API呼び出しとシステム負荷を用いたマルウェアの特徴抽出に関する一検討

佐藤 順子[1]　三須 剛史[1]　花田 真樹[1]　鈴木 英男[1]　布広 永示[1]

概要：近年のマルウェアの急激な増加に伴い，短期間でマルウェアの挙動が解析可能な動的解析が注目され，動的解析を用いたマルウェア検知手法が数多く提案されている．本研究では，API 呼び出しとその呼び出しに伴うシステム負荷に着目し，マルウェア検知の判定に有用なマルウェアの特徴抽出を行う．これまでに，API 呼び出しの遷移や API 呼び出し時の引数の変化など，API に着目した動的解析を用いたマルウェア検知手法が提案されている．本稿では，従来より提案されている API 呼び出しの遷移に加えて，マルウェアが身を隠すために有している性質として，API 呼び出しの時間間隔と API 呼び出しに伴うシステム負荷に着目し，マルウェア検知の判定に有用なマルウェアの特徴に関して分析した結果を報告する．

キーワード：動的解析，API，システム負荷

A Study of Characteristic of Malware Based on API Call Sequence and System Load Status Analysis

Junko Sato[1]　Takeshi Misu[1]　Masaki Hanada[1]　Hideno Suzuki[1]　Eiji Nunohiro[1]

Abstract: With rapid increase of malware, dynamic analysis has been focused because dynamic behavior of malware can be analyzed in a short time. In this research, we extract characteristic of malware based on API call sequence and system load status analysis for malware detection. In the past research, methods of malware detection based on API call (e.g., API call pattern and dynamic change of API arguments) were proposed. In this paper, in addition to API call pattern used in the past research, we focus on system load status on API call and time period of API call, and show the results of characteristic of malware using the system load status on API call and time period of API call.

Keywords: Dynamic Analysis, API, System Load Status

1. はじめに

近年，マルウェアが増加の一途をたどっており，今後も脅威が増大する可能性が高い．McAfee Labs[1] では，直近の 2 年間で収集したマルウェアの総数が 2 倍に増加したというレポートを発表している．このようなマルウェアの急激な増加に伴い，短期間でマルウェアの挙動が解析可能な動的解析が注目され，動的解析を用いたマルウェア検知手法が数多く提案されている．本研究では，API 呼び出しとその呼び出しに伴うシステム負荷に着目し，マルウェア検知の判定に有用なマルウェアの特徴抽出を行う．動的解析では，レジストリ，ファイル・ディレクトリ，ネットワークの状況確認が不可欠であり，その有力な手段として，API 呼び出しや API 呼び出し時の引数が用いられている．これまでに，API に着目したマルウェア検知に関する研究では，API 呼び出しの遷移・パターンに着目した手法 [2] やマルウェア実行毎に API 呼び出し時の引数の異なる点に着目した手法 [3] などが提案されている．

本研究では，従来より提案されている API 呼び出しの遷移・パターンに加えて，マルウェアが身を隠すために有し

[1] 東京情報大学 総合情報学部
Department of Information Sciences, Tokyo University of Information Sceinces

ている性質として，API 呼び出しの時間間隔と API 呼び出しに伴うシステム負荷に着目し，マルウェア検知の判定に有用なマルウェアの特徴に関して分析を行う．

一般的なマルウェアでは，防御機能として，検知を回避する機能や解析を困難にする機能が実装されており，主に，(1) デバッガや実行環境の検知，(2) セキュリティサービスの無効化，(3) 暗号化／圧縮による難読化，(4) ステルス化（身を隠すための機能）などが挙げられる．本研究では，上記 (4) に着目し，身を隠すために API 呼び出しの時間間隔や API 呼び出し時のシステム負荷に特徴が表れると想定し，その特徴を抽出する．

本稿では，まず **2.** で本研究の関連研究について述べる．**3.** では，マルウェア検知の判定に有用なマルウェアの特徴とその解析方法について述べる．**4.** で，解析結果について述べる．**5.** で，まとめと今後の課題について述べる．

