L-033

システムコールを用いたマルウェア検知における機械学習アルゴリズムの比較
Comparing Machine Learning Algorithms in Malware Detection Based on System Calls

梶原 友希† 鄭 俊俊† 毛利 公一†
Yuki Kajiwara Junjun Zheng Koichi Mouri

1. はじめに

近年，既存のマルウェアに加え，未知のマルウェアが毎年数多く発見されており，マルウェアによる脅威の拡大が問題となっている．そのため，マルウェアを正確に検知し，被害の予防や防止などの対策を早期に行うことが求められる．マルウェア検知において，機械学習を用いたマルウェアの挙動をベースとする検知手法が提案されている．機械学習は，コンピュータにさまざまなデータを学習させ，その学習経験に基づいてデータの予測や推定をさせる技術であり，教師あり学習アルゴリズムと教師なし学習アルゴリズムの 2 つに大別される．

機械学習を用いたマルウェアの挙動をベースとする検知手法には，静的解析や動的解析ログを用いるものがある．しかし，静的解析の場合，マルウェアのプログラムにパッキングや難読化などが施されるとその機能を検知できない可能性がある．そのため，難読化などの影響を受けない動的解析ログを用いた手法が有効とされる [1, 2]. また，マルウェアの動的解析手法の 1 つとして，マルウェアが発行したシステムコールを観測する手法がある．マルウェアが，ファイル操作や通信などを行うことでシステムに影響を与えるためには，必ずシステムコールの発行が必要であることから，システムコールトレースログはマルウェアの挙動把握に有効であると言える．また，動的解析の中でも，高速で網羅的にログの取得が可能であり，挙動の理解が容易なログが提供できるため，単純な特徴量からマルウェア検知に有用であると言えるものを明らかにしやすいと考えた．そこで，機械学習を用いた検知精度の高いシステムコールベースのマルウェア検知手法の実現を目指す．

本稿では，システムコールトレーサ Alkanet [3] を利用して取得したシステムコールトレースログを用いて，機械学習によるマルウェア検知を行い，その学習アルゴリズム間でのマルウェア検知精度の比較を行う．新たな入力データがマルウェアか否かを判別する 2 クラス分類を行うため，教師あり学習アルゴリズムを用いる．具体的には，機械学習で学習させるための訓練データとして，各プロセスの実行中に発行されたシステムコールを用いて，ロジスティック回帰，ランダムフォレスト，SVM（Support Vector Machine）の 3 つの手法による機械学習を実現し，検知精度の比較を行う．以下，2 章でシステムコールトレースログを用いたマルウェア検知手法について，3 章でマルウェア検知の比較評価実験について述べる．4 章で議論について，5 章で関連研究について述べ，最後に 6 章でまとめる．

†立命館大学
Ritsumeikan University

2. システムコールベースのマルウェア検知手法

本稿では，システムコールトレースログから特徴量を生成し，複数の機械学習アルゴリズムよるマルウェア検知を行う．本章では，検知手法におけるシステムコール列の抽出および特徴量の生成，使用した学習アルゴリズムについて述べる．

2.1 システムコール列の抽出

本稿では，機械学習で学習させるための訓練データとして，各プロセスの実行中に発行されたシステムコールを発行順に並べたものを，マルウェアの挙動を表すものとして使用する．以下，プロセスが発行したシステムコールを発行順に並べたものを「システムコール列」とする．システムコールを観測できる動的解析環境として，システムコールトレーサ Alkanet を利用し，取得したシステムコールトレースログ（以下，Alkanet ログ）を使用する．

2.1.1 Alkanet ログ

Alkanet は，本研究室で開発されている動的解析環境であり，マルウェアの挙動を観測する．仮想計算機モニタ（VMM: Virtual Machine Monitor）である BitVisor の拡張機能として動作し，ゲスト OS 上のプロセスが発行したシステムコールのログを記録する．具体的には，システムコールのエントリポイントがカーネル空間に存在するため，ユーザモードで動作するプロセスからはアクセスすることができず，エントリポイントを回避してシステムコールを呼び出すこともできないことを利用し，システムコールをフックすることでシステムコールトレースを実現している．そのため，マルウェアの挙動を確実に観測可能である．Alkanet には，マルウェアの耐解析機能によって解析環境を検知されにくいという特徴があり，動的解析妨害機能を持つマルウェアの挙動も観測可能であるため，マルウェアの解析に有効である．また，システムコールの発行に対して，sysenter 時のログと sysexit 時のログの 2 つのログを記録するが，本稿では sysenter 時のシステムコールのみを抽出するようフィルタリングを行い，システムコール列とする．

