Nish!gaki Laboratory

侵入挙動の反復性による ボット検知方式

静岡大学　酒井　崇裕
KDDI研究所　竹森　敬祐
NICT　安藤　類央
静岡大学　西垣　正勝

ボットの検知技術

Nish!gaki Laboratory

- **「パターンマッチング法」**
  - **ボットのバイトパターン**を定義し、マッチングすることで検出する
- **「ビヘイビアブロッキング法」**
  - **ボットの振る舞い**を定義し、その振る舞いを行っているかを監視することで検出する

パターンマッチング法

Nish!gaki Laboratory

- **「一般的なアンチウイルスソフトの主流」**
  - **既知のボット検知**にあたり、**簡素**かつ**確実**

しかし

- **「ボットの傾向」**
  - **様々な亜種**が生成
  - 暗号化、難読化技術による**自己改変**

そのため

パターンファイルにない未知のボットに対応できない

ビヘイビアブロッキング法

Nish!gaki Laboratory

- ビヘイビアから「**ボットらしさ**」を定義することで、
  未知のボットに対しても有効

しかし

- 「**ボットの挙動隠蔽**」

不正者は長期間にわたって、
ボットが感染したパソコンを悪用する

  - ファイル破棄やシャットダウン、大規模感染活動など、
    表立った行動をしない
  - 通信プロトコルや通信量の制御による
    正規通信へのなりすまし

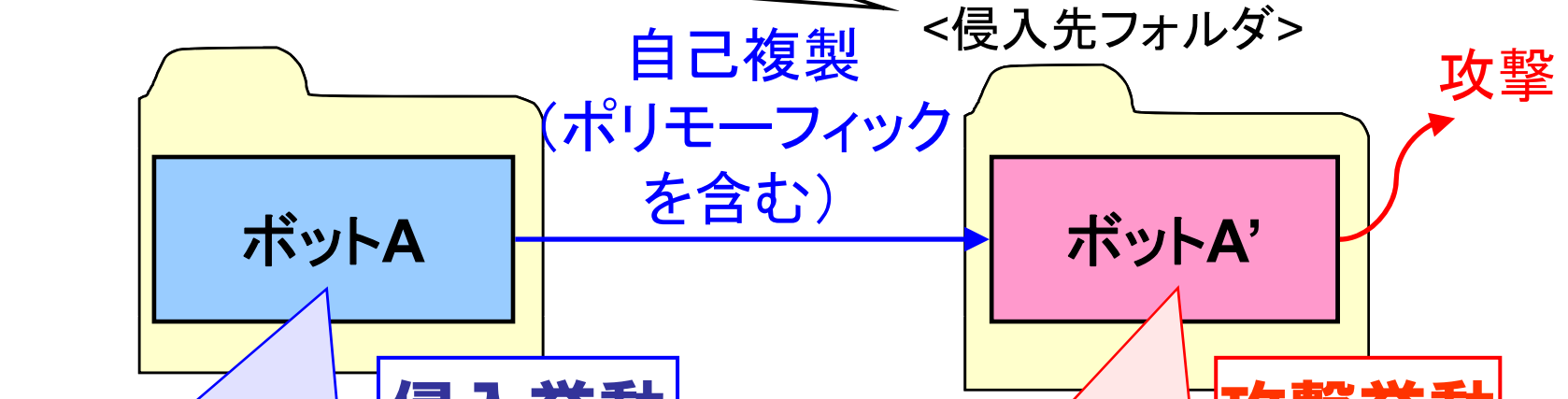
そのため

ボットの本質を捉えたビヘイビアの発見が必要

ボットの挙動の分析

Nish!gaki Laboratory

「システムフォルダへの潜伏」
・一般ユーザはシステムフォルダの構成、プロセスを把握していない
・多数のEXE,DLLが存在しているため、ボットの追加に気づきにくい



侵入挙動

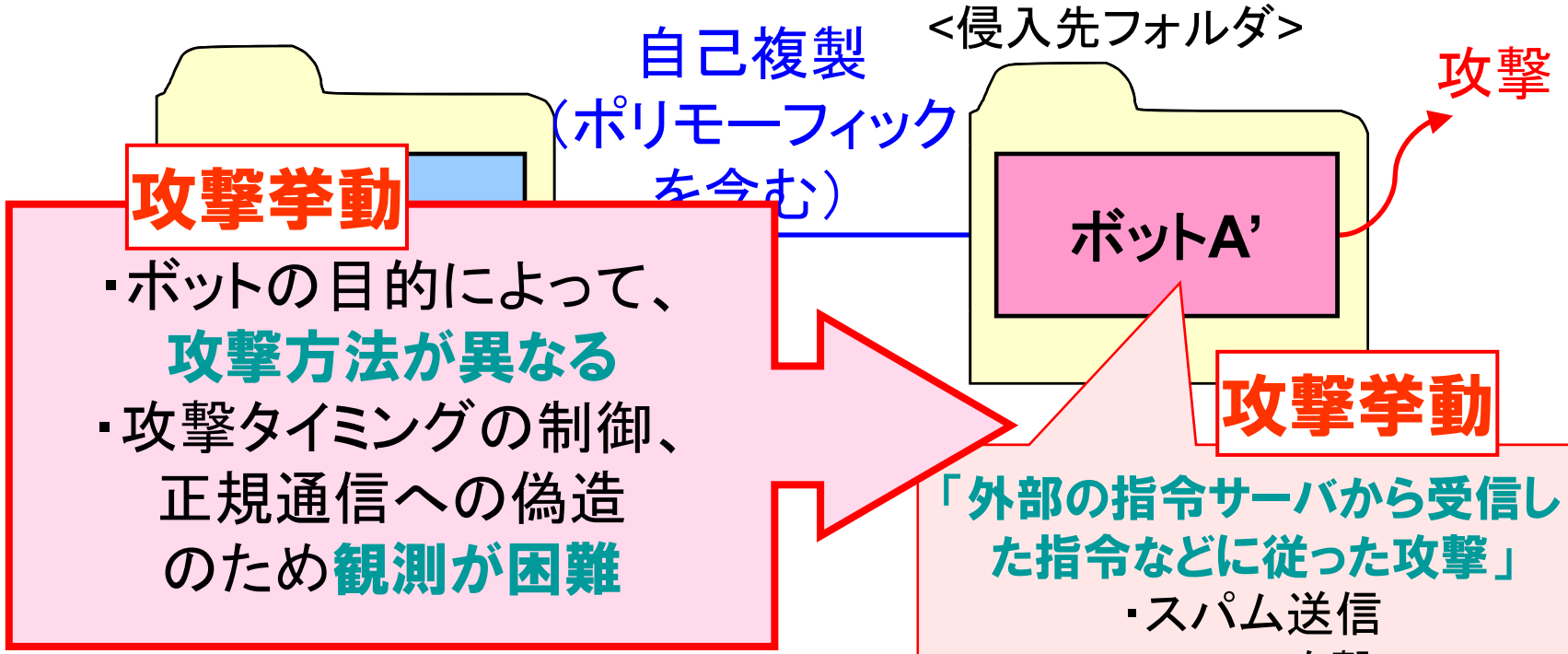
「潜伏のための環境を整える」
・侵入先フォルダへの実行ファイルの生成（自己複製など）
・OS上のファイル・レジストリの書き換え