2. マルウェアの防御機能と関連研究

まず一般的なマルウェアの特徴である防御機能について述べる．その後，関連研究について述べる．

(1) デバッガや実行環境の検知
感染しようとするマルウェアがデバッガの実行を検知し，動作を実行せずに終了する．また，感染しようとするマルウェアが実行環境を確認し，仮想環境であった場合に動作を行わない，特定の日時や一定時間が経過するまで動作を行わない，インターネット接続ができない場合は動作を行わない．

(2) セキュリティサービスの無効化
ウィルス対策ソフトウェアやファイアフォール，Windows Update 等の機能を停止する．これにより，マルウェアが駆除されることを回避したり，マルウェアの動作の妨げとなるサービスを停止する．

(3) 暗号化／圧縮による難読化
暗号化／圧縮により，ウィルス対策ソフトウェアが行うパターンマッチによるマルウェア検知を困難にする．また，アナリストによるマルウェア解析に対する時間を稼ぐ効果もある．

(4) ステルス化（身を隠すための機能）
マルウェア自身の存在を隠蔽したり，痕跡を消去したりする機能である．マルウェアが実行されているにも関わらず，そのプロセスを隠蔽したり，マルウェアが動作した形跡をログから消去する．

上記で述べた一般的なマルウェアの防御機能（(1) デバッガや実行環境の検知，(2) セキュリティサービスの無効化，(3) 暗号化／圧縮による難読化，(4) ステルス化（身を隠すための機能））を用いたマルウェア検知手法が提案されている．

デバッガや実行環境の検知（上記 (1)）を利用した手法として，[4] では，解析システムとそのシステムと異なる環境の 2 つを用意し，両環境でのマルウェアの挙動の変化を利用して，検知する手法を提案している．セキュリティサービスの無効化（上記 (2)）を利用した手法として，[5] では，疑似的なセキュリティサービスのプロセスを実行させ，そのセキュリティサービスのプロセスを強制終了させるマルウェアの挙動を利用して，検知する手法を提案している．

本研究に類似した関連研究として，API 呼び出しの遷移・パターンに着目した研究 [2] とマルウェア実行毎に API 呼び出し時の引数の異なる点に着目した研究 [3] がある．[2] では，マルウェアの API 呼び出しのバターンを抽出し，データマイニングツールを用いて，検知する手法を提案している．[3] では，マルウェア実行毎に API 呼び出し時の引数の異なる点に着目し，マルウェアを 2 回実行させ，その引数の変化があれば，マルウェアとして判定する検知手法を提案している．なお，[3] は，身を隠すための機能（上記 (2)）を利用した検知手法と考えられる．

3. マルウェアの特徴抽出のための解析

本研究で着目するマルウェアの特徴について述べ，その後，解析の流れについて述べる．

3.1 マルウェアの特徴抽出

本研究では，マルウェアが身を隠すために，API 呼び出しの時間間隔や API 呼び出し時のシステム負荷に特徴が表れると想定し，その特徴を抽出する．

マルウェアのファイルの読み込み・書き込みや乱数計算におけるシステム負荷が急激に高くなると，通常の状況と異なるためにユーザに気づかれる可能性がある．そのため，マルウェアが身を隠すために，API 呼び出し時のシステム負荷に特徴が表れると考えられる．また，システム負荷が変化すると，API 呼び出しの時間間隔にも影響があると考えられる．さらには，特定の時間的な条件を満たさないと動作を行わないマルウェアも存在するので，この点からも，API 呼び出しの時間間隔に特徴が表れると考えられる．

本研究では，既に提案されている API 呼び出しの遷移・パターンに加えて，API 呼び出しの時間間隔と API 呼び出しに伴うシステム負荷に着目し，マルウェアの特徴を抽出する．

本研究では，API 呼び出しの時間間隔および API 呼び出しに伴うシステム負荷に対する，API 呼び出しの連鎖した出現パターンを抽出する．API 呼び出しの連鎖した出現パターンに関しては，[2] に従い，抽出する．[2] で実施されている連鎖数による API 呼び出しパターンを図 1 に，本研究の API 呼び出しの時間間隔および API 呼び出しに伴うシステム負荷に対する，連鎖数による API 呼び出しパターンを図 2 に示す．図 1 が API 呼び出しパターンのみに着目しているのに対し，図 2 では，API 呼び出し時の時

