Nish!gaki Laboratory

自己ファイルREAD/DELETEの検出による ボット検知の可能性に関する一検討

静岡大学　酒井　崇裕
静岡大学　長谷　巧
KDDI研究所　竹森　敬祐
静岡大学　西垣　正勝

Nish!gaki Laboratory

発表の流れ

- 背景
- ボット検知方式の検討
  - ワームの既存検知方式
  - ボットの侵入時の特徴
- 提案方式
- 検証実験
- 考察

背景(1/3)

Nish!gaki Laboratory

- 「ボット検知の困難性」
  ボットはＰＣ内に潜伏する
  - ファイル削除、強制シャットダウンなど、目立った行動はしない
  - 他のＰＣへの攻撃、感染活動において正規の通信になりすます

感染端末に潜んでいるボットを検出することは難しい

背景(2/3)

Nish!gaki Laboratory

- 「ボット検知のアプローチ」
  - 潜伏しているボットを見つけるのは難しい

  - ボットのＰＣへの侵入挙動をとらえるべき

そのアプローチのために、ボットの特徴を考える

背景(3/3)

Nish!gaki Laboratory

- 「ボットについて」
  - ボットの機能は既存のマルウェア･攻撃の集合体
    - ＰＣへの侵入　→　ワーム
    - 他PCへの感染　→　ワーム
    - 情報収集　→　スパイウェア
    - DDoS攻撃　→　踏み台攻撃
    - スパムメール　→　踏み台攻撃