攻撃挙動

「外部の指令サーバから受信した指令などに従った攻撃」
・スパム送信
・DDoS攻撃
・外部へのボット拡散活動

ボットの挙動の分析～攻撃挙動～

Nish!gaki Laboratory



ボットの挙動の分析～侵入挙動～

Nish!gaki Laboratory

侵入挙動に注目する

(1)ファイル生成

↓

(2)自動実行登録

侵入挙動

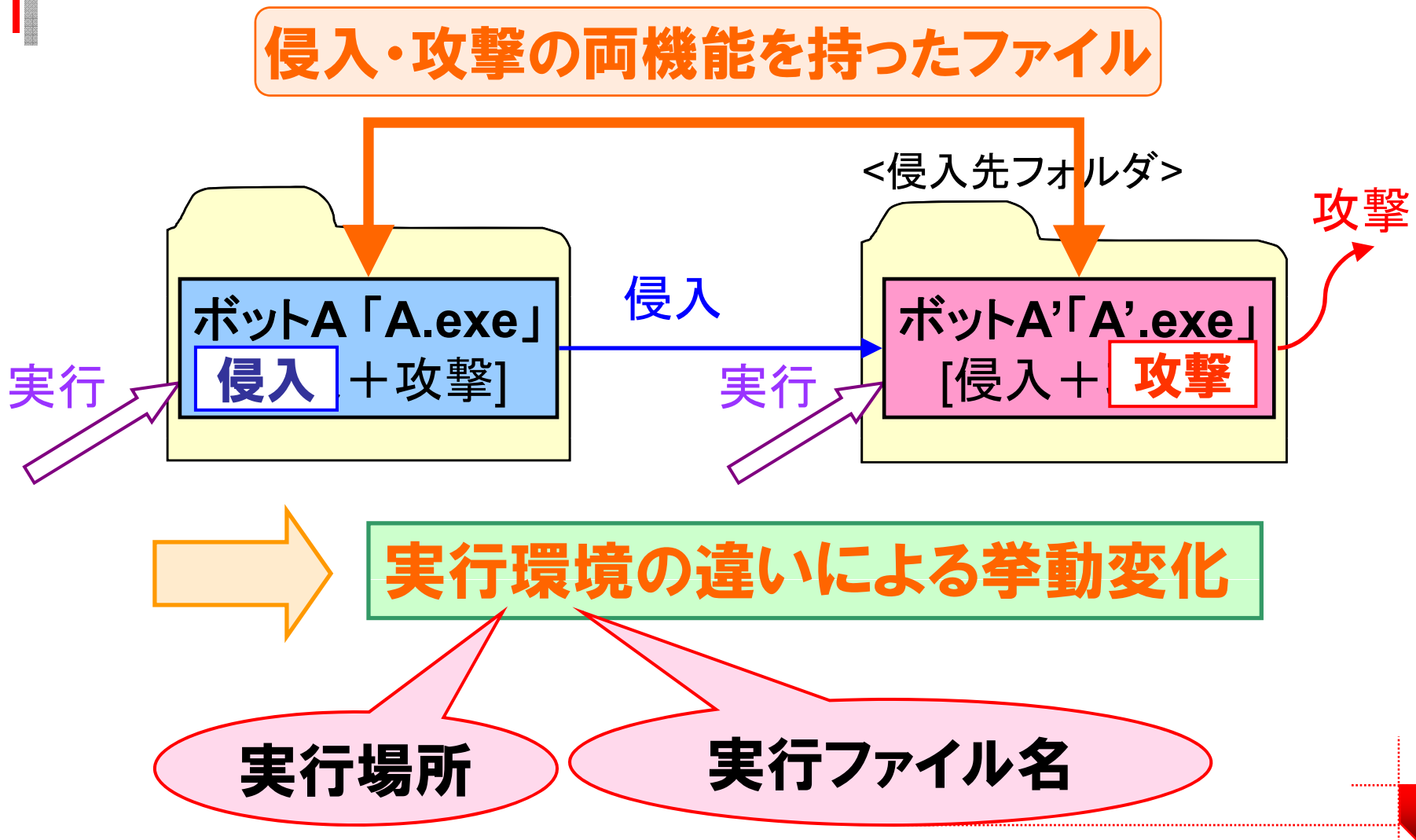
・潜伏/常駐するためには、
システムフォルダ等への侵入/自動実行登録は重要
・侵入は初めて実行された際に行われるため、
観測ポイントが一定

