エキスパートによるマルウェア解析レポートと動的解析ログの相関分析

藤野 朗稚 †　　森 達哉 †

† 早稲田大学
169-8555　東京都新宿区大久保 3-4-1
{fujino,mori}@nsl.cs.waseda.ac.jp

あらまし アンチウィルスベンダーは日々大量のマルウェアを解析し，その結果はマルウェア解析レポートとしてデータベースに蓄積されている．一般にマルウェア解析レポートは自然言語で記述されており，実際に使用された API や引数の詳細などは陽に書かれていない．また，解析レポートはマルウェア種別毎に独立しているため，同じ挙動を持つ他の種別を調べることは難しい．本論文では，マルウェア解析のエキスパートによる解析レポートと動的解析ログを対応付けるデータベースの作成を狙いとする．このデータベースを使うことにより，動的解析ログから悪性挙動を自動的に検出可能となることが期待される．実データを用いた解析の結果，異なるマルウェア種別や種別不明のマルウェアからも共通する悪性挙動が検出可能であることが明らかになった．

Correlating Experts' Malware Analysis Reports and Dynamic Malware Analysis Logs

Akinori Fujino†　　Tatsuya Mori†

†School of Fundamental Science and Engineering, Waseda University
3-4-1 Okubo, Shinjuku-ku, Tokyo 169-8555, JAPAN
{fujino,mori}@nsl.cs.waseda.ac.jp

Abstract Anti-virus vendors analyze a huge number of malware samples. Their analysis reports are stored on a database. In general, malware analysis reports are written in natural languages and do not include details of API calls and their arguments. This work aims to develop a database which relates malicious behaviors and dyanmic analysis logs. Such database enables us to automatically detect malicious behaviors from dynamic analysis logs. Using actual dynamic analysis logs of malware samples, we verified that a common set of malicious behaviors can be extracted from multiple, distinct malware samples and even from unclassified malware samples.

1　はじめに

背景: マルウェアの解析方法は静的解析と動的解析の 2 つに大別される．静的解析はマルウェアの実行ファイルを解析するためより深く解析することができるが，難読化のようなマルウェア解析対策技術が施されている場合は対処できない．一方，動的解析はマルウェアを実際の感染環境と近い状況で動作させ，その挙動を解析するため，難読化のようなマルウェア解析対策技術の影響を受けない．そのような理由から，マルウェア解析対策技術に対抗するためには動的解析を用いる必要がある．

Kaspersky Lab の調査では 2014 年末時点において毎日 32 万種類のマルウェアが検出されていることが報告されている [1]．各アンチウイルスベンダー (以下，AV ベンダーと記述する) はそのような大量生産されるマルウェアの解析を日々行っている．そして，一部の AV ベンダーでは解析したマルウェアのレポートをインターネット上で公開している．このレポートはマルウェア解析のエキスパートが作成したものであり，マルウェアの悪意ある挙動に関する詳細な情報が記載されている．具体的にはアクセスファイル，レジストリ，ネットワーク通信に関する情報，他のマルウェアとの関係，レポートの作成者等が記録されている．

解析レポートはマルウェアの種別毎に作成されているため，大量のレポートが各 AV ベンダーのデータベースに蓄積されている．これらのレポートは自然言語で記されており，実際に使用された API や引数はわからない．また，レポートはマルウェア種別毎に独立しており，同じ挙動を持つ他の種別を調べることは難しい．

提案: 上記のような大量の解析レポートは，新たにマルウェア解析を行う上での重要な手がかりとなりうる．例えば，解析レポートから悪性な挙動を定義することができれば，動的解析時に既知の悪性挙動として自動検出が可能になる．また，自然言語で記された悪性挙動の説明と実際の挙動を結びつけることでレポート作成の自動化も可能になる．本論文では，そのような背景にもとづきエキスパートによる解析レポートと動的解析ログの相関分析を行う．

本研究では以下の２つを分析対象にする．

- 約 5,646 のマルウェア検体に対して Cuckoo Sandbox

を適用して収集したログ

- Microsoft 社のマルウェア種別 1,351 種に関する解析レポート

上記の解析レポートから悪性挙動を抽出し，その情報を元に悪性挙動を定義する．次に，動的解析ログから定義した悪性挙動を検出し，解析レポートを作成する．また，これらの処理の全てを自動化する．

貢献: 本研究の主要な貢献は下記のとおりである．

- マルウェア解析レポートから悪性挙動を自動的に抽出する方法を開発した．
- マルウェア動的解析ログから悪性挙動を自動検出できることを示した．

本論文の構成は以下の通りである．はじめに 2 章では本論文の分析対象についてを述べる．つづいて 3 章では提案手法を紹介し， 4 章で分析結果を述べる． 5 章では本論文の制限事項と今後の展望を述べる．最後に 7 章にて本論文のまとめを述べる．

2 分析データ

本章では，分析に利用するデータ詳細と，解析レポートの収集方法を示す．

2.1 FFRI Dataset の概要

本研究では， MWS の研究用データセット [2] の一部として FFRI 社が提供している FFRI Dataset 2013 と FFRI Dataset 2014 を用いる．以下に FFRI Dataset 2013，2014 の主な特徴を記す．

共通する特徴: データは JSON 形式で保存されている動的解析ログである．動的解析に使用したマルウェア検体は PE 形式かつ実行可能で，1 検体あたりの実行時間は 90 秒となっている．