PCへの侵入におけるワームの検知方式がボットに適用可能かを検討する

発表の流れ

Nish!gaki Laboratory

- 背景
- ボット検知方式の検討
  - ワームの既存検知方式
  - ボットの侵入時の特徴
- 提案方式
- 検証実験
- 考察

ワーム検知の既存方式

Nish!gaki Laboratory

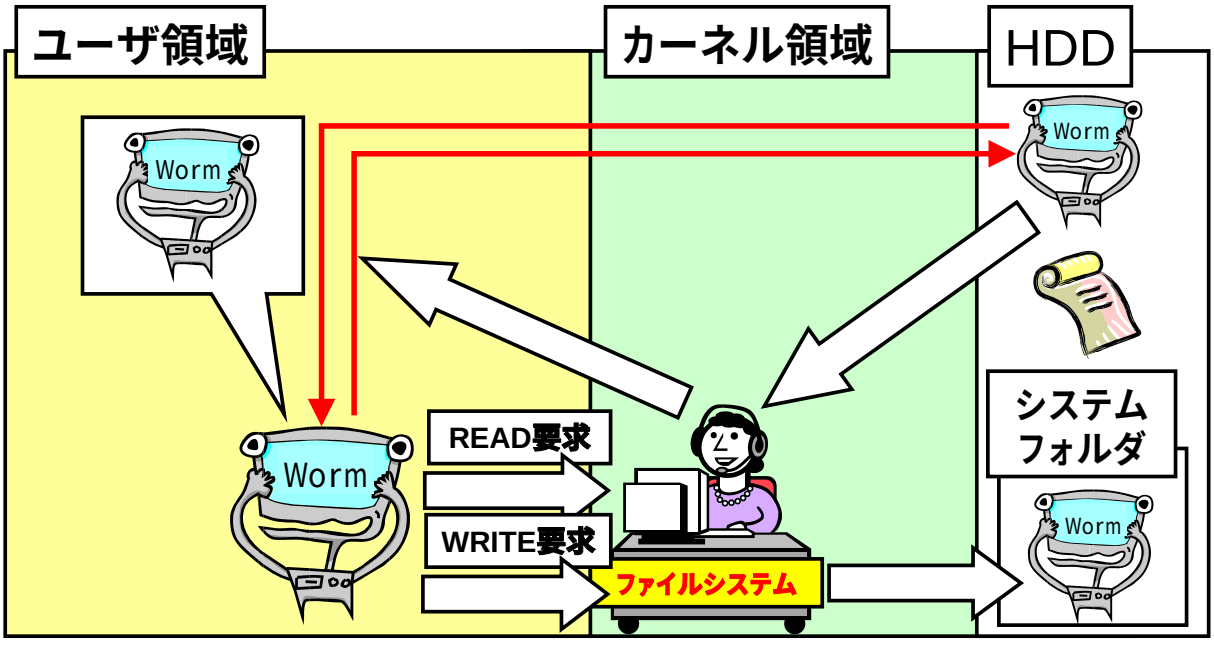
- 侵入の挙動におけるワーム検知

『自己ファイルREADの検出による未知ワームの検知方式』

– ワームはPCへの侵入時、システムフォルダ内に自己ファイルをコピーするため、自己ファイルをREADする

ワームのPC内への侵入

Nish!gaki Laboratory



自己ファイルREAD

Nish!gaki Laboratory

- ワーム
  - PC内への自己複製のために自分自身のファイルのすべてをREADする
- 正規のプログラム
  - 自分自身のファイルをすべてREADすることはない