侵入挙動

「潜伏のための環境を整える」
・侵入先フォルダへの実行ファイルの生成（自己複製など）
・OS上のファイル・レジストリの書き換え

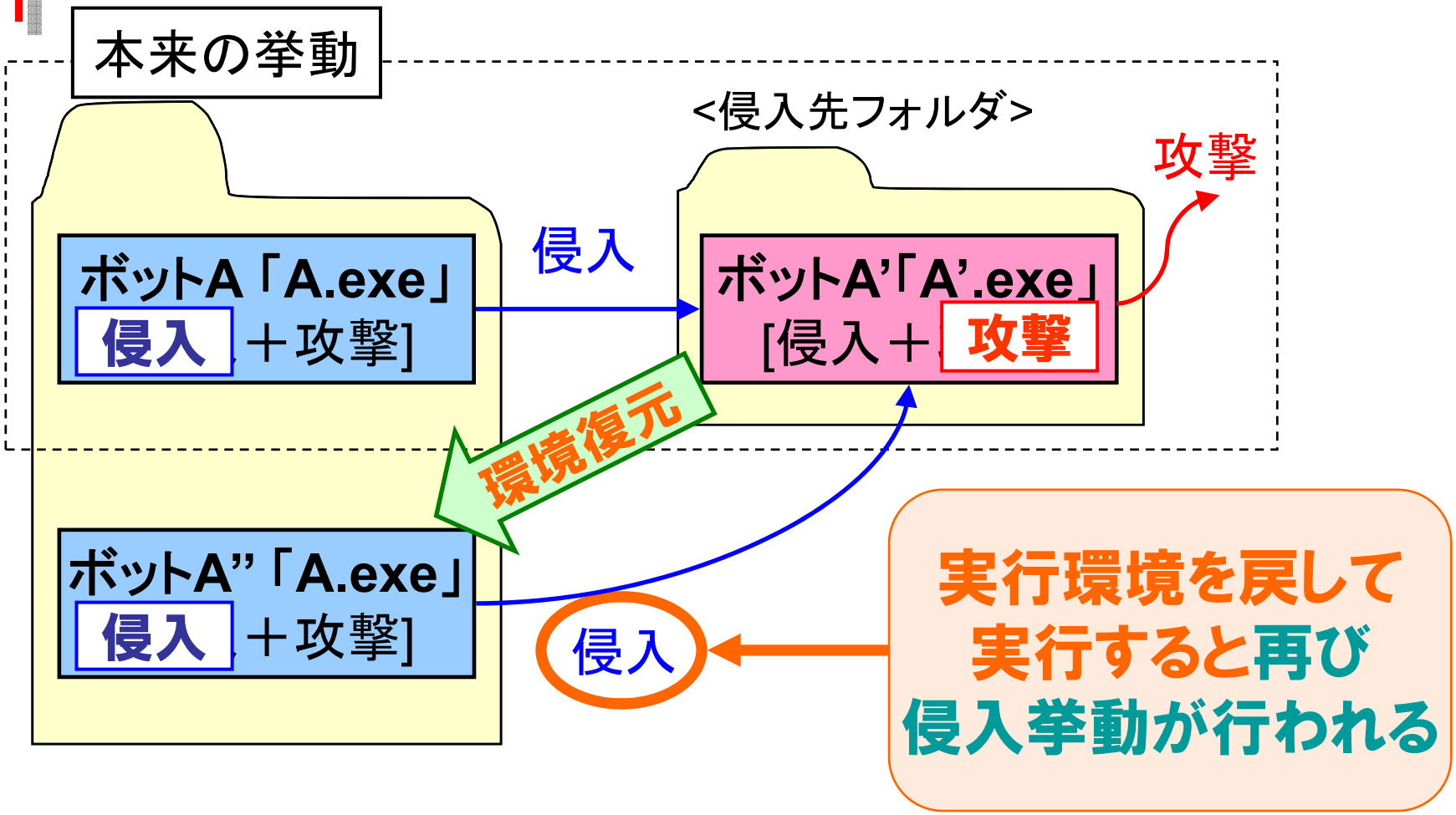
ボットの挙動の特徴

Nishigaki Laboratory



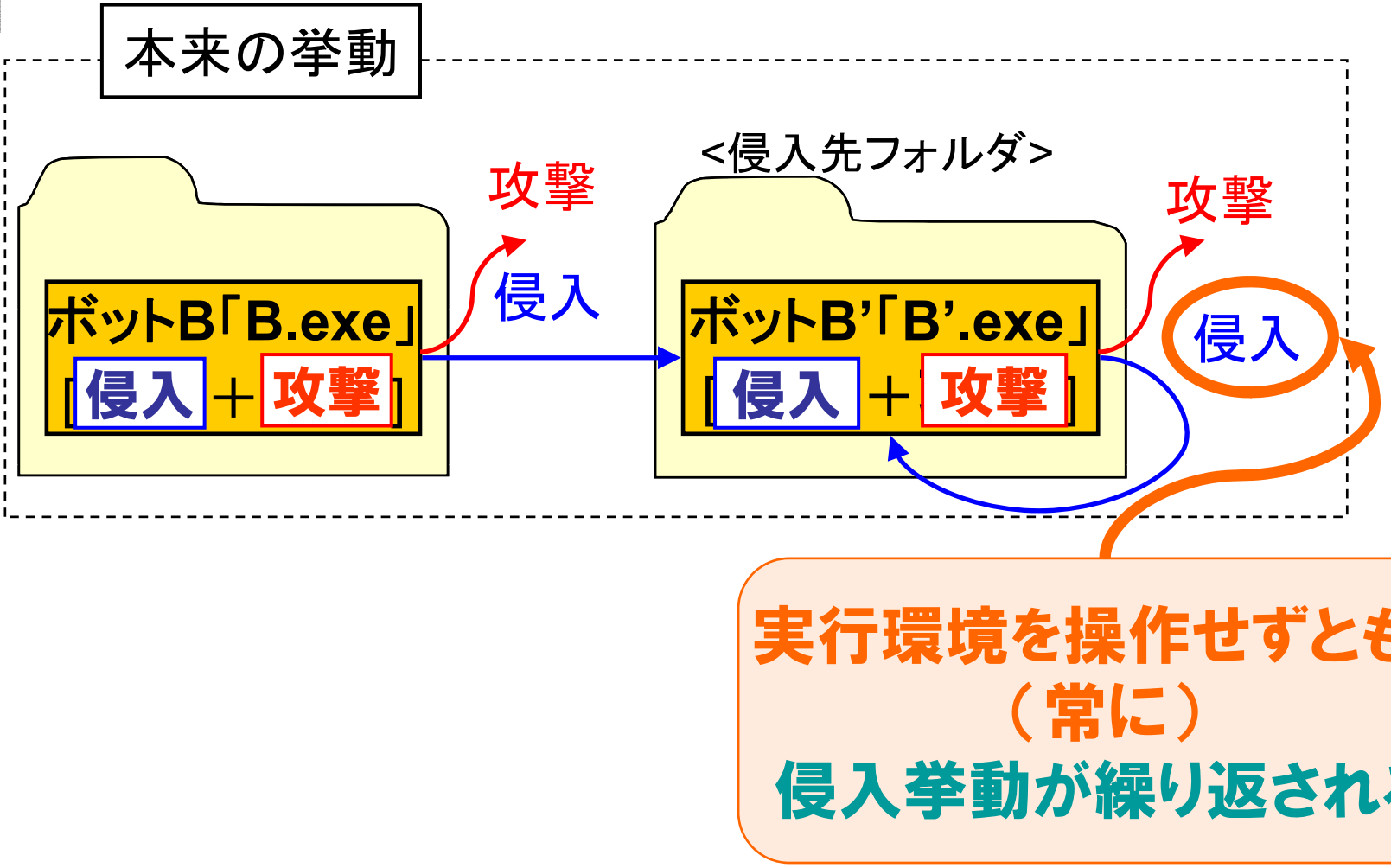
実行環境に影響するボットの挙動

Nish!gaki Laboratory



実行環境に影響しないボットの挙動

Nish!gaki Laboratory



ボットの侵入挙動の反復性

Nish!gaki Laboratory

「侵入/攻撃機能を使い分けるボット」

実行環境を戻して実行すると
