4Z-3

SVM を用いたプログラムの特徴に基づく異常検知システムの改良

新垣杏里[†1]　伊波靖[†2]

沖縄工業高等専門学校創造システム工学専攻[†1]　沖縄工業高等専門学校メディア情報工学科[†2]

1. はじめに

インターネットの急速な進展とともに，マルウェアの出現頻度や伝搬速度が深刻な脅威となっている．従来の検知手法であるシグネチャベースのマルウェア対策ソフトウェアでは，0-day 攻撃や亜種ウイルスなどの未知のマルウェアへの対応に限界があるため，その解決手法を探る研究が盛んに行われている．その研究の一つとして，プログラムを実行させ，呼び出されたシステムコールの履歴に基づいて検知する振る舞い検知が挙げられる．しかし，この手法だとプログラムを動作させ検知するため，コンピュータにリスクが発生する危険性があった．そこで，我々はこの振る舞い検知を行う前処理として，SVM（Support Vector Machine）を用いたプログラムの特徴に基づく異常検知システムを WHIPS（Windows Host Intrusion Prevention System）に実装した[1]．予備実験により，Open Test において認識率 98.35%が得られ，未知のデータについても検知が行えることを確認したが，その予備実験に用いた評価データには，マルウェアと挙動が類似しているインストーラーが含まれていなかった．Open Test と同様に WHIPS でインストーラーの認識実験を行った結果，その認識率は 65%であった．

よって，本研究ではインストーラーの認識率の向上を目的として SVM を用いた異常検知システムの改良を行った．

2. SVM(Support Vector Machine)

SVM は統計的学習理論に基づく新しい 2 クラスのパターン認識手法であり，ニューラルネットワークなどの従来法と比較して汎化能力が高い点と最適解が求まる点に特徴があり，学習に用いていないデータに対しても高い識別率を示す．SVM がこのような特徴を示すのは，その学習に認識誤りと汎化性能の両面から最適化が行われ，これが 2 次の凸計画問題として定式化されているため最適解を求める事ができるためである[2].

SVM は学習して求められた分離超平面による線形識別を行っているが，学習用データを線形分離することが不適切な場合，学習データをより高次のパターン空間に非線形写像を行い，高次元空間で分離超平面を構築し線形識別を行う．その結果，元のパターン空間ではできなかった学習用データの線形分離が可能となる．

3. WHIPS

WHIPS は Windows 上で動作する侵入防止システムの一種で，予め ACD に危険なプロセスとシステムコール及び引数をルールとして登録し，ルールベースにより異常検知を行う．現在，オープンソースソフトウェアとして GPL で公開されている[3]．WHIPS の動作概要を図 1 に示す．

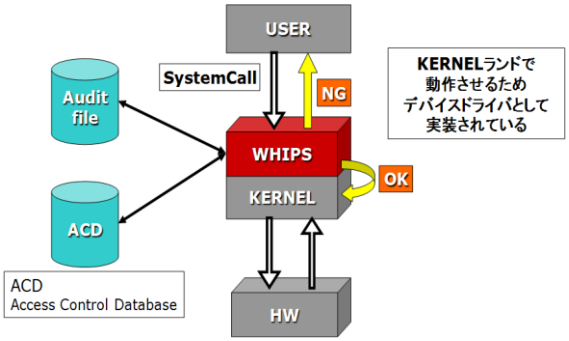

図 1：　WHIPS の動作概要

WHIPS は USER 側のプログラムから要求されるシステムコールをフックすることで，カーネルより先に WHIPS の処理を実行する仕組みとなっている．しかし，WHIPS のルールを記述するには高度な専門知識が必要となるため，一般の方が利用するには難しく，また，未知のマルウェアへの対応が困難だという問題があった．

4. プログラムの特徴に基づく異常検知システム

4.1. プログラムの特徴の定義

本研究では，プログラムの特徴をプログラム名の長さ，使用する DLL と API により定義する．プログラムが各々所持する DLL と API の情報をシステムコール”NtCreateProcessZw”の引数から取得し，その情報から SVM に用いる特徴ベクトルを生成する．具体的にはまず，取得した DLL と API をそれぞれのデータベースから検索し，存在した場合は検出された DLL と API のインデックスを特徴ベクトルの要素番号とし，要素値を 1 にすることで特徴ベクトルを生成する．本システムを WHIPS に実装

Improvement of an anomaly detection system based on the features of program using SVM.
†1 Anri ARAKAKI,
Creative System Engineering Advanced Course, Okinawa National College of Technology
†2 Yasushi IHA
Department of Media Information Engineering, Okinawa National College of Technology

した段階での特徴ベクトルの次元数は，プログラム名の長さと 347 種類の DLL，6222 種類の API の合計 6570 次元である．正常なプログラム 900 個とマルウェア 600 個を用意し，それぞれ二分割した数を SVM の学習データと評価データに充てた．図 2 に特徴ベクトルの生成法を示す．