FFRI Dataset 2013: 動的解析には Cuckoo Sandbox が用いられている．マルウェア検体の収集期間は 2012 年 9 月から 2013 年 3 月までとなっており，解析対象となる検体数は 2,643 検体となっている．

FFRI Dataset 2014: 動的解析には Cuckoo Sandbox と FFRI yarai analyzer Professional が用いられている．本論文では FFRI yarai analyzer Professional のログを使用しない．マルウェア検体の収集期間は 2014 年 1 月から 2014 年 4 月までの期間となっており，解析対象となる検体数は 3,000 検体となっている．

以下に Cuckoo Sandbox のログの詳細を述べる．ログには，マルウェア検体のファイル情報，実行時に生成したファイル情報，アクセスしたファイルやレジストリ等の概要，検体中に含まれる文字列情報，通信の概要，API コールログ，VirusTotal [3] の検査結果が記録されている．さらに，API コールログには API の呼び出しが時系列順に記録されており，各 API 呼び出しには関数名や引数，実行の成否，API のカテゴリ，実行日時，返り値等が記載されている．また，FFRI Dataset の各検体の VirusTotal の検査結果には 1 検体あたり最大で 51 種類のアンチウイルスエンジン (以下，AV エンジンと記述する) による検査結果が含まれている．本論文では Cuckoo Sandbox のログ内の API コールログ，VirusTotal の検査結果，通信の概要を用いる．

前述した API のカテゴリごとの呼び出し回数 Top 5 を表 1 に示す．表からわかるように上位 5 つのカテゴリで全 API コールの 96 % を占めている．

表 1: カテゴリごとの API の呼び出し回数 Top 5

カテゴリ	呼び出された回数
filesystem	10349631 (28.21 %)
registry	9124367 (24.87 %)
system	8625661 (23.52 %)
process	3857145 (10.52 %)
misc	3208983 (8.75 %)

本論文では解析レポートの収集のために VirusTotal の検査結果から得られる Microsoft 社のマルウェア種別名を用いる．検査結果から取得できる Microsoft 社のマルウェア種別数の総数は 976 となっている．また，データセット内で種別名が付いている検体が 3,056，付いていない検体が 2,587 であった．本論文では Microsoft 社の マルウェア種別名が記載されているものを既知の検体，種別名が記載されていないものと VirusTotal の検査結果が記録されていないものを未知の検体としている．Microsoft 社のマルウェアの種別名は以下の規則に従って命名されている [4]．

```
Type:Platform/Family.Variant[!Information]
```

Type はマルウェアが端末に与えうる脅威を表しており，Backdoor , Trojan , Worm , PWS などがある．Platform はマルウェアが対象とするプラットフォームやプログラミング言語，ファイルフォーマットを表しており， Win32 , JS , Java などがある．Family は同じ脅威をもつマルウェアのグループを表しており， Vobfus , Zbot などがある．このファミリー名は AV ベンダー間で異なるものを使用している場合がある．Variant はファミリーの亜種名を表しており， AE や AF のような意味を持たない文字列が与えられる．Information は追加情報を表している．

2.2 解析レポート の概要

本研究では，悪性挙動の抽出に Microsoft 社によるマルウェアの解析レポートを用いる．解析レポートは Microsoft 社のウェブサイト上で公開されており，URL は以下に示す生成規則によって取得することができる．

`http://www.microsoft.com/security/portal/threat/encyclopedia/entry.aspx?Name=`マルウェア種別名

Microsoft 社の解析レポートの詳細を以下に示す．解析レポートのファイルフォーマットは HTML であり，AV ベンダーに所属する解析者によって作成される．解析レポートは必ず Summary と Technical information の 2 つで構成されている．Summary には種別に関する概要が示されており，Technical information には種別に関する詳細情報が示されている．Technical information の構造の例を 1 に示す．

Technical information は例外なく id が tab_2 である div タグで表されている．さらに，Technical information は Threat behavior，Symptoms，Prevention

```
<div id="tab_2">
    <h3>Threat behavior</h3>
        ...
        <h5>Installation</h5>
            ...
        <h5>Spreads via...</h5>
            ...
        <h5>Payload</h5>
            ...
    <h3>Symptoms</h3>
        ...
</div>
```

図 1: Technical information の構造の例

で構成されており，それぞれ悪性挙動の詳細情報，悪性挙動によって変更される設定のまとめ，マルウェアへの対策方法が記載されている．Threat behavior と Symptoms はともに悪性挙動について述べられているが，Symptoms は Threat behavior の一部をまとめたものであり，内容が重複している．そのような理由により，本論文では Threat behavior のみを用いることとする．Threat behavior は主に Installation，Spread via，Payload の 3 項目で構成されている．それぞれの概要を以下に示す．

Installation: 感染時に行う挙動が示される．例えば，PC が起動した際に自身が自動で実行されるようにする設定変更，自身のコピーを作成する際のディレクトリやファイル名など．

Spread via: 感染活動の挙動や感染方法に関する情報が示される．例えば，リムーバブルドライブへ自身のコピーを作成する際のファイル名，感染活動に使われるスパムのタイトルや本文など．

Payload: メインの挙動に関する情報が示される．例えば，設定変更される項目，ダウンロードするマルウェア種別名，ネットワーク通信に関する情報など．

この 3 項目は解析レポート内に必ず含まれているわけではなく，項目名が異なるものや 3 項目全てが記述されていないものが存在する．3 項目全てが記載されていないレポートには，1. 他のレポートと同様に悪性挙動について述べられているもの，2. Summary を参照するように促しているもの，3. 同一の挙動をもつ種別のレポートへのリンクが記載されているものの 3 種が存在する．