再び侵入挙動が行われる

高機能ボット

侵入挙動の反復性

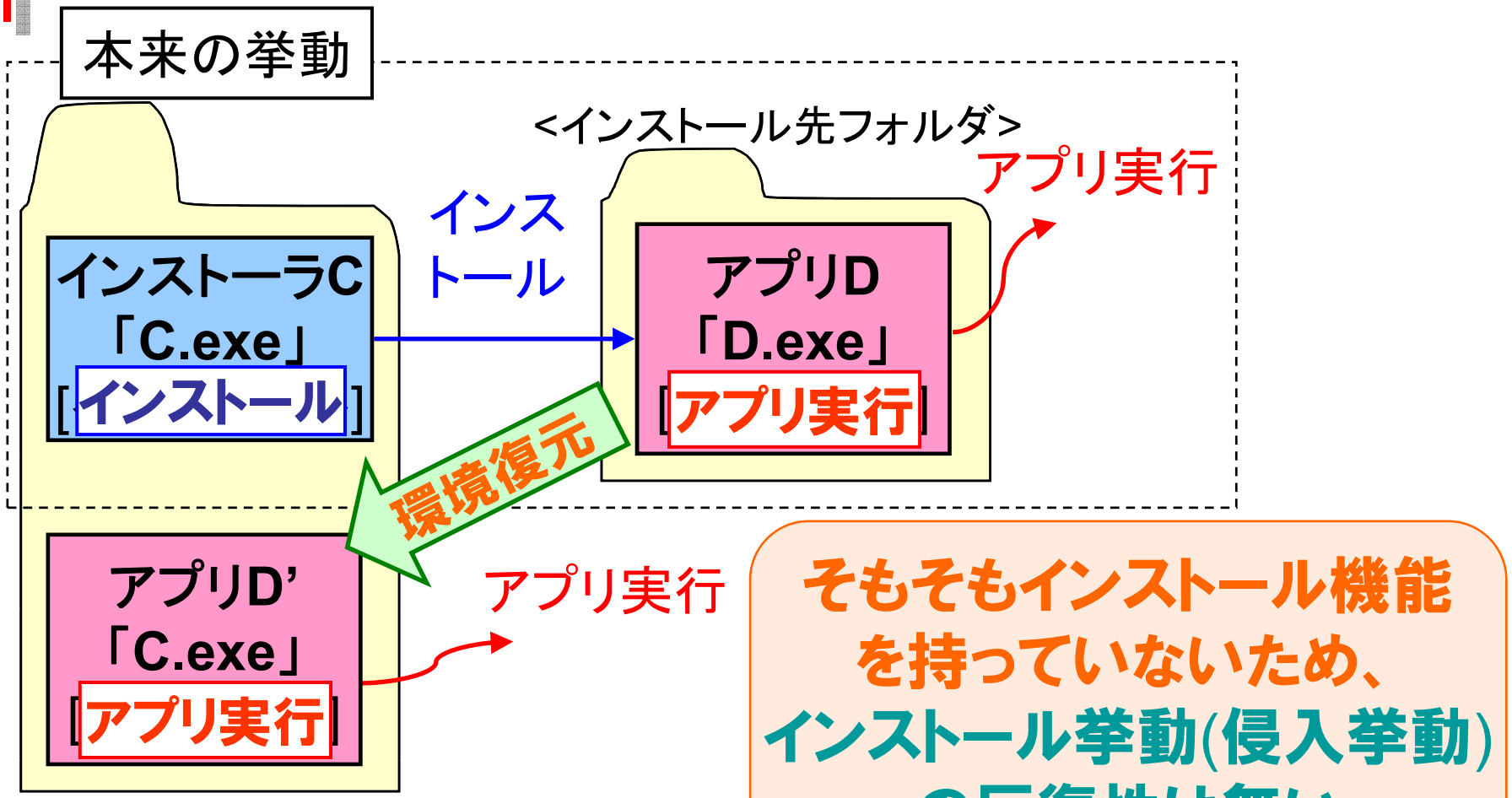
低機能ボット

「常に侵入/攻撃機能を両方使うボット」

実行環境を操作せずとも、
（常に）侵入挙動が繰り返される

実行環境による正規プログラムの挙動

Nish!gaki Laboratory



侵入挙動の反復性を利用したボット検知方式

Nish!gaki Laboratory

低機能ボット

高機能ボット

(1)侵入直後に実行したときの侵入挙動の反復性

Or

(2)実行環境戻して実行したときの侵入挙動の反復性

あり

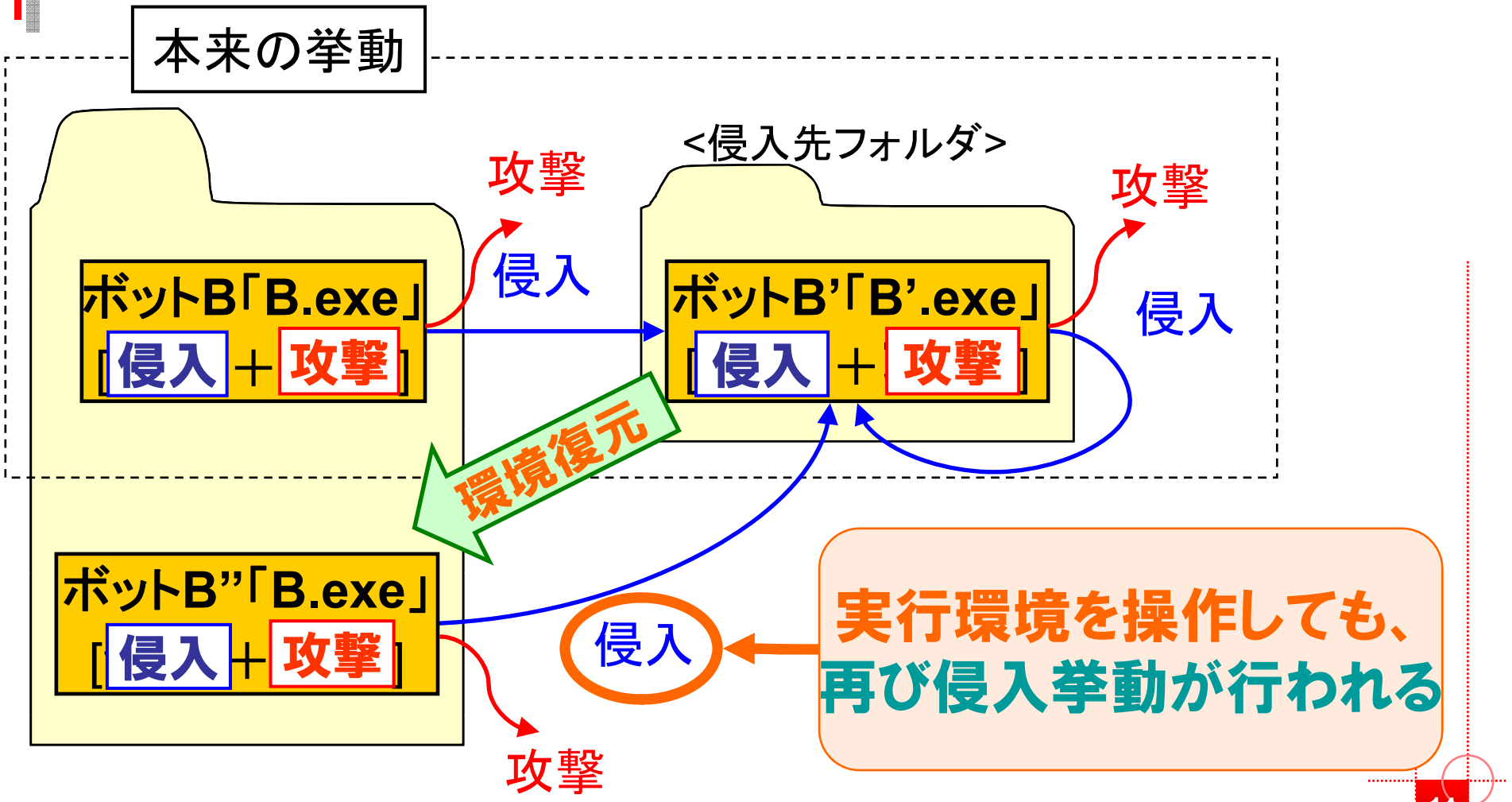
なし