自分自身のファイルのすべてをREADするという挙動を
自己ファイルREADと定義し、
自己ファイルREADの検出による未知ワーム検知

発表の流れ

Nish!gaki Laboratory

- 背景
- ボット検知方式の検討
  - ワームの既存検知方式
  - ボットの侵入時の特徴
- 提案方式
- 検証実験
- 考察

ボットの侵入時の特徴

Nish!gaki Laboratory

ボット

ワーム

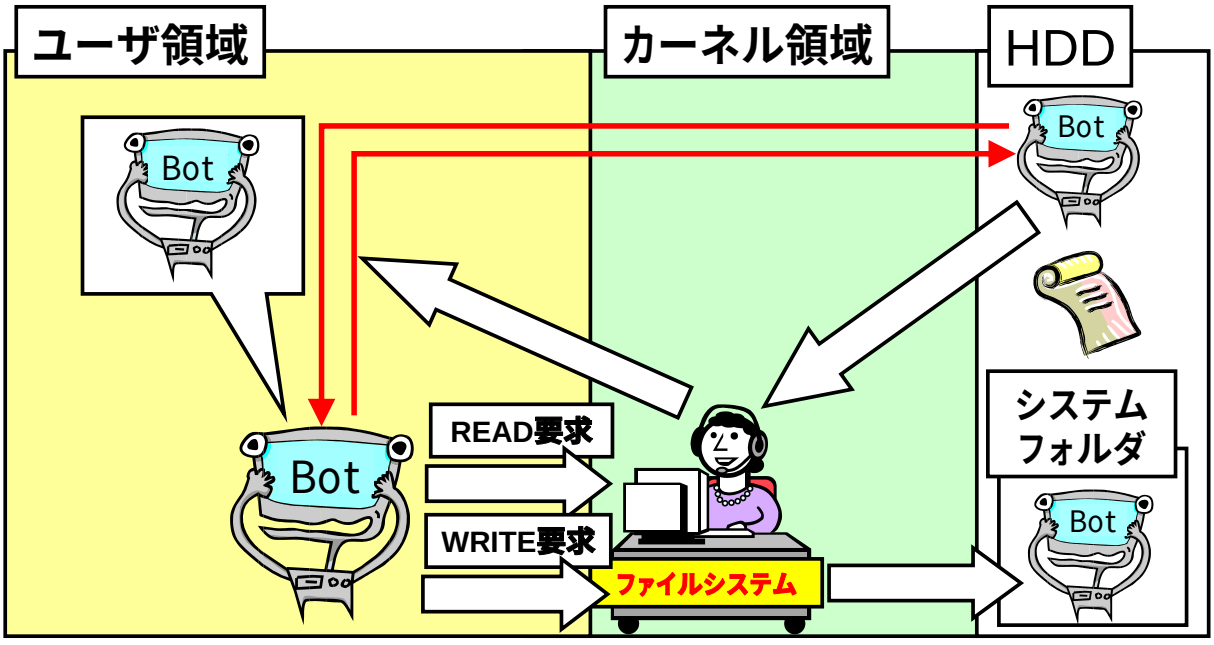
「自己複製」…システムフォルダ内に自己ファイルをコピー

＋

「自己隠蔽」…オリジナルの自己ファイルの削除

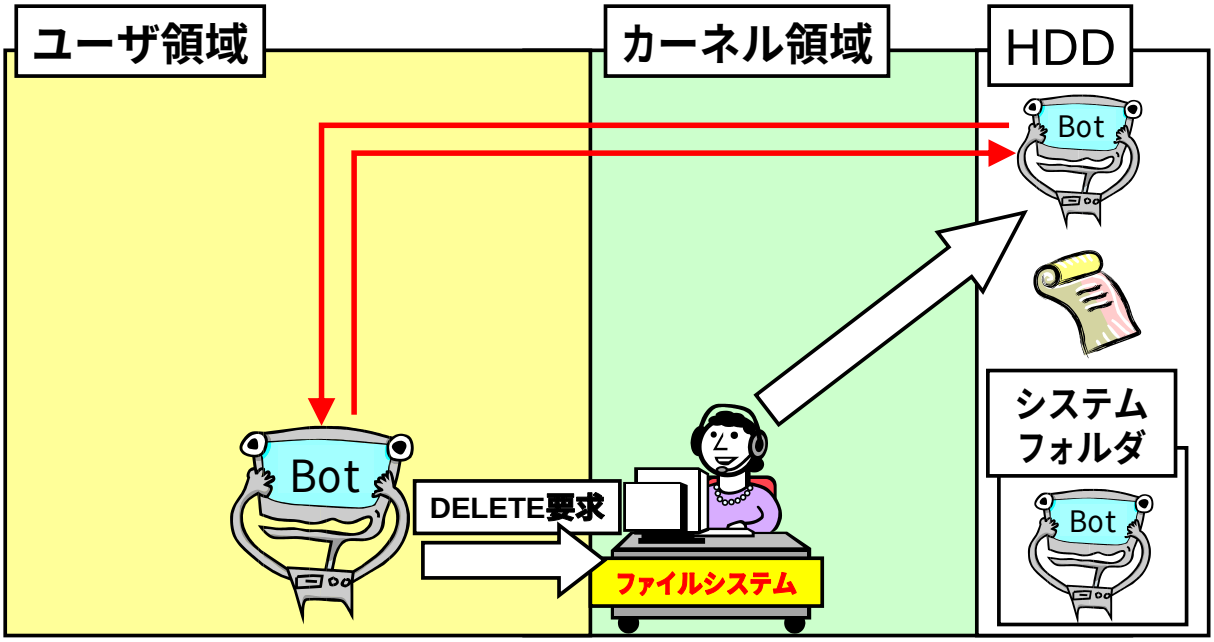
自己複製

Nish!gaki Laboratory



ボットの自己隠蔽

Nish!gaki Laboratory



ボットの侵入時の特徴

Nish!gaki Laboratory

ボット

ワーム

「自己複製」…システムフォルダ内に自己ファイルをコピー

⇒ 自己ファイルREAD

＋

「自己隠蔽」…オリジナルの自己ファイルの削除

⇒ 自己ファイルDELETE

自己ファイルへの挙動による検知方式

発表の流れ

Nish!gaki Laboratory

- 背景
- ボット検知方式の検討
  - ワームの既存検知方式
  - ボットの侵入時の特徴
- **提案方式**
- 検証実験
- 考察

提案方式

Nish!gaki Laboratory

- 自己ファイルREAD/DELETEの検出による ボット検知方式の提案
  - 実行プログラムとREAD/DELETEされるファイル のパスの相関
  - 実行プログラムとREADされるデータの領域