図 1 連鎖数による API 呼び出しパターン

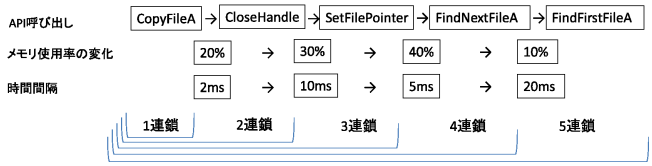

図 2 時間間隔およびシステム負荷に対する，連鎖数による API 呼び出しパターン

間間隔とシステム負荷の状態も考慮した API 呼び出しパターンとなっている．これにより，正規プログラムとマルウェアでは，同じ API 呼び出しパターンでも，API 呼び出し時の時間間隔とシステム負荷に変化があれば，それを特徴として抽出可能となる．

3.2 特徴抽出のための解析の流れ

本研究におけるマルウェアの特徴抽出のための解析の流れを以下に示す．

(1) マルウェア検体と正規プログラムのそれぞれに対して動的解析を行い，API 呼び出しの時刻も含めて API 呼び出しのログを記録する．また，マルウェア検体と正規プログラムの実行時にはシステム負荷（CPU 使用率，メモリ使用率，ハードディスク使用率，ネットワーク使用率など）も同時に取得し，ログに記録する．

(2) API 呼び出しのログから，API 呼び出しの時間間隔を算出する．また，API 呼び出しの時間間隔におけるシステム負荷の平均を算出する．

(3) API 呼び出しのログから，連鎖数による API 呼び出しパターンに対する出現度数（出現頻度）を取得する (図]1 の出現度数（出現頻度)）．また，時間間隔，システム負荷，連鎖数による API 呼び出しパターンに対する出現度数（出現頻度）を取得する（図 2 の出現度数 (出現頻度)）．

(4) マルウェア検体と正規プログラムのそれぞれに対して，連鎖数による API 呼び出しパターンに対する出現度数（出現頻度）と時間間隔，システム負荷，連鎖数による API 呼び出しパターンに対する出現度数（出現頻度）を比較する．

4. 解析実験と結果

本節では，本解析実験の環境と検体，解析結果について述べる．

4.1 解析実験の環境と検体

本解析実験では，動的解析を行うために，Cuckoo Sandbox[6] を用いた．仮想環境には Windows XP と Windows 7 を用いた．なお，Cuckoo Sandbox では，API 呼び出しの時刻がミリ秒単位でしか取得できないため，マイクロ秒で取得できるように変更を加えている．

マルウェア検体に関しては，CCCDataset 2013[7] の 4648 個の検体を使用した．また，正規プログラムに関しては，窓の杜 [8] で公開されているフリーのプログラム中からダウンロードした 260 個の検体を使用した．

4.2 解析結果

まず，連鎖数による API 呼び出しパターン（図]1）に関して，マルウェア検体と正規プログラムの比較を行う．その後，時間間隔と連鎖数による API 呼び出しパターン (図]2）に関して，マルウェア検体と正規プログラムの比較を行う．システム負荷に関しては今後の課題としており，解析結果では，時間間隔と連鎖数による API 呼び出しパターンに関する比較結果を示す．なお，本解析実験では，連鎖数 2 のみで実験を行った．

4.2.1 連鎖数による API 呼び出しパターンの結果

マルウェア検体の連鎖数（連鎖数 2）による API 呼び出しパターンに対する出現頻度を図 3 に，正規プログラムの連鎖数（連鎖数 2）による API 呼び出しパターンに対する出現頻度を図 4 に示す．図 3 の横軸は，API 呼び出しパターン（2 つの API をスペース区切りで結合して表記）であり，縦軸は出現頻度である．マルウェア検体（図 3）に関しては，出現頻度の高い順から表示させており，正規プログラム（図 4）の横軸も図 3 と同じ API 呼び出しパターンにしている．なお，出現頻度は，マルウェア検体と正規プログラムの検体数に差があるために，最大出現度数を 100 として正規化している．

図 3 と図 4 より，連鎖数 2 において，マルウェア検体と正規プログラムの出現頻度の分布には，明らかな違いが見られた．なお，本解析実験では，マルウェア検体と正規プログラムの出現頻度（出現度数）に関して，識別判定可能な統計的手法による比較は行っていない．これに関しては，今後の課題としている．