ボット

正規プログラム

実行環境による低機能ボットの挙動

Nish!gaki Laboratory



侵入挙動の反復性を利用したボット検知方式

Nish!gaki Laboratory

低機能ボット

高機能ボット

実行環境戻して実行したときの侵入挙動の反復性

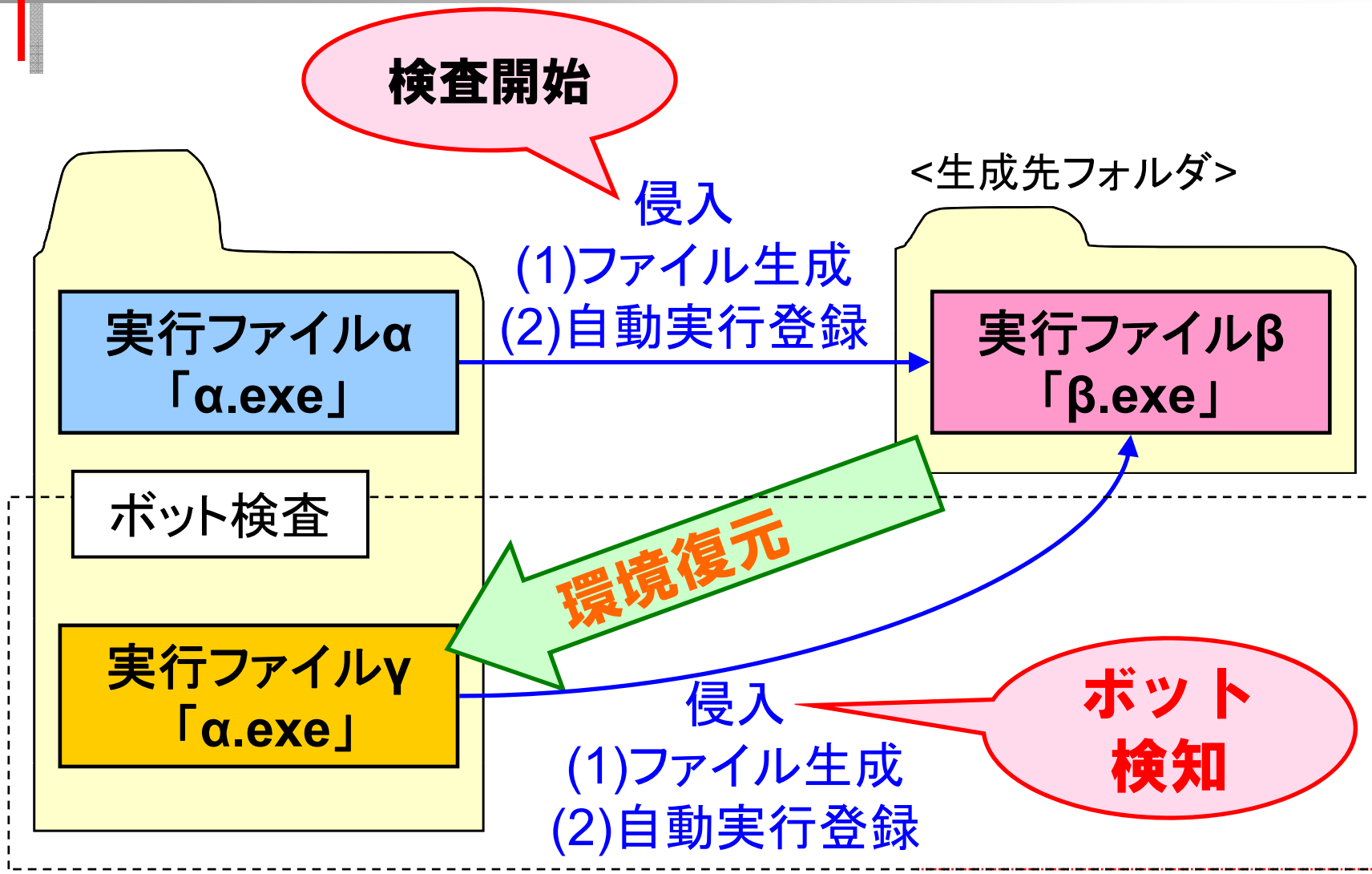
あり

なし

ボット

正規プログラム

提案方式


検査開始
侵入
(1)ファイル生成
(2)自動実行登録
<生成先フォルダ>
実行ファイルα
「α.exe」
実行ファイルβ
「β.exe」
ボット検査
環境復元
実行ファイルγ
「α.exe」
侵入
(1)ファイル生成
(2)自動実行登録
ボット
検知