OSのファイルシステムを監視することにより リアルタイムで検知可能

提案方式（検知アルゴリズム）

Nish!gaki Laboratory

Step1. PC内で発生したすべてのファイルアクセスをフック

Step2. ファイルアクセスの中でREAD/DELETEに関するもののみを検出

Step3. Step2で検出したファイルアクセスを発生させたプロセスのパスと，アクセスするファイルのパスの相関をチェック

Step4. Step3で検出したプロセスがプロセス自身のファイル（の大部分）をREADまたはDELETEしていた時，そのプロセスをボットの疑いありと判定

提案方式（パスの相関）

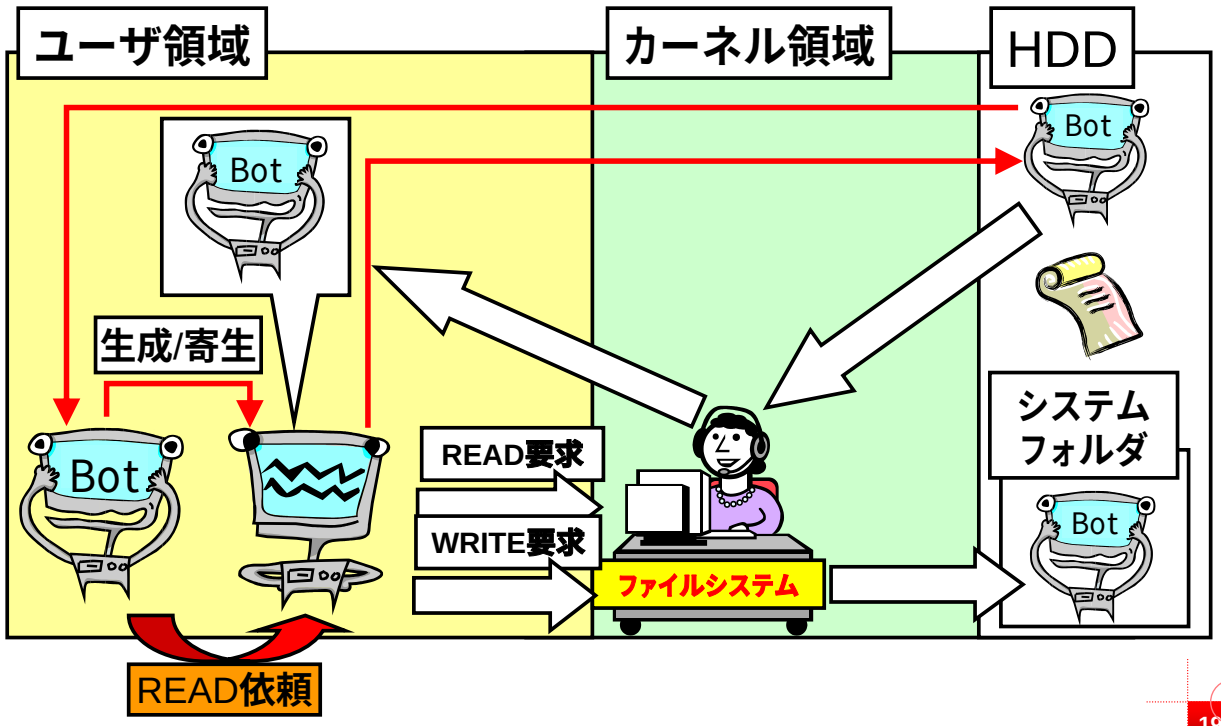
Nish!gaki Laboratory

- **単純な自己ファイルアクセス**
  - プロセスのパスとアクセスしたファイルのパスが同一

- **間接的な自己ファイルアクセス**
  - 他プロセスを介した自己ファイルアクセス
    - ⑴ 別プロセスを起動し、自己ファイルのREAD/DELETEを依頼
    - ⑵ 他の既存プロセス、実行ファイルに寄生し、自己ファイルのREAD/DELETEを依頼

間接型自己ファイルREAD

Nish!gaki Laboratory



提案方式（パスの相関）

Nish!gaki Laboratory

- 直接的な自己ファイルアクセス
  - プロセスのパスとアクセスしたファイルが同一

間接型自己ファイルREAD/DELETE

- 間接的な自己ファイルアクセス
  - 他プロセスを介した自己ファイルアクセス
    - ⑴ 別プロセスを起動し、
      自己ファイルのREAD/DELETEを依頼
    - ⑵ 他の既存プロセスに寄生し、
      自己ファイルのREAD/DELETEを依頼

発表の流れ

Nish!gaki Laboratory

- 背景
- ボット検知方式の検討
  - ワームの既存検知方式
  - ボットの侵入時の特徴
- 提案方式
- 検証実験
- 考察

検証実験

Nish!gaki Laboratory

- ProcMonを用いた有効性の評価

ファイルアクセス、プロセス生起が監視可能なモニタツール

  - ボットを用いて検知実験
  - 正規プログラムを用いた誤検知実験

- 実験環境
  - 仮想マシン（VMWare Workstation6）
  - ゲストOS：WindowsXP Professional SP2
  - 隔離されたローカルマシン