Threat behavior には項目に関係なく，図 2 で示す形式のいずれかで情報が記載されている．書式 1 では悪性挙動やマルウェア種別についての説明など様々な記述がされる．書式 2 では悪性挙動の説明とその対象となるファイル，ディレクトリ，プロセス，通信先ドメイン，フックされる API，関連するマルウェアなどが記載される．また，挙動そのものについてだけでなく，挙動の目的やレポート内で使用された抽象的な表現の例なども記載される．本論文では <random> <user name> <product name> などの一意に定めることができない表現のことを抽象的な表現としている．書式 3 ではレジストリの設定変更に関する情報が記載される．図 2 は一例であり，エントリ名やエントリ値に関する記述がないもの，1 つのレジストリキーに対して複数のエントリ名，エントリ値の組が記載されるものなど複数の記述方法が存在する．書式 4 ではマルウェア感染に用いられるスパムメールに関する情報が記載される．書式 3 と書式 4 は書式 2 の要素として列挙されることもある．

```
書式 1
    悪性挙動の説明文など

書式 2
    悪性挙動の説明文など:
    ・要素 1
    ...
    ・要素 X

書式 3
    レジストリ操作に関する説明文:
    In subkey:  "対象のレジストリキー"
    Sets value: "追加，変更されるレジストリエントリ名"
    With data:  "追加，変更されるレジストリエントリ値"

書式 4
    スパムに関する説明文:
    Subject: "メールタイトル"
    Attachment: "添付ファイル"
```

図 2: 解析レポートの書式の例

作成者が異なる解析レポートには図 2 で示す書式や HTML タグの使い方が異なるものが存在する．また，人為的なミスによってスペルミスが含まれるレポートも存在する．

2.3 解析レポートの収集

本節では，悪性挙動の抽出に用いる Microsoft 社の解析レポートの収集方法を示す．解析レポートの中には 2.2 節でも述べたように挙動に関する記述がなく，同一の動作をする種別のレポートへのリンクが記載されているものが存在する．そのようなレポートを持つマルウェア種別の挙動を抽出するためには，リンク先のレポートを用いる必要がある．また，亜種のレポート内には元となるファミリーのレポートへのリンクが含まれていることが多い．元となるファミリーのレポート内にはファミリーに共通する挙動が記載されており有用である．上記の理由により，本研究ではリンク先の解析レポートも収集する．本研究における解析レポートの収集方法を以下に示す．

1. FFRI Dataset からマルウェア種別名を抽出する
2. 抽出した種別名から解析レポートの URL リストを生成する
3. 生成した URL リストを用いて解析レポートを収集する
4. 収集した解析レポートから他の解析レポートへのリンクを抽出する
5. 抽出したリンクの中から解析レポートを収集済みのものを削除する
6. 3 ～ 5 を繰り返す

上記の方法によって得られた解析レポートの数は 1,347 である．その内 FFRI Dataset から抽出した種別のものは 976，リンクを用いて収集した種別のものは 371 となっている．それらの種別に含まれるファミリーの数は 492 であった．

3 提案手法

本章では，提案手法の概要， 2.3 節で収集した解析レポートに対する前処理，レポートからの悪性挙動の抽出方法，動的解析ログから悪性挙動を検出する方法，各マルウェア検体のレポート作成方法を示す．

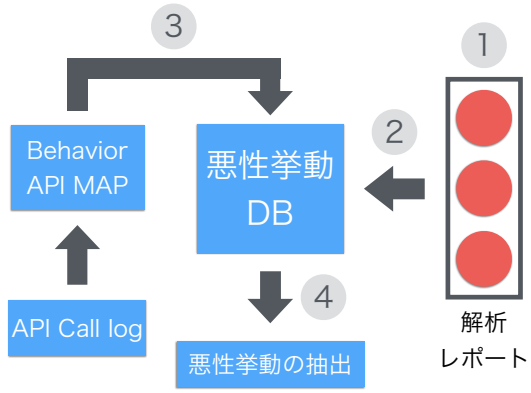

図 3: 提案手法の概要図

3.1 提案手法の概要

提案手法の概要を図 3 に示す．図 3 中の構成要素の詳細を以下に示す．解析レポートは 2.3 節で収集したものであり，図 3 中の 1 つの点が 1 つのレポートを表している．悪性挙動 DB は解析レポートから抽出した悪性挙動をカテゴリに分けて保存しているデータベースである．カテゴリには，copy_file，create_file，create_directory，set_value などがある．Behavior API MAP は悪性挙動 DB のカテゴリと API を結びつけるためのものである．例えば，copy_file と結びつけられている API は CopyFile [5]，CopyFile2 [6] などがある．API Call log は動的解析ログの中の API の呼び出しが記載されている部分である．図 3 中の 1 ～ 4 についてを以下に示す．

1. 解析レポートに対して前処理を行う
2. 解析レポートから悪性挙動と挙動の説明文を抽出する
3. API コールに対応する悪性挙動を Behavior API MAP を介して悪性挙動 DB に問い合わせる
4. API コールに対応する悪性挙動が存在する場合，その悪性挙動と挙動の説明文を結果に追加する