提案方式の実現方法

Nish!gaki Laboratory

- 「システムコールを監視」
  - 実行ファイルの生成
    - ファイルアクセスの監視
  - 自動実行登録
    - 自動実行の関係のあるレジストリ書込みの監視など

**OSのシステムコール(API)をフックすることにより
リアルタイムで監視可能**

検証実験の概要

Nish!gaki Laboratory

「ProcMonとAutorunsを用いた有効性の評価」

ファイルアクセスを
監視可能なモニタツール

自動実行されるプログラムを
監視可能なモニタツール

(1)「ファイル生成」の観測

(2)「自動実行登録」の観測

「侵入挙動」の観測

検証実験

Nish!gaki Laboratory

「有効性の評価」

- **検知実験：** ボットを用いる
- **誤検知実験：**正規プログラムを用いる

「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

- 研究用データセットCCC DATAset 2009のボット検体10体に対して、提案方式に基づいて、監視ツールを用いた検証実験を行い、提案方式の有効性をしめす。

検知実験結果

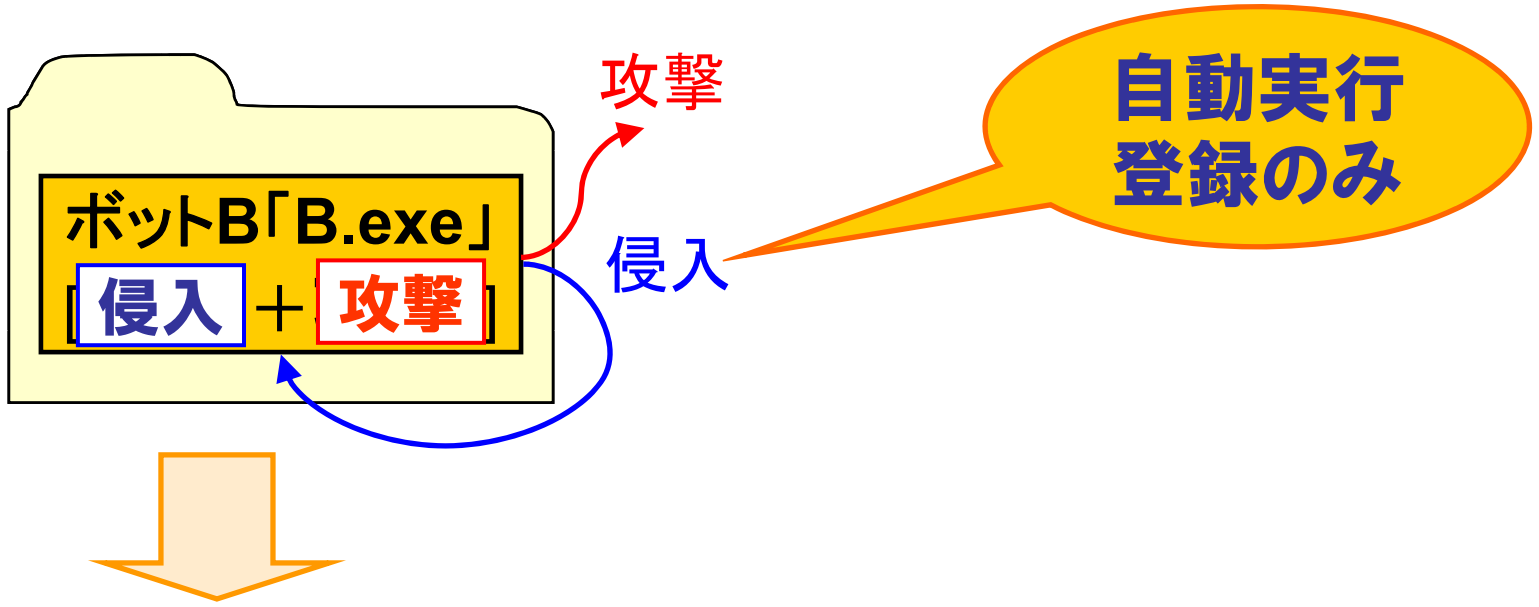
Nish!gaki Laboratory

ボット検体	タイプ			判定
	高機能		低機能	
	場所	名前		
検体A	―	―	○	○(検知)
検体B	×	○	―	○(検知)
検体C	―	―	○	○(検知)
検体D	―	―	○	○(検知)
検体E	○	○	―	○(検知)
検体F	○	○	―	○(検知)
検体G	―	―	○	○(検知)
検体H	―	―	○	△(一部検知)
検体I	×	×	×	×(検知漏れ)
検体J	×	×	×	×(検知漏れ)