検知実験

Nish!gaki Laboratory

- ボットを用いてファイルアクセス、プロセス生起を観測し、自己ファイルREAD/DELETEされるかを測定

- 実験に用いたボット
  - CCC2008のマルウェア検体を含む9個の検体

検知実験結果

Nish!gaki Laboratory

ボット名	自己ファイルREAD	自己ファイルDELETE
検体A	○	○ (間接型)
検体B	○	○ (間接型)
検体C	○	○ (間接型)
検体D	○	○ (間接型)
検体E	○	○ (間接型)
検体F	○	○ (間接型)
検体G	○	○ (間接型)
検体H	○	○ (間接型)
検体I (CCC2008検体)	○	検知漏れ

自己ファイルREAD観測結果の一部

Nish!gaki Laboratory

プロセス名	ｵﾍﾟﾚｰｼｮﾝ	パス	詳細
0689(省略).exe	CreateFile	C:¥WINDOWS¥system32¥lsass.exe	(省略)
0689(省略).exe	ReadFile	C:¥0689(省略).exe	Offset:0, Length:65,536
0689(省略).exe	WriteFile	C:¥WINDOWS¥system32¥lsass.exe	Offset:0, Length:65,536
0689(省略).exe	ReadFile	C:¥0689(省略).exe	Offset:65,536, Length:65,536
0689(省略).exe	WriteFile	C:¥WINDOWS¥system32¥lsass.exe	Offset:65,536, Length:65,536
：	：	：	：
0689(省略).exe	ReadFile	C:¥0689(省略).exe	Offset:262,144, Length:7,680
0689(省略).exe	WriteFile	C:¥WINDOWS¥system32¥lsass.exe	Offset:262,144, Length:7,680

END OF FILE

自己ファイルDELETE観測結果の一部

Nish!gaki Laboratory

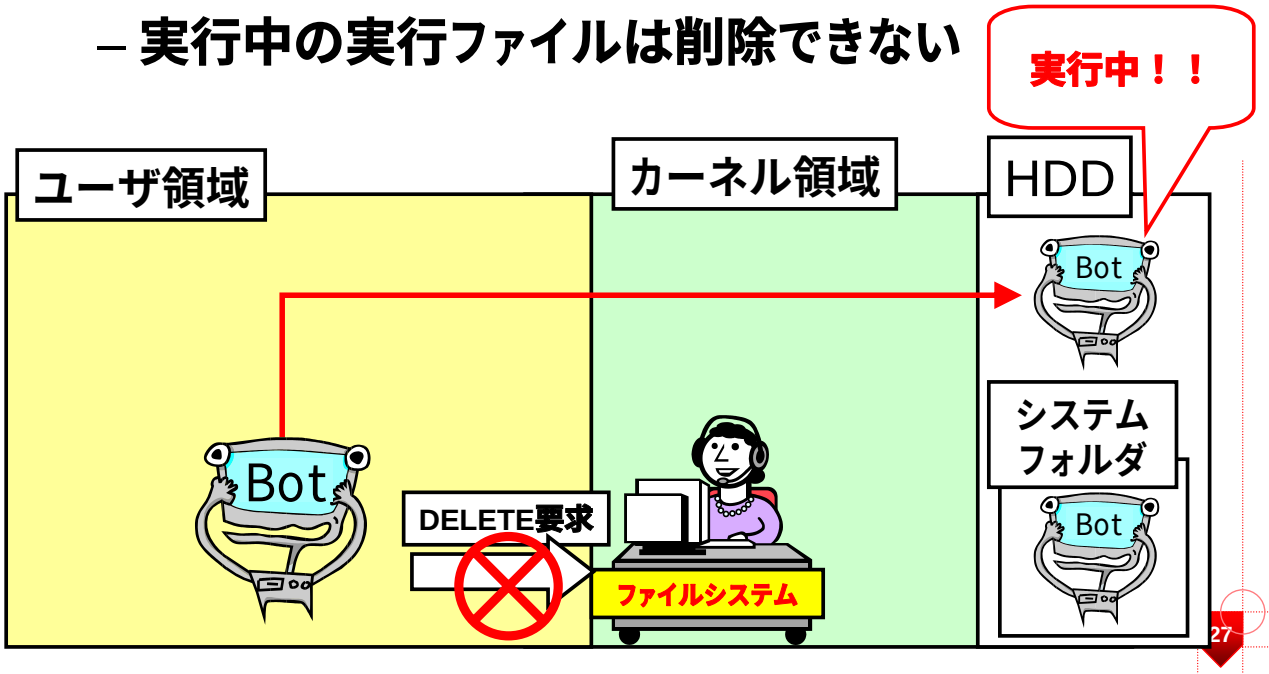
プロセス名	オペレーション	パス	詳細
0689(省略).exe	CreateFile	C:¥oeoeemct.bat	(省略)
0689(省略).exe	WriteFile	C:¥oeoeemct.bat	Offset: 0, Length: 202
0689(省略).exe	Process Create	C:¥WINDOWS¥system32¥cmd.exe	Command line: cmd /c "C:¥oeoeemct.bat" "
cmd.exe	Process Start		Parent PID: 1424
0689(省略).exe	Process Exit		(省略)
cmd.exe	SetDispositionInformationFile	C:¥0689(省略).exe	Delete: True
cmd.exe	SetDispositionInformationFile	C:¥oeoeemct.bat	Delete: True