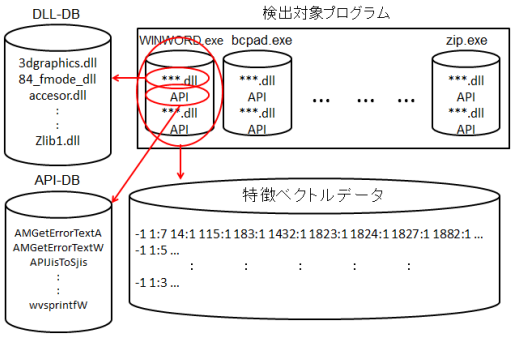

図 2：特徴ベクトルの生成法

4.2. 異常検知システムの概要

本システムはまず，プログラムが実行する際に発行されるシステムコール“NtCreateProcessZw”をフックし，割り込み処理を発生させる．次に，“NtCreateProcessZw”の引数からそのプログラムが利用する API と DLL の情報を取得し，特徴ベクトルを生成する．そして，WHIPS に組み込まれた SVM によってその特徴ベクトルを認識させる．SVM がマルウェアと判断した場合は STATUS_UNSUCCESSFUL が返され，そのプログラムの実行を拒否する． 図 3 に本システムの流れを示す．

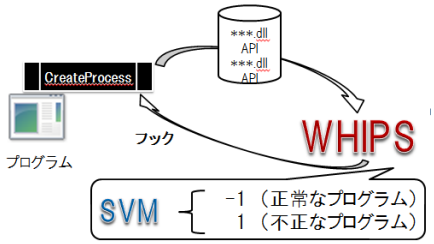

図 3：プログラムの特徴に基づく異常検知の流れ

4.3. 改良点と予備実験

システムの見直しを行った際に， DLL と API を正確に取得できていないことが判明した．そのため，学習データと評価データの更新を行った．現在，特徴ベクトルの次元数は，DLL が 282 種類，API が 5342 種類の合計 5625 次元である．学習データに正常なプログラム 547 個とマルウェアを 423 個，評価データに正常なプログラム 546 個とマルウェア 422 個を用いた． libsvm を使用した予備実験において，学習データを用いた Closed Test では 100%，評価データでの Open Test では 97.70%であった．表 1 に改良版の予備実験結果を示す．

また，インストーラーの学習データ及び評価データをそれぞれ 100 個用意し，インストーラーの認識実験を行った．その結果は 49%であった．このような結果になった原因として，インストーラーとマルウェアの類似性が挙げられる．これら 2 つのプログラムは，新たなファイルの作成やレジストリへのアクセス，システムファイルの上書き等の処理が他のプログラムより比較的多い．そのため，特徴ベクトルが類似しインストーラーの認識率が低い値となったと考えられる．インストーラー追加後の認識率は 90%と上昇し，インストーラーを学習させることは有効であることが示せた．表 2 にインストーラーの予備実験結果を示す．表 2 において，追加前の Open Test のデータにはインストーラーが含まれている．

表 1：改良版の予備実験結果

	認識率
Closed Test	100%（870/870）
Open Test	97.70%（848/868）

表 2：インストーラーの予備実験結果

	Closed Test	Open Test	Installer
追加前	100%(870/870)	92.67%(897/968)	49%(49/100)
追加後	100%(970/970)	95.97%(929/968)	90%(90/100)

5. まとめと今後の課題

本研究では，プログラムの特徴に基づく異常検知システムに，インストーラーの認識率向上を目的として改良を 2 点施した．実験に用いる学習データと評価データを更新し，予備実験により Open Test で 97.70%と，以前と大差ない認識率が得られた．また，インストーラーの特徴ベクトルを学習データに追加することで，追加前に 49%だった認識率が追加後 90%となり 41%改善された．よって，インストーラーを学習させることは有効であることが示せた．

今後の課題として，WHIPS におけるインストーラーの認識実験と，リアルタイムで WHIPS 上の SVM が学習する学習モードの追加，そして振る舞い検知との組み合わせが挙げられる．その後評価実験によりシステムの性能を評価し，本システムの有効性を確かめる．

謝辞

本研究は科研費 23500106 の助成を受けたものである．

参考文献

[1] 伊波靖，新垣杏里：SVM を用いたプログラムの特徴に基づく異常検知システムの実装，情報処理学会第 75 回全国大会講演論文集，pp.513-515 (2013).

[2] Cristianini, N. and Shawe-Taylor, J.: サポートベクターマシン入門，共立出版(2005).

[3] Battistoni, R, Gabrielli, E, and Mancini, L. V.: A Host Intrusion Prevention System for Windows Operating Systems, Proceedings of the 9th European Symposium On Research in Computer Security (ESORICS 2004), pp. 352-368(2004).