検知実験結果　～一部検知～

Nish!gaki Laboratory

- 侵入挙動として「自動実行登録」のみが反復されるタイプの検体



侵入挙動の定義を見直すことによって検知できる可能性がある

検知実験結果

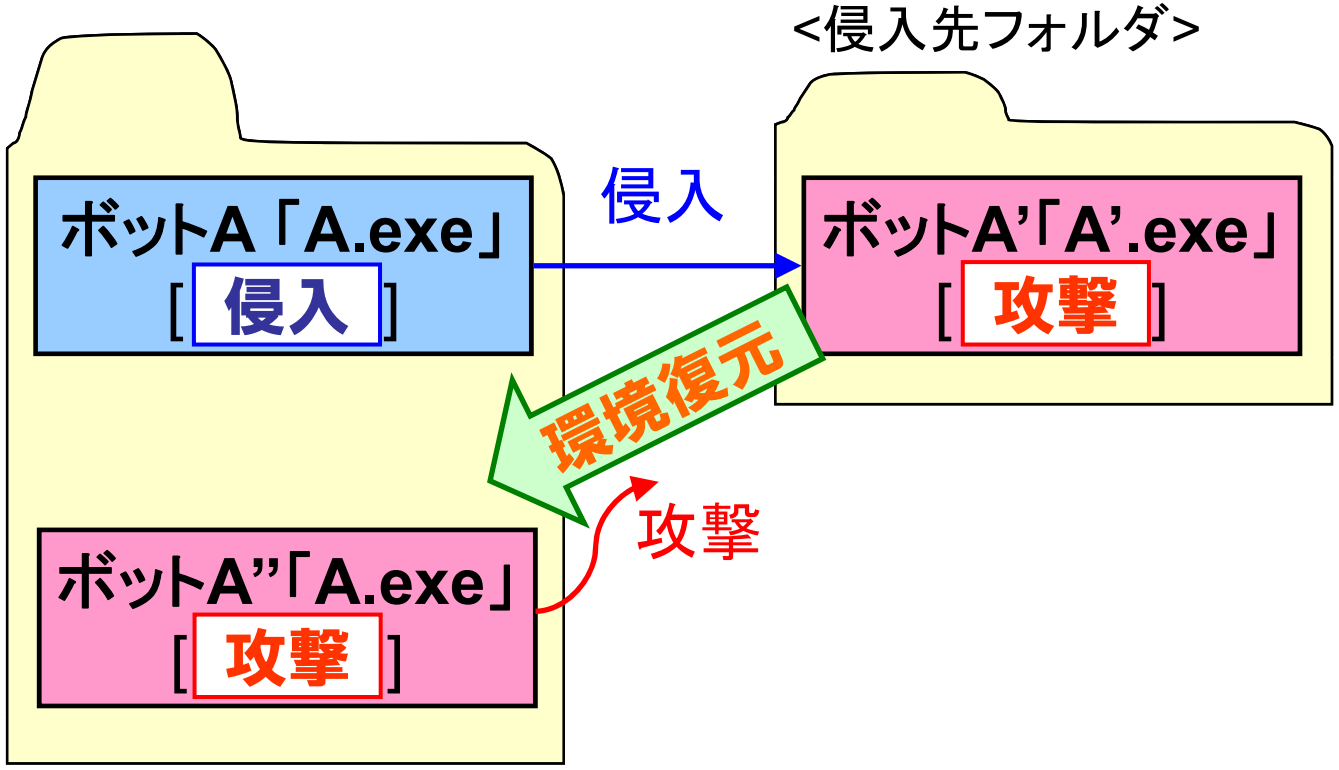
Nish!gaki Laboratory

ボット検体	タイプ			判定
	高機能		低機能	
	場所	名前		
検体A	ー	ー	○	○(検知)
検体B	×	○	ー	○(検知)
検体C	ー	ー	○	○(検知)
検体D	ー	ー	○	○(検知)
検体E	○	○	ー	○(検知)
検体F	○	○	ー	○(検知)
検体G	ー	ー	○	○(検知)
検体H	ー	ー	○	△(一部検知)
検体I	×	×	×	×(検知漏れ)
検体J	×	×	×	×(検知漏れ)

検知実験結果　～検知なし～

Nish!gaki Laboratory

- 生成された実行ファイルが攻撃機能のみを備えるタイプの検体



誤検知実験

Nish!gaki Laboratory

- 一般的に使用されるアプリケーションに対して、本提案方式で誤検知されないかどうかを実験した。

誤検知実験結果

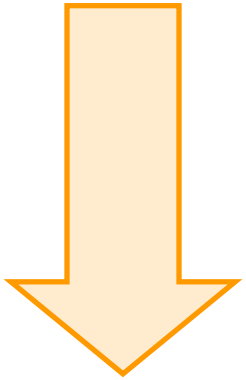
Nish!gaki Laboratory

正規プログラム	判定
MS WORD2003	○（誤検知なし）
InternetExplorer	○（誤検知なし）
Adobe Reader	○（誤検知なし）
Skype	△（一部誤検知）
Rainlendar （スケジュール管理ツール）	△（一部誤検知）
ExtendQuickBar （キーバインドツール）	△（一部誤検知）