0689(省略).exeの子プロセス

検知実験に関する考察(1)

Nish!gaki Laboratory

- 自己ファイルDELETEは間接型のみ
  - 実行中の実行ファイルは削除できない

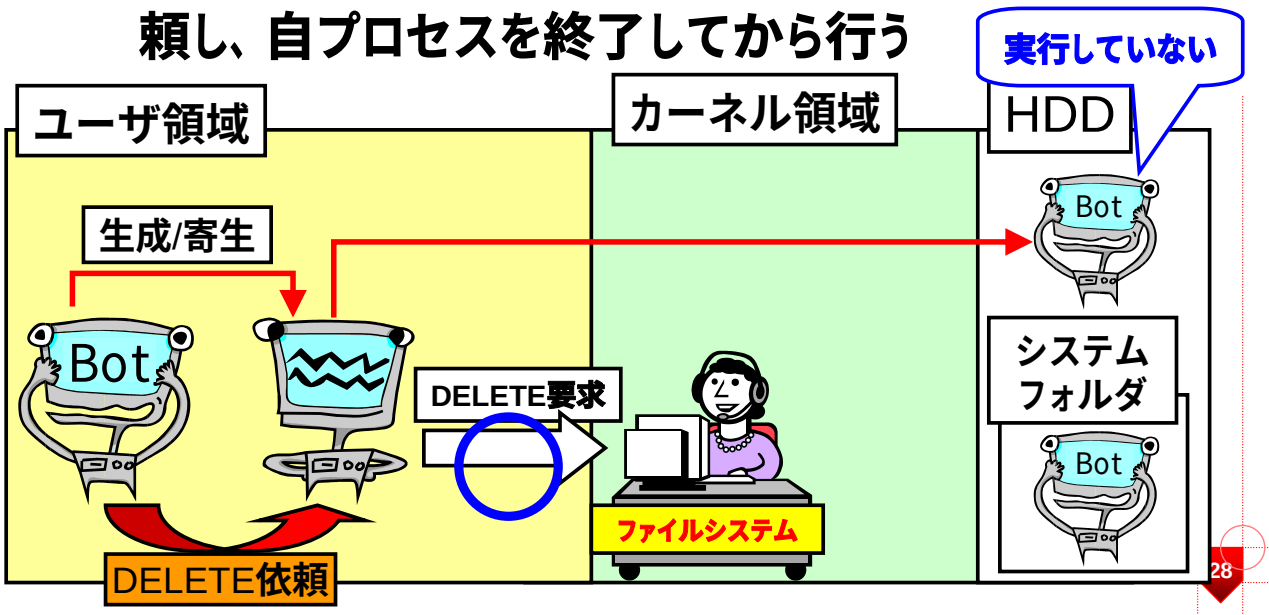


検知実験に関する考察(1)

Nish!gaki Laboratory

- 自己ファイルDELETEは間接型のみ

  - 自己ファイルのDELETEは、別のプロセスに依頼し、自プロセスを終了してから行う



検知実験結果

Nish!gaki Laboratory

ボット名	自己ファイルREAD	自己ファイルDELETE
検体A	○	○（間接型）
検体B	○	○（間接型）
検体C	○	○（間接型）
検体D	○	○（間接型）
検体E	○	○（間接型）
検体F	○	○（間接型）
検体G	○	○（間接型）
検体H	○	○（間接型）
検体I（CCC2008検体）	○	検知漏れ

検知実験に関する考察(2)