それぞれの詳細は後述の各節で述べる．

3.2 解析レポートに対する前処理

本論文では，作成者ごとの解析レポートの差異を除去するために前処理を行う．前処理の手順を以下に示す．

1. 解析レポートから Threat behavior を抽出する
2. 抽出した Threat behavior を HTML タグを含まないテキストに変換する
3. 同じ意味を持つ複数の単語を一つの単語に統一する
4. 改行文字で分割したリストに変換する

2.2 節で述べたように，解析レポートは作成者によって HTML タグの使い方が異なる場合がある．その違いを吸収するために HTML 内の全ての文字列を抜き出し，HTML タグを含まないテキストに変換する．このとき，特定の HTML タグでは文字列を抜き出すだけではなく，追加の処理を行う．その一覧を表 3 に示す．

表 2: 追加処理の一覧

HTML タグ	追加処理の内容
li	文字列の前後に改行文字を挿入
br	改行文字を挿入

一般的に HTML タグの li はリスト構造を表すときに使用される．このタグ内の文字列は明示的に改行を記入することなく改行されるため，タグを削除し，文字列のみを抽出する場合には，リスト構造を維持するために前後に改行を挿入する．また，HTML タグの br はプレーンテキストの改行文字に相当するものであり，タグ内に文字列を持たない．テキストの構造を維持するため，br タグを改行文字に置換する．解析レポートの作成者によっては同じ意味をもつものに対して異なる表記を用いる場合がある．その表記の抜粋を表 3 に示す．スペースの関係上示していないが，表 3 は一部分であり，それ以外の表記も存在する．本論文では，これらの表記のそれぞれを一つの表記に統一することで，より多くの悪性挙動の抽出を可能にしている．

表 3: 同じ意味を持つ表記の抜粋

意味	表記
Registry key	In subkey:, In subkeys:, Under key:, Under keys:, To subkey:, To subkeys:,
Registry value	Sets value:, Sets values:, Adds value:, Add value:, Modifies value:, Value:
Registry data	With data:, To data:, Data:, From data:
Random string	<random string>, <random phrase>, <random filename>
Random num	<random number>, <random hexadecimal number>, <random nine digit number>
User name	<User name>, <user name>, <user>, <username>
Dot	”.”, <dot>

3.3 悪性挙動の抽出方法

本節では Microsoft 社の解析レポートから悪性挙動を抽出する方法を示す．2.2 節で述べたように解析レポートは図 2 で示した書式 1 ～ 4 のいずれかを用いて述べられている．2.2 節で述べた書式の説明の内，悪性挙動についてまとめたものを 4 に示す．

表 4: レポートの書式と記述内容

書式	記述内容
書式 1	悪性挙動全般
書式 2	ファイル操作，通信先ドメイン， プロセス，フックされる API， ダウンロードする種別
書式 3	レジストリ操作
書式 4	スパムのタイトル，添付ファイル

このうち，書式 1 は文の構成が多様であり，全てのレポートから同じように情報を抽出するのは難しいため，本論文では使用しない．また，書式 4 で表されるスパムの情報はマルウェアの感染経路の一つであり，マルウェアの挙動ではないため，本論文では使用しない．そのため，本論文で使用する書式は書式 2 と書式 3 の 2 つとなる．また，本論文では書式 2 の内，ファイル操作，通信先ドメインに関する情報のみを用いる．つ

```
L1 の例
[
    ...
    'In subkey: key1','Sets value: val1',
    'With data: data1','Sets value: val2',
    'With data: data2','In subkey: key2',
    'or subkey: key3','Sets value: val3',
    'With data: data3',
    ...
]

抽出できる挙動
    {"subkey": [key1],"value": [val1,val2],
        "data": [data1,data2]}
    {"subkey": [key2,key3],"value": [val3],
        "data": [data3]}

悪性挙動 DB に保存する挙動の組み合わせ
    [key1,val1,data1]
    [key1,val2,data2]
    [key2,val3,data3]
    [key3,val3,data3]
```

図 4: 3.2 節で得たリストの例と抽出結果

まり，本論文で抽出する悪性挙動はファイル操作，レジストリ操作，通信先ドメインの 3 種である．こうした理由を以下に示す．

- 表 1 で示したようにファイル操作，レジストリ操作で使用される API は全 API コール中の 50 % 以上を占めており，情報量が多かったため
- 他のマルウェアや設定ファイルのダウンロードなど外部との通信はマルウェアの挙動として重要なものだと考えたため
- 上記の 3 種は挙動の抽出と検出が比較的容易であるため

本論文では悪性挙動 DB に保存する挙動のカテゴリを copy_file，create_file，create_directory，set_value，query_value，domain の 6 つとしている．そして，copy_file，create_file，create_directory をファイル操作系悪性挙動，set_value，query_value をレジストリ操作系悪性挙動，domain を通信系悪性挙動とする．それぞれの悪性挙動を 3.2 節で得たリスト (以下，L1) から抽出する方法を以下に示す．

3.3.1 ファイル操作系悪性挙動の抽出

ファイル操作系悪性挙動は書式 2 によって記載されており，悪性挙動の説明文によって要素のカテゴリが copy_file，create_file，create_directory のどれかを判別している．カテゴリの判別方法を以下に示す．

- "copy" と "file" が含まれる ⇒ copy_file
- "create" と "file" が含まれる ⇒ create_file
- "create" と "folder" か "directory" が含まれる ⇒ create_directory