2.1.2 マルウェアデータとクリーンウェアデータ

本稿では，悪性な挙動を示すマルウェアのログ（以下，マルウェアログ）と悪性な挙動を示さないプログラム（以下，クリーンウェア）のログを利用する．

マルウェアログとして，プロセスの実行中に発行されたシステムコールを記録したシステムコールトレースログを使用する．マルウェアプロセスのプライマリスレッドのみを対象としてシステムコール列を抽出し，プロセス生成やコードインジェクションは考慮しないものとする．また，クリーンウェアとして，マルウェアログと同様に取得したシステムコールトレースログ（以下，クリーンウェアログ）を使用する．クリーンウェアは，プ

表 1　Alkanet がトレースするシステムコール

カテゴリ	システムコール名
File	NtCreateFile, NtOpenFile, NtWriteFile, NtReadFile, NtSetInformationFile, NtDeleteFile
Registry	NtCreateKey, NtOpenKey, NtQueryKey, NtSetInformationKey, NtDeleteKey, NtSetValueKey, NtQueryValueKey, NtDeleteValueKey
Virtual Memory	NtWriteVirtualMemory, NtReadVirtualMemory, NtAllocateVirtualMemory, NtProtectVirtualMemory
File Mapping	NtCreateSection, NtOpenSection, NtMapViewOfSection
Network	NtDeviceIoControlFile
Process	NtCreateProcess, NtCreateProcessEX, NtTerminateProcess
Thread	NtCreateThread, NtSuspendThread, NtResumeThread, NtTerminateThread, NtGetContextThread, NtSetContextThread, NtQueueApcThread
Driver	NtLoadDriver, NtUnloadDriver
Time	NtQueryPerformanceCounter
Sleep	NtDelayExecution

ライマリスレッド以外のスレッドも考慮し，クリーンウェアログから各プロセスのスレッド ID（TID）ごとにシステムコール列を抽出する．

2.2 特徴量生成

本稿では，36 種類のシステムコールの各出現回数とシステムコール列内における 1296 パターンの遷移の各回数およびそれらの確率をそれぞれ特徴量として使用する．これは，マルウェアがシステムに影響を与えるためには，必ずシステムコールが必要となることから，マルウェアの挙動がシステムコールの発行順序や発行回数に反映される可能性が高いと考えられるためである．また，ログ内におけるシステムコールの出現および遷移に関する情報は，既存研究でも特徴量として使用され，高い検知精度を出しているものが多い [1, 4]．今回のマルウェア検知では，出現や遷移情報といった単純な特徴量を使用することで，特徴量として有用と言える情報を明らかにする．

2.2.1 システムコールの出現回数および確率

Alkanet がトレース対象としている 36 種類のシステムコールを用いて，各システムコール列内に出現する回数を特徴量とする．Alkanet がトレースするシステムコールを表 1 に示す．これらは，マルウェアの挙動を把握するのに重要とされるシステムコールであり，ファイル操作やレジストリ操作に加え，プロセス生成などの挙動を把握することができる．また，各システムコールが出現する確率を特徴量として使用し，出現回数を特徴量とした場合の検知精度と違いがあるかの検証を行う．

2.2.2 システムコール列内における遷移回数および確率

あるシステムコールが別のシステムコールへの遷移回数を特徴量とする．Alkanet がトレースする 36 種類のシステムコールの全遷移パターンは，1296（= 36×36）パターンあり，各遷移をカウントする．また，各遷移における遷移確率を特徴量として使用し，遷移回数を特徴量とした場合の検知精度と違いがあるかの検証を行う．システムコール列内において，システムコール S_i からシステムコール S_j への遷移確率 p_{ij} は以下のようになる．

$$p_{ij} = count(S_i \rightarrow S_j) / \sum_{k=1}^{36} count(S_i \rightarrow S_k)$$

表 2　混合行列

実際＼予測	マルウェア	クリーンウェア
マルウェア	TP：真陽性 (True Positive)	FN：偽陰性 (False Negative)
クリーンウェア	FP：偽陽性 (False Positive)	TN：真陰性 (True Negative)

2.3 学習アルゴリズム

本稿では，ロジスティック回帰，ランダムフォレスト，SVM の 3 つの手法による機械学習を実現し，検知精度の比較を行う．今回，学習に使用するデータ数が，深層学習を行うほど多くないため，データ数が少なくてもある程度の正確性があり，また，調整するパラメータ数が少なく，学習時間の短縮が行えるようなアルゴリズムとして，上記 3 つの手法に着目する．