Nish!gaki Laboratory

- CCC2008検体における耐「ボット検知」機能

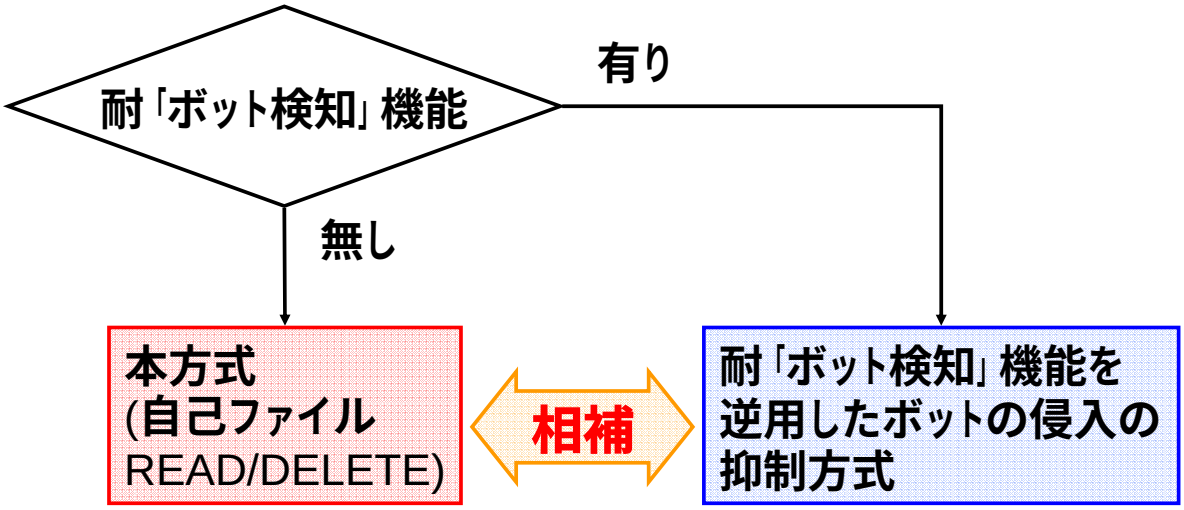
監視ツールを検知した時点で
PCへの侵入活動を強制終了

耐「ボット検知」機能を逆用したボットの侵入の抑制

– 「悪性プログラムの耐解析機能を逆用した活動抑制手法の提案」（松木さんら）

検知実験に関する考察(2)

Nish!gaki Laboratory



誤検知実験

Nish!gaki Laboratory

- **正規プログラムのファイルアクセスを観測し、自己ファイルREAD/DELETEされるかを測定**
- **実験に用いた正規プログラム**
  - MS WORD
  - MS EXCEL
  - Internet Explorer
  - Adobe Reader
  - **インストーラ/アンインストーラ**

誤検知実験結果

Nish!gaki Laboratory

正規プログラム	自己ファイルREAD	自己ファイルDELETE
MS WORD	○	○
MS EXCEL	○	○
Internet Explorer	○	○
Adobe Reader	○	○
インストーラ (wireshark-setup-1.0.2.exe)	誤検知	○
アンインストーラ (wiresharkのアンインストーラ	誤検知	誤検知 (間接型)
Windows Installer (apache_2.2.9-win32 -x86-no_ssl-r2.msi)	○	○

インストーラの誤検知

Nish!gaki Laboratory

- Winsowsが提供しているインストーラ
  - インストールファイル(インストール情報)を Windows InstallerがREADする

- アプリケーション固有のインストーラ
  - インストーラ自体がインストール情報であるため、自分自身をREADする ⇒ 自己ファイルREAD

アンインストーラの誤検知

Nish!gaki Laboratory

- Windowsが提供しているアンインストーラ
  - Windows Installerが適宜アプリケーション構成ファイルを削除する
- アプリケーション固有のアンインストーラ
  - 自分自身も含むアプリケーション構成ファイルを削除する ⇨ 自己ファイルDELETE
  - 自分自身を削除するために、テンポラリファイルに自己複製を作成する ⇨ 自己ファイルREAD