"copy" に関しては 活用形が含まれるかも調べる．また，"directory" に関しては複数形が含まれるかも調べる．これは "copies" のような 3 人称単数現在形や "directories" が使われている場合にも抽出できるようにするためである．この方法でカテゴリを判別した後，リストの各要素を悪性挙動 DB の該当するカテゴリに保存する．

3.3.2 レジストリ操作系悪性挙動の抽出

レジストリ操作系悪性挙動は書式 3 によって記載されており，基本的に レジストリキー (以下，S)， レジストリエントリ名 (以下，V)，レジストリエントリ値 (以下，D) で構成されている．内での SVD の出現順序について以下に示す．

- SVD の出現順序は固定ではない
- S は必ず最初か最後に示される
- ひとつの S に対して複数の V と D の組が存在する場合もある
- "or subkey:" を使ってレジストリキーを併記する場合もあり，その場合の順序は SSVD になる
- SVD の間には挙動の説明文などの他の要素は入らない

そのような法則から SVD の順序を正規表現で表すと以下のようになる．

```
SS?((VD)+|(DV)+)?  ((VD)+|(DV)+)?S S(V|D)? (V|D)?S
```

上記の正規表現を満たす SVD の組みを 1 つのレジストリ操作系悪性挙動として悪性挙動 DB に保存する．S と VD の組み合わせが複数存在する場合は，その全てを 悪性挙動 DB に保存する．

L1 の例とその抽出結果を図 4 に示す．

3.3.3 通信系悪性挙動の抽出

通信系悪性挙動はファイル操作系悪性挙動と同様に書式 2 で記載されている．書式 2 の各要素のうち，末尾が文献 [7] で示されるトップレベルドメインのいずれかになっているものを使用する．

3.4 悪性挙動の検出

本節では動的解析ログから悪性挙動を検出する方法を示す．動的解析ログには 2.1 節で述べたように，既知検体のものと未知検体ものがあるが，ここでは全てのログに対して検出を行うこととする．検出する際にはファイル操作系悪性挙動とレジストリ操作系悪性挙動は API コールログを，通信系悪性挙動は動的解析ログ中の通信情報が記載されている項目を調べる．

通信情報からの悪性挙動の検出では悪性挙動 DB に保存されているドメインが動的解析ログに含まれているかを調べる．本論文では通信する目的やドメインが悪性なものであるかは調べない．

レジストリ操作系悪性挙動では，実際に特定のレジストリキー，エントリ名に対してエントリ値の変更を行った場合だけでなく，現在のエントリ値の確認を行った場合も検出している．これは既に設定を変更済みの環境ではエントリ値の変更を行う必要がなく，確認だけが行われると考えたためである．

ファイル操作系とレジストリ操作系の検出には Behavior API MAP を用いる．Behavior API MAP で結びつけている カテゴリと API を表 5 に示す．悪性挙動の検出時は，呼び出された API を使った悪性挙動がないかを Behavior API MAP を介して悪性挙動 DB に問い合わせる．問い合わせ時に必要な情報がカテゴリごとに存在する．その必要な情報の一覧を表 6 に示す．表 6 に示されている情報は API の引数として与えられる場合がほとんどである．例外として，set_value と query_value で必要となるレジストリキーだけは引数で与えられない．レジストリ操作系の API は以下の手順で処理を行う．

1. open_key (create_key) でレジストリキーを開く

表 5: Behavior API MAP

カテゴリ	API
copy_file	CopyFile,CopyFile2, CopyFileEx
create_file	NtCreateFile,CreateFile, CreateFileA,CreateFileW, CreateFile2,
create_directory	CreateDirectory, CreateDirectoryEx,
open_key	RegOpenKeyEx, NtOpenKey, NtOpenKeyEx
create_key	RegCreateKeyEx, NtCreateKey
set_value	RegSetValueEx, NtSetValueKey
query_value	RegQueryValueEx, NtQueryValueKey
close_key	RegCloseKey, NtClose

2. set_value (query_value) は引数で与えられた 1 の API で得たハンドル，エントリ名，エントリ値を用いて処理を行う
3. close_key でレジストリキーを閉じる

レジストリキーが明示的に引数として与えられるのは open_key と create_key だけである．それ以外のレジストリ操作系の API は open_key と create_key の実行時に得られたハンドルを介してレジストリキーへの処理を行う．また， open_key や create_key が入れ子で使用されることがある．この場合，それぞれが開いたレジストリキーを連結したものがレジストリキーのフルパスとなる．このレジストリキーのフルパスとハンドルの組を保存しておくことで set_value と query_value で必要となるレジストリキーを特定する．

表 6: 悪性挙動の検索時に必要な情報

カテゴリ	必要となる情報
copy_file	コピー先のファイル名
create_file	作成するファイル名
create_directory	作成するディレクトリ名
set_value	キー，エントリ名，エントリ値
query_value	キー，エントリ名

4 結果

本章では動的解析ログから検出した悪性挙動についてと検出できた悪性挙動と正解となる解析レポートに含まれている挙動の比較，未知のマルウェア検体に対する検出結果を示す．