回帰は，1 つの依存変数（予測される値：目的変数）と 1 つ以上の独立変数（予測するための値：説明変数）との関係を規定するものである．ロジスティック回帰は，教師あり学習手法の 1 つであり，目的変数が 2 値の場合にその分類を行う．シグモイド関数を用いて予測モデルを作成し，説明変数から確率的に目的変数を予測する．

ランダムフォレストは，決定木を複数用いたアンサンブル学習である．異なった方向に過剰適合した決定木を複数作成し，それらの予測結果の平均を取ることで過学習の度合いを減らす．分類の場合，各決定木による予測結果が出た後，多数決によって最終予測結果を出力する．

SVM では，データとその特徴の値を表現する多次元世界にプロットされたデータポイントの間に境界を作る平面を求めることで予測・分類を行う．この境界を超平面（Hyperplane）と呼ぶ．超平面は，多次元宇宙における平面である．特にパターン認識での応用が多いが，学習データが増えると計算量が膨大になり，予測結果の解釈が難しいという欠点がある．

2.4 評価指標

本稿では，10 分割交差検証を用いて学習を行い，マルウェア検知の結果を評価する．評価指標として，混合行列を基に適合率，再現率，F 値を算出して使用する．混合行列は，実際のクラスを縦軸，モデルが予測したクラスを横軸として表した行列である．マルウェアもしくはクリーンウェアの 2 クラスに分類する場合の混合行列の各要素を表 2 に示す．

適合率と再現率は，混合行列の結果をまとめる方法として最もよく使用される指標である．適合率の値が大きいほど，誤検知が少なく，再現率の値が大きいほど，検知漏れが少ないと判断できる．適合率と再現率は，トレードオフの関係にあり，これらのバランスを示す値が F

表 3　ロジスティック回帰の場合

特徴量	適合率	再現率	F 値
出現回数	0.897143	0.872222	0.884507
出現確率	0.873016	0.916667	0.894309
遷移回数	0.966667	0.966667	0.966667
遷移確率	0.956757	0.983333	0.969863

表 4　ランダムフォレストの場合

特徴量	適合率	再現率	F 値
出現回数	0.947090	0.994444	0.970190
出現確率	0.941489	0.983333	0.961957
遷移回数	0.957447	1.0	0.978261
遷移確率	0.937173	0.994444	0.964960

値である．F値は，適合率と再現率の調和平均値であり，使用したデータセットの各データがどれだけ正しく分類できているかを表す指標である．各値を算出する式をそれぞれ以下に示す．

- 適合率 = TP / (TP + FP)
- 再現率 = TP / (TP + FN)
- F 値 = (2 ×適合率×再現率) / (適合率 + 再現率)

3. マルウェア検知の比較評価実験

本章では，使用データ，実験手順および学習アルゴリズムのマルウェア検知精度の比較評価について述べる．また，既存研究の評価値との比較について論じる．

3.1 実験データおよび手順

Alkanet の実行環境は，ゲスト OS として，Windows XP servicePack3 を利用する．マルウェアとして，MWS データセット [5] に記録されている検体を利用し，取得されたログを用いる．また，クリーンウェアとして，ゲスト OS にプリインストールされたアプリケーションを利用して取得されたログを考える．マルウェアとクリーンウェアそれぞれ 180 個からシステムコール列を抽出して得られた合計 360 個のデータからデータセットを作成した．また，実験手順は：1）システムコール列の抽出；2）4 種類の各特徴量の算出；3）学習手法ごとに各特徴量を用いてマルウェア検知；と 4）比較評価である．

3.2 結果

ロジスティック回帰，ランダムフォレスト，SVM を用いて 10 分割交差検証を行い，4 種類の各特徴量における検知結果を算出した．各実験結果をそれぞれ表 3，表 4，表 5 に示す．また，手法ごとに，最良の結果となった特徴量，精度，F 値についてまとめたものを表 6 に示す．精度は，マルウェア検知の正確度を示す数値であり，混合行列の各要素を用いて以下の式で算出する．

精度 = (TP + TN) / (TP + TN + FP + FN)

3.3 比較評価

表 3，表 4，表 5，表 6 より，ロジスティック回帰の場合は遷移確率，ランダムフォレストの場合は遷移回数，SVM の場合は遷移回数を特徴量としたときが，F 値，精度ともに高い数値を示した．したがって，これらの特徴量を用いることで，各手法において最良の結果を得られると言える．

表 5　SVM の場合

特徴量	適合率	再現率	F 値
出現回数	0.900524	0.955556	0.927224
出現確率	0.943820	0.933333	0.938547
遷移回数	0.972067	0.966667	0.969359
遷移確率	0.946237	0.977778	0.961749

