4ZD-04

Linuxのシステムコールシーケンスに基づくマルウェアの振る舞いの動的な検出

米田 成海 †　秦野 亮 ‡　西山 裕之 §
東京理科大学 理工学部 †　東京理科大学 理工学部 ‡　東京理科大学 理工学部 §

1 はじめに

デジタルトランスフォーメーションの普及に伴い，マルウェアの脅威は深刻な問題になっている．IBM社によると，クラウド・ワークロードの90%で稼働しているLinux OSに対するマルウェアによる攻撃が増加していることが報告されている[1]．このことから，Linuxマルウェアの検出に対する需要も増加すると言える．一方，マルウェアの検出を行う研究の大半はWindows上で動作するものを検出対象としており，Linux上で動作するマルウェアを対象にする研究は十分にされていない．以上から，本研究はLinux上で動作するマルウェアを動的解析を用いて検出することを目指す．

2 関連研究

大月ら[2]は，Windows XP x86-64環境上で動作するシステムコールトレーサーであるAlkanetを仮想計算機モニタBit Visorを援用して実装した．仮想環境におけるOSの占めるメモリ領域を直接参照し，独自にWindowsのデータ構造の解釈を行うことでシステムコールの情報と呼び出し元のスレッドの情報を取得した．マルウェアが作成したスレッドかどうかの識別は，発行されたシステムコールの引数などから行った．また，効率的にマルウェアの挙動を観測するため，マルウェアが使用するシステムコールに絞って情報を取得した．

本研究ではLinuxのカーネルソースを改造することでシステムコールトレースを実現する．

3 提案手法

3.1 システムコールトレーサー

本研究では，VirtualBox(6.1.36)を用いて仮想環境を構築する．なお，ゲストOSには，Debian Linux(11.6)を用いる．仮想環境上のLinuxカーネル空間とホストOS上のアプリでUDP通信を行い，リアルタイムでシステムコールの情報を取得する．これは，entry/common.cの関数do_syscall_64と，fork.cの関数kernel_cloneに通信用コードを追加して実現する．

また，マルウェアは動的解析されることを防ぐ機能を持っているため，具体的な機能とその対策を説明する．例えば，動的解析では実行されなかった挙動は取得できない．そのため，マルウェアは自身が解析下にあることを検出すると，本来の挙動を隠蔽する．本研究では，情報の取得をすべてカーネル空間上で完結させることで，ユーザ空間のプログラムであるマルウェアが解析下にあることを感知できないようにする．また，マルウェアは悪意あるコードの実行を他のプロセスに行わせることで，解析を困難にする．マルウェアを実行する環境内では，バックグラウンドプロセスなども動作しているため，それらの動作とマルウェアの動作とを混交せずに識別する必要がある．

本研究では，マルウェアの動作のみを取得するため，OSの起動から呼び出された全てのシステムコールのIDとその呼び出し元のPID，

Dynamic Malware Behavior Detection using Linux System Call Sequences
† Narumi Yoneda, Tokyo University of Science
‡ Ryo Hatano, Tokyo University of Science
§ Nishiyama Hiroyuki, Tokyo University of Science

また PID の親子関係も取得する (図 1).

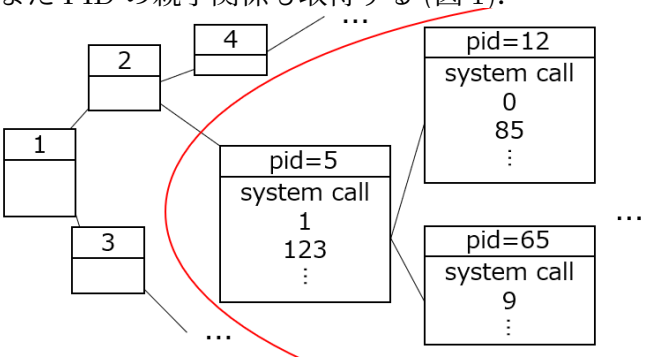

図 1　取得するデータ，マルウェアの pid=5
pid=5 のサブツリーのデータを取得

3.2　機械学習によるマルウェアの検出

手順は以下の通りである．まず，本研究のシステムコールトレーサーを用いてクリーンウェア又はマルウェアを実行し，システムコールの時系列生データを取得する．そして，システムコールの ID ごとに呼び出された回数を集計する．最後に，集計したデータを特徴量とし，LightGBM を用いてマルウェアを検出する．

4　実験及び考察

本研究では，2022 年年末時点での malware bazaar のデータベースからマルウェアを収集した．x86-64 環境上で動作する，ReversingLabs TitaniumCloud の評価が Linux.Trojan となっているものを用いた (elf ファイル 25 件，sh ファイル 25 件)．クリーンウェアは，bash で apt install (package) を実行して取得した (50 件)．パッケージ名は，bash で apt search a を実行し，取得したパッケージ名のリストからランダムに選択した．これらのプログラムに対し，ターゲットとなる PID のデータのみに選別し，システムコールの ID ごとに呼び出された回数を集計した．この集計したデータを特徴量とし，LightGBM を用いてマルウェアの検出を行った．5 分割交差検証を行ったところ，全ての正解率が 100% となった．そこで，LightGBM の特徴量の重要度を算出し，一番重要度が高い特徴量を削除してもう一度学習，正解率を算出するということを繰り返した (図 2).

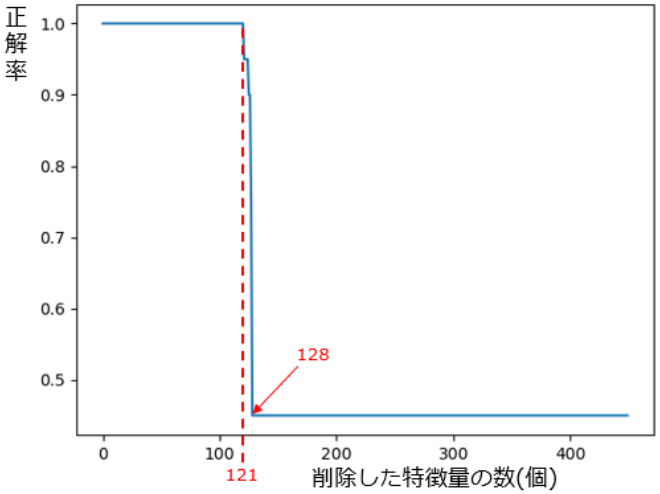

図 2　特徴量削除と再学習した際の正解率の推移

図 2 より，121 個の特徴量を削除するまで，正解率が 100% となった．削除した特徴量のほとんどは，クリーンウェアが 1 回以上呼び出し，マルウェアは 1 度も呼び出していなかった．削除したシステムコールの具体例としては，gettimeofday(時刻を取得/設定) など，マルウェアが使う可能性が低いものが多かった．

本研究の実験は，データの量や生データの部分から頑健な実験であったとは言えない．しかし，全てのシステムコール ID の情報を取得することで，クリーンウェアがよく呼び出し，マルウェアがほとんど呼び出さないシステムコールの情報を取得できるようになった．結果的に検出すべき対象に関して一定の知見が得られたと考えられる．

5　おわりに

現時点では，システムコールを時系列・リアルタイムで取得できる利点が十分に活かせていない．今後は上記の改善を目指す．

参考文献

[1] IBM 社 X-Force
脅威インテリジェンス・インデックス 2022
https://www.ibm.com/reports/threat-intelligence/

[2] 大月 勇人, 仮想化技術に基づいたマルウェア解析のためのシステムコールトレース手法に関する研究, 博士学位論文, 立命館大学大学院, 2016.