4.1 検出した悪性挙動について

検出できた悪性挙動の例を図 9，図 10 に示す．図では空の項目を省略している．例に示したとおり，検出した悪性挙動は DB のカテゴリごとに記録されている．各悪性挙動については，検出時の API の全引数を記録している．さらに，ファイル操作系悪性挙動においては挙動の説明文も記録している．この説明文は解析レポートのものを流用している．図 9，図 10 の説明を以下に示す．

copy_file: ”ExistingFileName”，”NewFileName”，”Discription” がコピー元のファイル，コピー先のファイル，挙動の説明文となっており，図 9 では update.exe という自身のコピーを複製していることがわかる．

create_file: ”FileName” が作成するファイル名となっており，図 9 では copy_file のものと同じファイルに関して記載されている．

create_directory: ”DirectoryName” が作成するディレクトリ名となっており，図 9 では rotinom という隠しディレクトリを作成していることがわかる．

set_value: ”value”，”data” がエントリ名，エントリ値となっており，図 9 ではエントリ名 ”Hidden”，”HideFileExt”，”ShowSuperHidden” にエントリ値 2，1，0 を保存することにより隠しファイルを非表示にしている．

query_value: マルウェア検体が参照したレジストリ情報の中で，悪性挙動 DB に保存されているものレジストリキー，エントリ名のものが記録されている．

domain: マルウェアが通信を試みたドメインの中で，悪性挙動 DB に保存されているものが記録されている．

4.2 悪性挙動の検出結果について

データセットの各検体から検出できた悪性挙動数の CDF を図 5 に示す．この図の縦軸は検体の割合を，横軸は 1 検体に含まれる悪性挙動数，赤い破線は平均値を表している．対象となるのは 5,646 検体で，検出された悪性挙動数の最小値は 0，最大値は 29，平均値は 1.61 であった．また，悪性挙動が検出できていない検体の正解となる解析レポート (以下，正解レポート) の 1 つからの抜粋を図 11 に示す．

図 5 から，約 6 割の検体から悪性挙動を検出できていないこと，悪性挙動数が 10 以上の検体はほとんど存在しないことがわかる．悪性挙動が検出出来ない理由として考えられるものを以下に示す．

要因 1: 図 11 からわかるように，正解レポートでは <folder>，<random>，<GUID> などの抽象的な表現を用いることがある．本論文では上記の表現のものも 1 つ悪性挙動として抽出しているが，検出方法は実装していない．そのため，図 11 ような正解レポートをもつ検体からは悪性挙動を検出することが出来ない．

要因 2: 本論文では抽出する悪性挙動をファイル操作，レジストリ操作，通信に絞っている．また，この 3 種に関しても全てを抜き出せているわけではない．なので，ファイル操作，レジストリ操作，通信以外の悪性挙動や，抽出に失敗している悪性挙動に関しては検出することが出来ない．

正解レポートから悪性挙動を 1 つ以上抽出できる種別かつ悪性挙動を 1 つ以上もつ検体に対してカバレッジを計算し，CDF にしたものを図 6 に，正解レポートに含まれない悪性挙動数の CDF を図 7 に示す．本論文では，正解レポートに記載されている悪性挙動の再現率をカバレッジとしている．計算方法は，検出された悪性挙動の内，正解レポートに含まれるものの数を正解レポートから抽出できる悪性挙動数で割ったものとしている．図 6 では平均値が 10 %，最大値が 100 % となっている．平均値が小さくなっている要因として考えられるものを以下に示す．

要因: 使用した正解レポートから抽出できる悪性挙動数

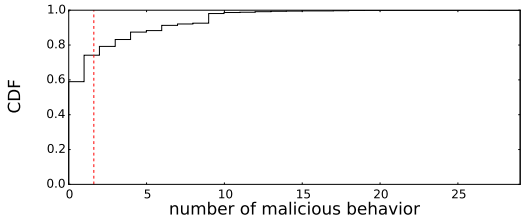

図 5: 1 検体に含まれる悪性挙動数の CDF

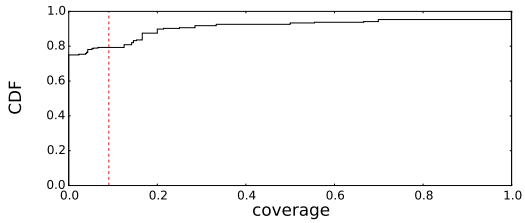

図 6: 正解レポートに対するカバレッジの CDF

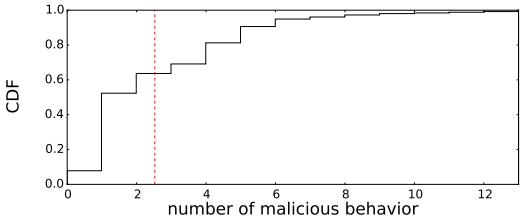

図 7: 解析レポートに含まれない悪性挙動の CDF

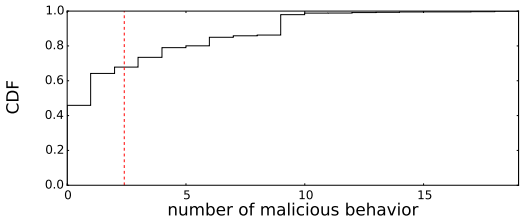

図 8: 種別名がない検体の悪性挙動数の CDF

の中間値は 5 である．そして，抽出する悪性挙動には抽象的な表現を持つものも含まれる場合がある．この場合，検出可能な悪性挙動の数が少なくなってしまい，カバレッジが小さくなる．