誤検知実験結果　～誤検知なし～

Nish!gaki Laboratory

- MS WORD2003等のインストーラは、
  そもそも自動実行登録を行わない



「検査開始条件」
(1)ファイル生成
(2)自動実行登録

検査自体が開始されること無いため、誤検知も無い

誤検知実験結果

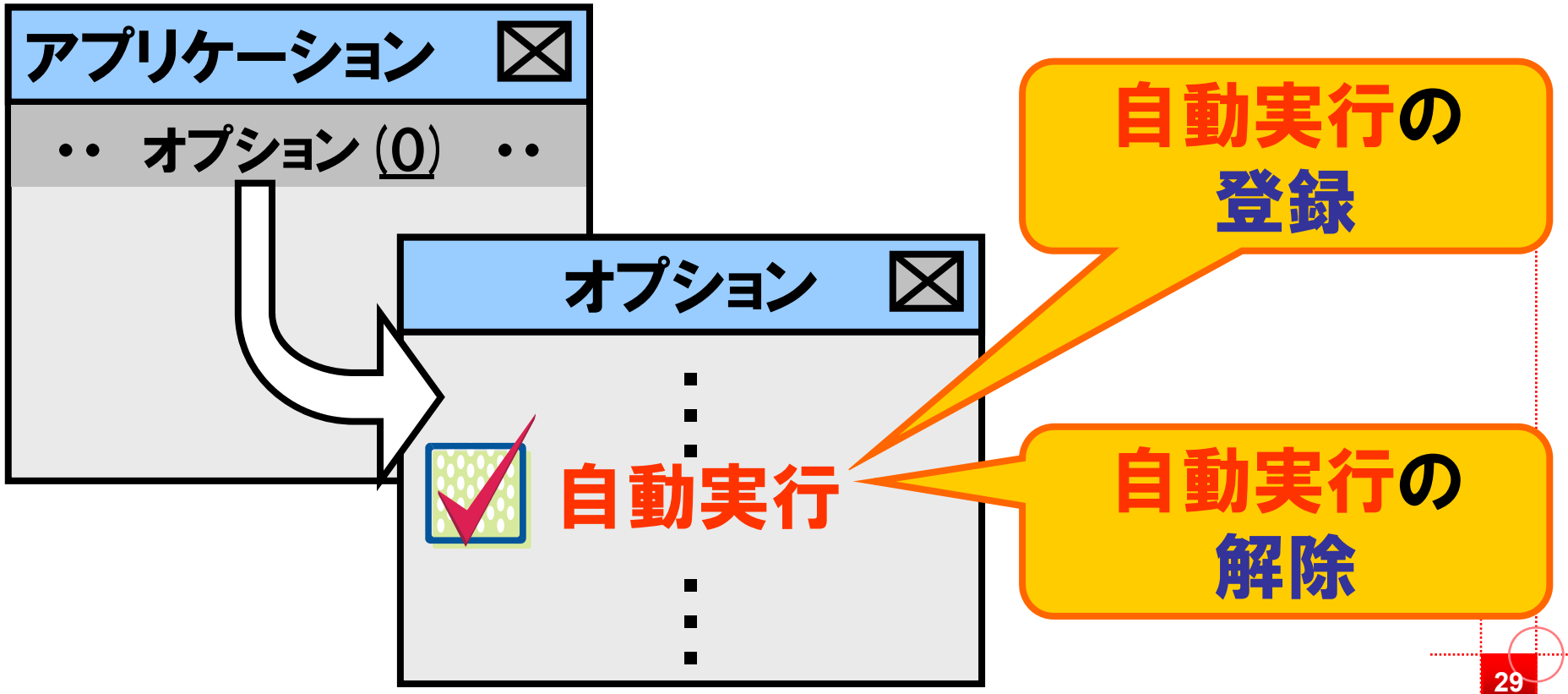
Nish!gaki Laboratory

正規プログラム	判定
MS WORD2003	○（誤検知なし）
InternetExplorer	○（誤検知なし）
Adobe Reader	○（誤検知なし）
Skype	△（一部誤検知）
Rainlendar （スケジュール管理ツール）	△（一部誤検知）
ExtendQuickBar （キーバインドツール）	△（一部誤検知）

誤検知実験結果　～一部誤検知～

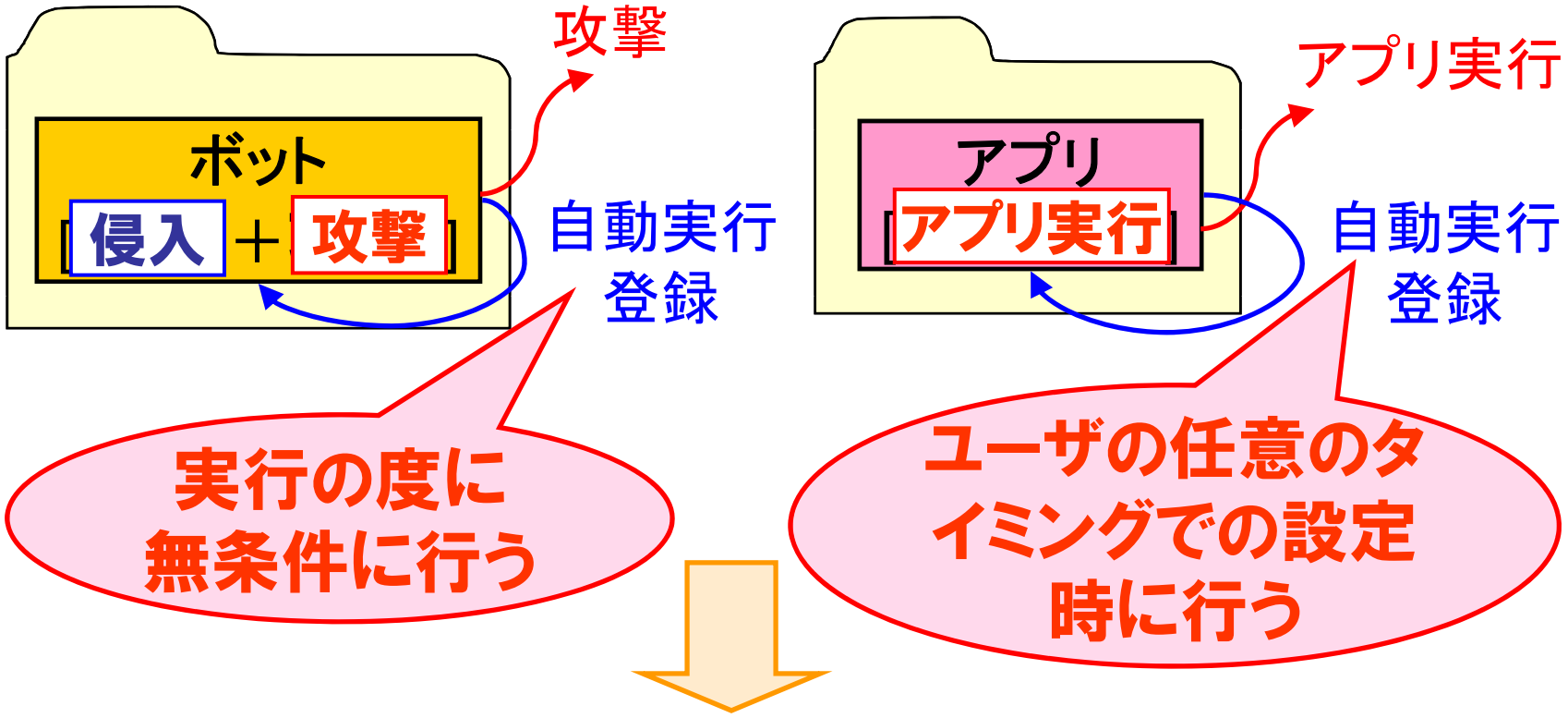
Nish!gaki Laboratory

- Skype等は、ユーザが任意に自動実行を設定できる機能を持っている



一部誤検知について

Nish!gaki Laboratory



条件を整理することによって、ボットは検知/正規プログラムは誤検知なしに改良できる

考察

Nish!gaki Laboratory

- ボットにおいて、本方式では検知出来ない検体があった。

ボットとして潜伏・常駐して機能するためには、何かしらの方法でPC内に侵入する必要はある

考察　～侵入挙動の定義～

Nish!gaki Laboratory

- 検体I:システム系プロセスへの寄生？
- 検体J:DLLを利用した寄生？

(1)ファイル生成
(2)自動実行登録

今回定義した「侵入挙動」では検知出来ない

「ファイル生成」「自動実行登録」以外の侵入挙動を定義することによる、検知方式の発見

まとめ

Nish!gaki Laboratory

- 侵入挙動の反復性を用いて、ボットの検知方式を提案した。
- 検知実験、誤検知実験を行い、その有効性や問題点における対策を示すことが出来た。
- 今後は、侵入挙動の定義をより明確にすることで、検知精度を高めていきたい。