@mm	○	－	○	Mytob.AP@mm	○	－	○
Mytob.C@mm	－	－	－	Mytob.AS@mm	○	－	○
Mytob.M@mm	○	－	○	Mytob.CH@mm	○	○	○
Mytob.U@mm	○	－	○				

おわりに

- ワクチンを自動生成しウイルス駆除を行うUSBフラッシュメモリを開発
- 実験では通常のファイルを誤検出し誤って駆除することなく，１５種類のウイルスのうち１３種類のウイルスを駆除することができた
- 今後の課題
  - 動的ヒューリスティック手法の精度の向上
  - 本システムを未知ウイルス検出専用のLinuxディストリビューションとして一般配布可能な形へ改良