図 7 から，悪性挙動を 1 つ以上もつ検体の約 9 割が正解レポートに含まれていない挙動をもつことがわかる．これは，解析レポートに記載されている悪性挙動がそのマルウェアの挙動の全てではなく，レポート作成者が省略した，もしくは解析者が発見できなかった悪性挙動がマルウェアに含まれている可能性があるということを示している．また，異なる種別であっても共通する挙動を行うものが存在するということがわかる．

4.3 未知のマルウェア検体について

図 5 の中から，未知の検体の結果のみを抽出したものを図 8 に示す．この図から，未知の検体の約 6 割から悪性挙動を抽出できていることがわかる．すなわち，本論文で使用したマルウェア検体に関して言えば，悪性挙動を検出できた検体数は未知の検体の方が既存の検体よりも多かったということである．これは以下の 2 つのことを意味している．

意味 1: 未知のマルウェア検体であっても既存の悪性挙動をもっている場合があること

意味 2: 本論文の提案手法はマルウェアの既知，未知に関わらず，適用することができること

5 議論

本章では本論文に関する制限事項と今後の展望について述べる．

5.1 制限事項

本論文の手法は悪性挙動の検出結果は動的解析ログに含まれる API コールの情報量に左右されるが，動的解析ログには 実行時間が 90 秒という制限があり，そこから得られる情報には限りがある．

本論文では悪性挙動の抽出をファイル操作，レジストリ操作，通信先ドメインの 3 種に絞っている．そのため，その他の悪性挙動を検出することができない．マルウェア種別によってはこの制限の影響を強く受け，挙動が一つも検出できないというが起こりうる．また，ファイル操作系悪性挙動のカテゴリの判別方法が単純であるため，他の表現が使われている場合に抽出することができない．

本論文では解析レポート中のスペルミスを一部を除き修正せずに悪性挙動 DB に保存している．そのため，DB に保存済みの悪性挙動が出現していた場合でもスペルミスが原因で見逃している可能性がある．

本論文では抽出した悪性挙動のうち， 4.3 節で述べた抽象的な表現をもつ挙動の検出はできない．そのような挙動は解析レポートで具体的な例が与えられた場合に限り検出することができている．

通信系悪性挙動については DB に保存したドメインが悪性かどうかの判定を行っていないため，悪性ではないドメインが含まれている可能性がある．

5.2 今後の展望

本研究では抽出可能な悪性挙動の種別の拡大，悪性挙動の抽出精度向上，正解レポートに対するカバレッジの改善を今後の課題とする．この課題の解決のために今後は以下の実現を目指す．

- 今回対象外とした書式 1 (自由記述) の利用
- 今回対象外としたプロセス情報等の利用
- レポート中のスペルミスの修正方法の確立
- 悪性挙動のカテゴリ判別方法の改善
- 悪性ではないドメインの除外
- 抽象的な表現を含む悪性挙動の検出方法の確立

6 関連研究

本章では関連研究について述べる．マルウェア解析レポートを扱った研究は筆者の知る限りない．ここでは動的解析ログ中を分析した研究について述べる．

Ahmed [9] らは API コールログの時系列情報と API の入出力情報を組み合わせることでマルウェアが検出できることを示した．Alazab [10] らは IDA Pro ディスアセンブラによって得られた API コールに n-gram を適用するで正常検体とマルウェア検体を分類できることを示した．Ravi [11] らも同様に API コールに n-gram を適用することで正常検体とマルウェア検体を分類している．

```
"File Info": {
    "copy\_file": [{
        "Discription": "Upon execution, this file
            creates a copy of itself as the
            following file:",
        "ExistingFileName": "c:\users\cuckoo
            \appdata\local\temp\update.exe",
        "NewFileName": "c:\users\cuckoo\appdata
            \local\start\update.exe"}],
    "create\_directory": [{
        "DirectoryName": "c:\users\cuckoo\appdata
            \local\s-1-5-31-1286970278978-57136694
            91-166975984-320\rotinom",
        "Discription": "this file also creates the
            following hidden folders:"}],
    "create\_file": [{
        "CreateDisposition": "1",
        "DesiredAccess": "0x80100080",
        "Discription": "Upon execution, this file|
            creates a copy of itself as the
            following file:",
        "FileName": "c:\users\cuckoo\appdata\local
            \start\update.exe",
        "ShareAccess": "5"}],}
"Registry Info": {
    "RegQueryValue": {
        "software\microsoft\windows\currentversion
            \explorer\advanced": [
            "Hidden", "HideFileExt",
            "HideIcons", "ShowSuperHidden"],
        "software\microsoft\windows\currentversion
            \explorer\cabinetstate": ["FullPath"]},
    "RegSetValue": {
        "software\microsoft\windows\currentversion
            \explorer\advanced": [
            {"data": "2","value": "Hidden"},
            {"data": "1","value": "HideFileExt"},
            {"data": "0","value": "ShowSuperHidden"}]}}
```


図 9: 検出した悪性挙動の例 1


```
{"Registry Info": {
     "RegQueryValue": {
        "software\microsoft\windows nt\currentversion\
            svchost": ["netsvcs"]}},
 "domain": [
     "conf.f.360.cn", "qup.f.360.cn", "qup.qh-lb.com",
     "sdup.360.cn", "sdup.qh-lb.com", "sdupm.360.cn",
     "qd.code.360.cn", "qd.code.qihoo.com",
     "stat.360safe.com", "stat-s.360safe.com",
     "update.360safe.com", "update-s.360safe.com",
     "tr.p.360.cn", "updateh.360safe.com", "w.360.cn"
     "stat.sd.360.cn", "sdl.360safe.com",
     "dl.360safe.com", "dl.qh-lb.com", "www.360.cn",
     "www.360safe.com", "softm.update.360safe.com",
     "softm-s.update.360safe.com", "bo.duba.net",
     "antispy.db.kingsoft.com", "f-sq.beike.cn"]}
```