表 6　最良の結果

手法	特徴量	精度	F 値
ロジスティック回帰	遷移確率	0.969444	0.969863
ランダムフォレスト	遷移回数	0.977778	0.978261
SVM	遷移回数	0.969444	0.969359

表 6 より，最良の各結果を比較すると，今回使用したデータセットに関しては，特徴量を遷移回数としたランダムフォレストによる学習が，データセットへの当てはまりが良く，マルウェア検知に有効な手法であると考えられる．また，表 3，表 4，表 5 より，特徴量ごとに各手法の F 値を比較すると，遷移確率の場合はロジスティック回帰が最高値であるが，それ以外の特徴量の場合はランダムフォレストが最高値である．したがって，どの特徴量を用いた場合でも，今回使用したデータセットに関しては，ランダムフォレストで検知を行うことで，他手法に比べ，安定して高い精度を得られると考えられる．

表 6 より，ロジスティック回帰，ランダムフォレスト，SVM の各手法による精度は，どれも約 96%を超えていた．このことから，システムコール列内における各システムコールの出現回数や確率，システムコール間の各遷移回数や確率は，マルウェアを検知するための特徴量として有用であると言える．特に，遷移回数や確率といったシステムコールの遷移情報を用いた場合は，より高い精度で検知可能であると言える．

表 3 より，ロジスティック回帰の場合，出現回数よりも出現確率の F 値の方が高く，遷移回数よりも遷移確率の F 値の方が高くなった．このことから，回数よりも確率を特徴量とした方が，検知精度が良いことが分かる．また，出現情報よりも遷移情報の方が，F値が高い．したがって，ロジスティック回帰を用いる場合，出現情報よりも遷移情報，また，回数よりも確率を特徴量とする方が，マルウェアを検知する上で有効であると考えられる．

表 4 より，ランダムフォレストの場合，出現確率よりも出現回数の F 値の方が高く，遷移確率よりも遷移回数の F 値の方が高くなった．このことから，確率よりも回数を特徴量とした方が，検知精度が良く，ロジスティック回帰の場合とは逆の結果となっていることが分かる．遷移回数を特徴量とした場合の再現率は 100%であり，検知漏れのデータが 1 つも無かったことを表している．他の特徴量においても再現率が高いことから，ランダムフォレストの場合，他の手法と比べて検知漏れが特に少なくなるということが分かる．

表 5 より，SVM の場合，出現回数よりも出現確率の F 値の方が高く，遷移確率よりも遷移回数の F 値の方が高くなった．このことから，出現情報については回数の方が，遷移情報については確率の方が，マルウェアをより正確に検知可能となると言える．しかし，ロジスティック回帰やランダムフォレストとは違い，F値が高くなると再現率が低くなり，適合率が高くなっていた．誤検知は

表 7　文献 [4] における各評価値

	F 値	TPR	FPR
文献[4]	0.982773	0.975570	0.01

表 8　各文献における精度

	文献 [1]	文献 [6]
精度	94.2%，93.4%	97.7%

少なくなったが，検知漏れしたデータが増えたと考えられるため，実際に使用する際は，適合率と再現率のトレードオフの関係を検討するべきであると考える．

使用したデータセットや特徴量，機械学習手法などが違うため，厳密な比較はできないが，既存研究の中における評価値の良し悪しの目安として，文献 [4] における F 値，TPR，FPR，文献 [1, 6] における精度の数値のみをそれぞれ表 7，表 8 に示す．文献 [4] は，システムコールシーケンスをマルコフ連鎖に落とし込み，逆伝播ニューラルネットワークを用いてマルウェア検知を行う手法を提案した．文献 [1] は，システムコールの遷移に着目し，マルコフ連鎖を用いたマルウェア検知手法を考えて，2 つのデータセットに対して検知を行った．文献 [6] は，ANN（Artificial Neural Network）を用いたマルウェア検出手法について論じた．

4. 議論

今回使用したデータ数は，360 個であり，文献 [1, 4, 6] に比べると少ない．データ数は，多い方が有用である場合が多く，また，モデルを改良するよりも，単純にデータ数を増やした方が精度向上が期待できる場合がある．そのため，データ数は，可能な限り多く用意するべきである．また，その際，類似データを収集するのではなく，様々な種類のデータを収集し，過剰適合の抑制に努める．