4.2.2 時間間隔と連鎖数による API 呼び出しパターンの結果

マルウェア検体の時間間隔と連鎖数（連鎖数 2）による API 呼び出しパターンに対する出現頻度を図 5 に，正規プログラムの時間間隔と連鎖数（連鎖数 2）による API 呼び出しパターンに対する出現頻度を図 6 に示す．時間間隔の区間は，150 マイクロ秒未満とそれ以上の 2 区間としている．

図 5 の横軸は，時間間隔と API 呼び出しパターンであり，縦軸は出現頻度である．横軸は，2 つの API と時間間隔（150 マイクロ秒未満が 0，150 マイクロ秒以上が 1 と表記）をスペース区切りで結合して表記している．マル

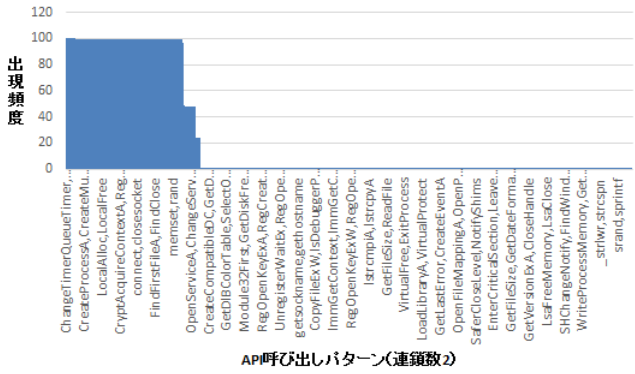

図 3　マルウェア検体の連鎖数（連鎖数 2）による API 呼び出しパターンに対する出現頻度

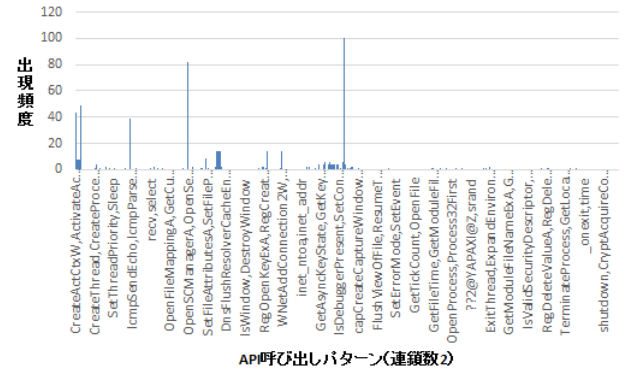

図 4　正規プログラムの連鎖数（連鎖数 2）による API 呼び出しパターンに対する出現頻度

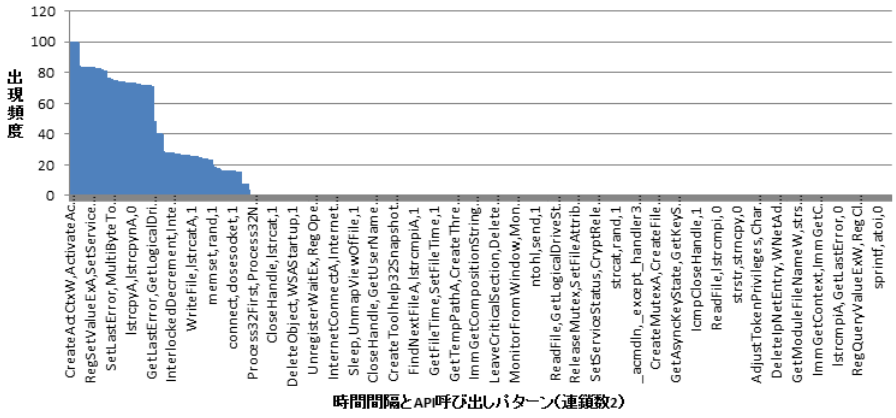

図 5　マルウェア検体の時間間隔と連鎖数（連鎖数 2）による API 呼び出しパターンに対する出現頻度

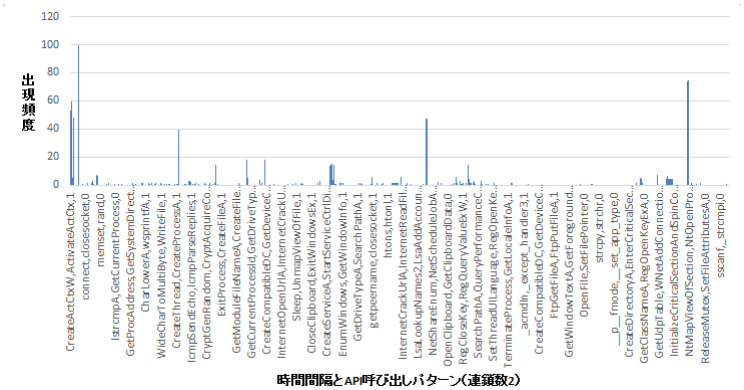

図 6　正規プログラムの時間間隔と連鎖数（連鎖数 2）による API 呼び出しパターンに対する出現頻度

ウェア検体（図 5）に関しては，出現頻度の高い順から表示させており，正規プログラム（図 6）の横軸も図 5 と同じ API 呼び出しパターンにしている．なお，出現頻度は，マルウェア検体と正規プログラムの検体数に差があるために，最大出現度数を 100 として正規化している．

図 5 と図 6 より，連鎖数 2 において，マルウェア検体と正規プログラムの出現頻度の分布には，明らかな違いが見られた．また，時間間隔を考慮せず，連鎖数だけによる解析結果（**4.2.1** 節）と比較しても，その違いが顕著になっていると考えられる．なお，**4.2.1** 節と同様に，マルウェア検体と正規プログラムの出現頻度（出現度数）に関して，識別判定可能な統計的手法による比較は行っていない．これに関しても，今後の課題としている．

5. まとめと今後の課題

本稿では，既に提案されている API 呼び出しの遷移・パターンに加えて，マルウェアが身を隠すために有している性質として，API 呼び出しの時間間隔と API 呼び出しに伴うシステム負荷に着目し，マルウェア検知の判定に有用なマルウェアの特徴を抽出した．解析結果より，従来の API 呼び出しの遷移・パターンだけでなく，時間間隔を加えることにより，マルウェア検体と正規プログラムの違いが顕著となる結果が得られた．