図 10: 検出した悪性挙動の例 2

上記の研究は全て API コールの時系列情報を利用した特徴量を作成し，教師あり学習を適用することでマルウェアの分類を行っている．それに対し，文献 [12]，[13] では教師なし学習を用いた研究を行っている．Bayer [12] らは各マルウェア検体から特徴量を抽出し，クラスタリングアルゴリズムを適用することで類似する検体が同じクラスタに分類できることを示した．Li [13] らはマルウェアのクラスタ分析に使用する正解データが分類結果の精度に大きな影響することを示した．この研究では文献 [12] を含む先行研究の各手法を用いて評価を行っている．

7 まとめ

本研究では，エキスパートによる解析情報から悪性な挙動を自動抽出する手法，抽出した挙動を動的解析ログから自動検出する手法を開発した．この手法を用いて 5,646 のマルウェア検体を分析した結果から，マルウェアの挙動は必ずしも種別に特有出ないこと，およびアンチウイルスソフトウェアの検出において種別が不明であったマルウェア検体からも悪性挙動が検出できることを示した．動的解析ログから抽出した悪性挙動から解析レポートの草案を自動作成する技術の開発は今後の課題である．


```
This threat drops a copy of itself to a folder with a random
file and folder name, such as:
    %windir%\<folder>\
    %windir%\system32\<folder>\
    %APPDATA%\<random>

It changes the following registry entry so that it runs each
time you start your PC:
    In subkey: HKLM\Software\Microsoft\Active Setup\
        Installed Components\{<GUID>}
    Sets value: "StubPath"
    With data: "<location and name of malware file>.exe
        restart"

For example:
    In subkey: HKLM\Software\Microsoft\Active Setup\
        Installed Components\
        {08B0E5JF-4FCB-11CF-AAA5-00401C6XX500}
    Sets value: "StubPath"
    With data: “%windir%\system32installserver.exe restart"

Removable drives
This worm spreads by copying itself with one of the following
file names to all accessible removable drives:
    system.exe
    task.exe
    update.exe
    winbackup.exe
    windows.exe

It sends the logged and collected information to
a remote server. Some of the command and control (C&C)
servers we have seen it try to connect to in the wild are:
    extremesc.no-ip.org
    hopto.dynu.com
    mateusmacedo.no-ip.org
    ralacapeta.no-ip.biz
    zerocool6.no-ip.biz
```


図 11: 正解レポートの例 [8]

参考文献

[1] Kaspersky Lab, “Kaspersky Lab is Detecting 325,000 New Malicious Files Every Day.” http://www.kaspersky.com/about/news/virus/2014/Kaspersky-Lab-is-Detecting-325000-New-Malicious-Files-Every-Day.

[2] 神薗雅紀他, “マルウェア対策のための研究用データセット ～ MWS Datasets 2014～.” MWS 2013 http://www.iwsec.org/mws/2014/, Oct 2013.

[3] “VirusTotal.” http://www.virustotal.com.

[4] Microsoft Malware Protection Center, “Naming malware.” http://www.microsoft.com/security/portal/mmpc/shared/malwarenaming.aspx.

[5] Windows Dev Center, “CopyFile function.” https://msdn.microsoft.com/en-us/library/windows/desktop/aa363851(v=vs.85).aspx.

[6] Windows Dev Center, “CopyFile2 function.” https://msdn.microsoft.com/en-us/library/windows/desktop/hh449404(v=vs.85).aspx.

[7] Japan Network Information Center, “ドメイン名の種類.” https://www.nic.ad.jp/ja/dom/types.html#kinds-gtld.

[8] Microsoft Malware Protection Center, “Worm:Win32/Rebhip.A.” www.microsoft.com/security/portal/threat/encyclopedia/entry.aspx?Name=Worm:Win32/Rebhip.A.

[9] F. Ahmed, H. Hameed, M. Z. Shafiq, and M. Farooq, “Using spatio-temporal information in api calls with machine learning algorithms for malware detection,” in *Proceedings of the 2Nd ACM Workshop on Security and Artificial Intelligence*, AISec ’09, (New York, NY, USA), pp. 55–62, ACM, 2009.

[10] M. Alazab, S. Venkataraman, and P. Watters, “Towards Understanding Malware Behaviour by the Extraction of API Calls,” in *Cybercrime and Trustworthy Computing Workshop (CTC), 2010 Second*, pp. 52–59, 2010.

[11] C. Ravi and R. Manoharan, “Malware detection using windows api sequence and machine learning,” *International Journal of Computer Applications*, vol. 43, pp. 12–16, April 2012. Published by Foundation of Computer Science, New York, USA.

[12] U. Bayer, P. M. Comparetti, C. Hlauschek, C. Krgel, and E. Kirda, “Scalable, behavior-based malware clustering.,” in *NDSS*, The Internet Society, 2009.

[13] P. Li, L. Liu, D. Gao, and M. K. Reiter, “On challenges in evaluating malware clustering,” in *Proceedings of 13th International Symposium on Recent Advances in Intrusion Detection*, RAID 2010, pp. 238–255, 2010.