発表の流れ

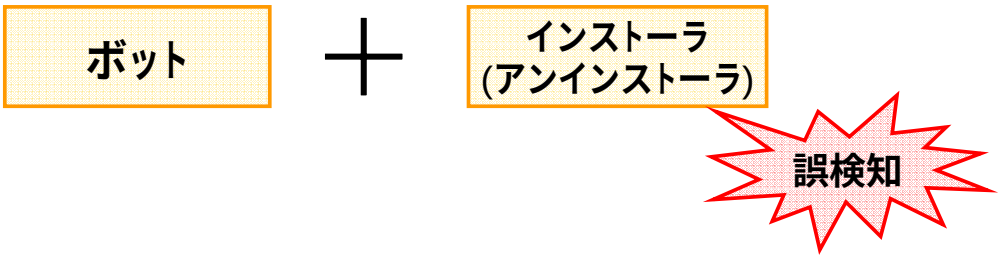
Nish!gaki Laboratory

- 背景
- ボット検知方式の検討
  - ワームの既存検知方式
  - ボットの侵入時の特徴
- 提案方式
- 検証実験
- 考察

考察

Nish!gaki Laboratory

- 本方式での検出されるプログラム



インストーラをボットと切り分ける必要がある

インストーラの切り分け

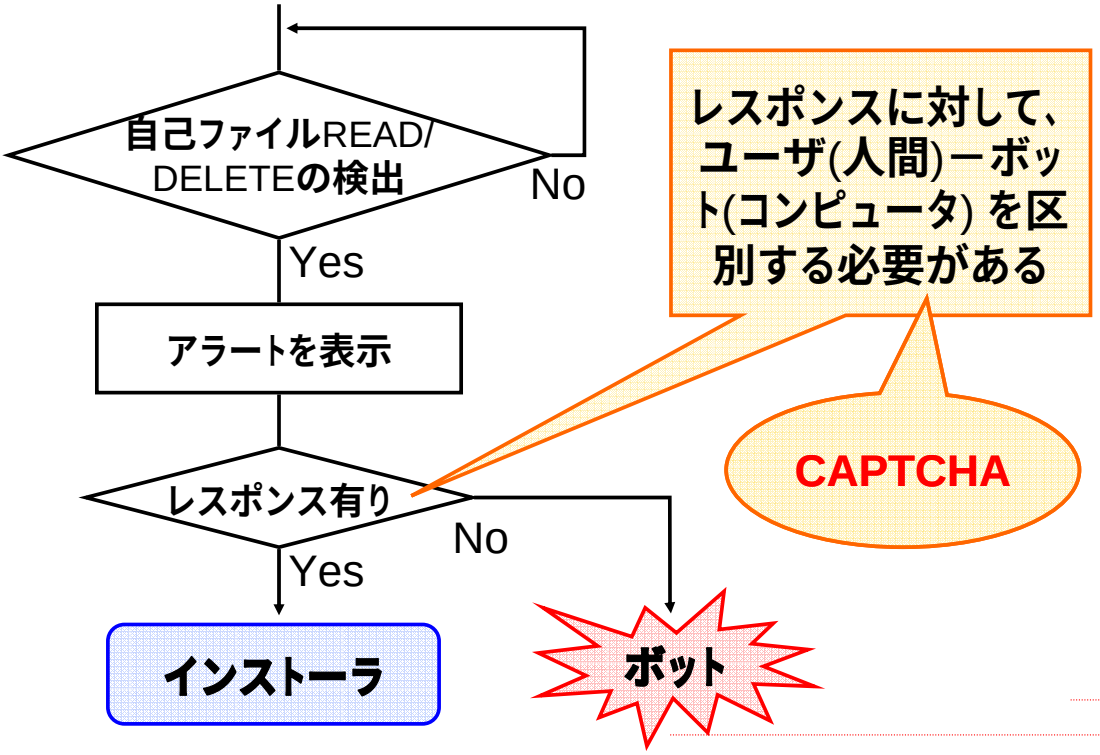
Nish!gaki Laboratory

- 一般にアプリケーションのインストールは
  ユーザの意思によって行われる

- インストールには必ずユーザが介入する

「ユーザのレスポンスによる判定」

ユーザのレスポンスによる判定



まとめ

Nish!gaki Laboratory

自己ファイルREAD/DELETEによるボット検知

– 多くのボットに対して有効

検知漏 **耐「ボット検知」機能を持つボット**

→ 耐「ボット検知」機能を逆用したボットの侵入の抑制による対策

誤検知 **インストーラ/アンインストーラ**

→ ユーザからのレスポンスを義務付けることでボットとの切り分け