今回，アルゴリズム選択の際に学習時間についても選択基準として挙げていたため，各手法における学習時間の比較も行うべきであった．したがって，今後の課題として，学習時間の比較実験を検討する．また，今回対象としたシステムコール間の遷移は 1296 パターンあるため，遷移回数や遷移確率を特徴量としたとき，特徴量が多く，学習に時間がかかった．そのため，特徴量の削減を行い，学習時間の短縮を検討すべきである．特徴量の削減を行うことで，検知精度の向上も期待できるため，どの遷移がマルウェア検知に有用であるかを明確化することも可能であると考える．

システムコールを用いたマルウェア検知では，ニューラルネットワークやマルコフ連鎖などを利用したものが多く，表 7 や表 8 を見るとその精度は高い．そのため，本稿で使用したような単純な特徴量ではなく，マルコフ連鎖などを使用した特徴量についても考慮していく．また，機械学習に使用するデータセットが持つ特徴の違いによって，有用な特徴量やその検知結果が変わると考えられるため，データセット間の比較を検討する．

5. 関連研究

文献 [6] は，ANN を用いて，悪意のあるソフトウェアを分類・検出する手法を提案した．この手法では，収集された実行可能ファイルがアンパックして使用されている．特徴量として，API 呼出しに基づく N-gram 特徴量と Windows Portable Executables（PE）が使用されている．N-gram は，長さが n グラムの部分文字列のことであり，長さが n グラムの単語の頻度を把握し，特徴量とされている．この手法では，実行可能ファイルをアンパックして使用しているが，本稿ではシステムコールをベースとした動的解析ログから特徴量を抽出するため，アンパックする必要がない．また，本稿では，API ではなくシステムコールに焦点を当てて特徴量の生成を行った．

文献 [7] は，Android ゲームアプリケーションを実行してその挙動を分析することで，マルウェア検出に有用なシステムコールの存在を示した．良性または悪性の Android ゲームアプリケーションを一定時間実行し，strace コマンドを利用してシステムコールを取得した．その後，各システムコールの出現頻度をカウントすることで，良性・悪性間で相違があるシステムコールを示し，それらのシステムコールを用いることでマルウェア検出に利用可能であることを示している．文献 [7] では，Android のシステムコールを対象としているが，本稿では，Alkanet による解析を行い，Alkanet がトレース対象としているシステムコールを用いた．

6. おわりに

本稿では，システムコールトレーサ Alkanet により取得したシステムコールトレースログを用いて，システムコール列内における各システムコールの出現回数や確率，システムコール間の各遷移回数や確率を特徴量として 3 種類の機械学習によるマルウェア検知を行い，学習アルゴリズム間でのマルウェア検知精度を比較した．また，マルウェアデータとして，マルウェアプロセスのプライマリスレッドのみを対象としてシステムコール列を抽出しており，プロセス生成やコードインジェクションに関しては考慮していない．今後は 4 章で述べた事柄に加え，プロセス生成やコードインジェクションも考慮する場合の検討を行う．

参考文献

[1] S. Naval, V. Laxmi, M. Rajarajan, M. S. Gaur, and M. Conti, “Employing program semantics for Malware detection”, *IEEE Transactions on Information Forensics and Security*, Vol. 10, No. 12, pp. 2591-2604 (2015).

[2] A. Pektaş and T. Acarman, “Malware classification based on API calls and behavior analysis”, *IET Information Security*, Vol. 12, Iss. 2, pp. 107-117 (2017).

[3] 大月勇人, 瀧本栄二, 齋藤彰一, 毛利公一, “マルウェア観測のための仮想計算機モニタを用いたシステムコールトレース手法”, 情報処理学会論文誌, Vol. 55, No. 9, pp. 2034-2046 (2014).

[4] X. Xiao, Z. Wang, Q. Li, S. Xia, and Y. Jiang, “Back-propagation neural network on Markov chains from system call sequences: a new approach for detecting Android malware with system call sequences”, *IET Information Security*, Vol. 11, Iss. 1, pp. 8-15 (2017).

[5] 高田雄太, 寺田真敏, 松木隆宏, 笠間貴弘, 荒木粧子, 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 Datasets 2018～”, 研究報告コンピュータセキュリティ（CSEC）, Vol. 2018-CSEC-82, No. 38, pp. 1-8 (2018).

[6] M. Chowdhury, A. Rahman, and R. Islam, “Protecting data from Malware threats using machine learning technique,” in *Proceeding of the 2017 12th IEEE Conference on Industrial Electronics and Applications (ICIEA’17)*, IEEE CPS, pp. 1691-1694 (2017).

[7] M. Jaiswal, Y. Malik, and F. Jaafar, “Android gaming malware detection using system call analysis,” in *Proceedings of the 2018 6th International Symposium on Digital Forensic and Security (ISDFS’18)*, pp. 1-5 (2018).