今後は，マルウェアの検体数を増やし，ボット以外のマルウェア検体における調査を進めて行く予定である．また，マルウェア検体と正規プログラムの識別判定を行うアルゴリズムを取り入れ，マルウェア検知手法の検討を行う予定である．

謝辞　本研究を進めるにあたって，株式会社日立システムズ関係者各位に助言と協力を頂きました．ここに深く感謝します．

参考文献

[1] Bruce Snell, Quarterly Threat Report: What Do the Numbers Mean to Me?, https://blogs.mcafee.com/consumer/quarterly-threat-report-numbers-mean/, 参照 August. 12, 2016.

[2] D. Balzarotti, M. Cova, C. Karlberger, C. Kruegel, E. Kirda, and G. Vigna, “Efficient Detection of Split Personalities in Malware,” In Proceedings of 17th Annual Network and Distributed System Security Symposium (NDSS 2010), 2010.

[3] 松木 隆宏, 新井 悠, 寺田 真敏, 土居 範久, “セキュリティ無効化攻撃を利用したマルウェアの検知と活動抑止手法の提案,” 情報処理学会論文誌 Vol.50, No.9, pp.2127-2136, 2009.

[4] 青木 一樹, 後藤 滋樹, “マルウェア検知のための API コールパターンの分析,” 電子情報通信学会 総合大会, D-19-3, 2014.

[5] 笠間 貴弘, 吉岡 克成, 井上 大介, 松本 勉, “実行毎の挙動の差異に基づくマルウェア検知手法の提案,” 情報処理学会 コンピュータセキュリティシンポジウム, Vol.2011, No.3, pp.726 - 731, 2011.

[6] Norman Sandbox, http://www.norman.com/security_center/security_tools/, 参照 August. 12, 2016.

[7] 高田 雄太 , 寺田 真敏 , 村上 純一 , 笠間 貴弘 , 吉岡 克成 , 畑田 充弘, “マルウェア対策のための研究用データセット ～ MWS 2016 Datasets ～,” 情報処理学会, Vol.2016-CSEC-74, No.17, 2016 年 7 月.

[8] 窓の杜, http://www.forest.impress.co.jp/, 参照 August. 12, 2016.