2010 Second International Conference on Advances in Computing, Control, and Telecommunication Technologies

ANALYSIS OF MACHINE LEARNING TECHNIQUES USED IN BEHAVIOR-BASED MALWARE DETECTION

Ivan Firdausi, Charles Lim, Alva Erwin
Department of Information Technology
Swiss German University
Tangerang, Indonesia
ivan.firdausi@student.sgu.ac.id
charles.lim@sgu.ac.id
alva.erwin@sgu.ac.id

Anto Satriyo Nugroho
Center for Information & Communication Technology,
Agency for the Assessment & Application of Technology
(BPPT)
Jakarta, Indonesia
asnugroho@ieee.org

Abstract—**The increase of malware that are exploiting the Internet daily has become a serious threat. The manual heuristic inspection of malware analysis is no longer considered effective and efficient compared against the high spreading rate of malware. Hence, automated behavior-based malware detection using machine learning techniques is considered a profound solution. The behavior of each malware on an emulated (sandbox) environment will be automatically analyzed and will generate behavior reports. These reports will be preprocessed into sparse vector models for further machine learning (classification). The classifiers used in this research are *k*-Nearest Neighbors (*k*NN), Naïve Bayes, J48 Decision Tree, Support Vector Machine (SVM), and Multilayer Perceptron Neural Network (MLP). Based on the analysis of the tests and experimental results of all the 5 classifiers, the overall best performance was achieved by J48 decision tree with a recall of 95.9%, a false positive rate of 2.4%, a precision of 97.3%, and an accuracy of 96.8%. In summary, it can be concluded that a proof-of-concept based on automatic behavior-based malware analysis and the use of machine learning techniques could detect malware quite effectively and efficiently.**

Keywords—**malware analysis, dynamic analysis, behavior analysis, data mining, machine learning, classification, malware detection**

I. INTRODUCTION

The problem to be examined involves the high spreading rate of computer malware (viruses, worms, Trojan horses, rootkits, botnets, backdoors, and other malicious software) and conventional signature matching-based antivirus systems fail to detect polymorphic and new, previously unseen malicious executables. Malware are spreading all over the world through the Internet and are increasing day by day, thus becoming a serious threat. The manual heuristic inspection of static malware analysis is no longer considered effective and efficient compared against the high spreading rate of malware.

Nevertheless, researches are trying to develop various alternative approaches in combating and detecting malware. One proposed approach (solution) is by using automatic dynamic (behavior) malware analysis combined with data mining tasks, such as, machine learning (classification) techniques to achieve effectiveness and efficiency in detecting malware.

II. RELATED WORKS

Trinius et al. [2] introduced a new representation for monitored behavior of malicious software called Malware Instruction Set (MIST). The representation is optimized for effective and efficient analysis of behavior using data mining and machine learning techniques. It can be obtained automatically during analysis of malware with a behavior monitoring tool or by converting existing behavior reports.

Rieck et al. [3] aim to exploit specific shared patterns for classification of malware. The authors said that variants of malware families share typical behavioral patterns reflecting its origin and purpose. Their method proceeds in three stages: (a) behavior of collected malware is monitored in a sandbox environment, (b) based on a corpus of malware labeled by an anti-virus scanner a malware behavior classifier is trained using learning techniques and (c) discriminative features of the behavior models are ranked for explanation of classification decisions.

Rieck et al. [4] propose a framework for automatic analysis of malware behavior using machine learning. The framework allows for automatically identifying novel classes of malware with similar behavior (clustering) and assigning unknown malware to these discovered classes (classification).

Christodorescu et al. [5] propose a technique by comparing the execution behavior of a known malware against the execution behaviors of a set of benign programs. The authors mine the malicious behavior present in a known malware that is not present in a set of benign programs. The output of the authors' algorithm can be used by malware detectors to detect malware variants.

III. METHODOLOGY

The research methodology process will be explained in this section. The general overview of the research methodology is shown in Fig. 1.

A. Data Acquisition and Storage

The data set consists of malware data set and benign instance data set. Both malware and benign instance data sets are in the format of Windows Portable Executable (PE) file binaries. A total of 220 unique malware (specifically

978-0-7695-4269-0/10 $26.00 © 2010 IEEE
DOI 10.1109/ACT.2010.33

Indonesian malware) samples were acquired. The benign instance data set samples were collected from system files located in the "System32" directory of a clean installation of Windows XP Professional 32-bit with Service Pack 2. A total of 250 unique benign software samples were acquired.

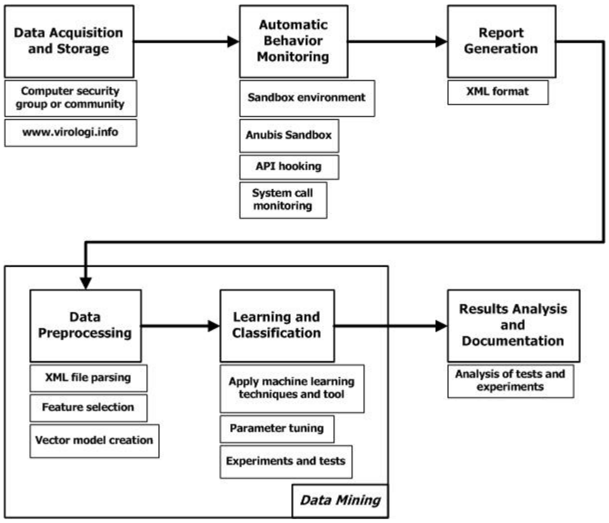

Figure 1. General overview of the research methodology.

B. *Automatic Behavior Monitoring and Report Generation*

The next step is conducting dynamic analysis (behavior monitoring) of both the malware and benign instance data sets. This process is done by submitting each and every sample to a free-online automatic dynamic analysis service: Anubis [6]. Binary submission and execution of Anubis result in the generation of a report file. In this research, all the generated report files were downloaded in XML format.

C. *Data Preprocessing*

The next step is conducting data preprocessing. The data preprocessing steps of this research are described as follows:

1. All the XML report files were parsed to select the most relevant and important attribute values (feature selection).
2. A term dictionary was created, which contains all the attribute values that were previously parsed and selected.
3. Each XML report file was compared against the term dictionary by counting the existence (or non-existence) of each term word in the term dictionary based on binary weight and term frequency weight.
4. Sparse vector models were created for each XML report file and Attribute-Relation File Format (ARFF) files were created.

D. *Learning and Classification*

The next step is to conduct learning and classification based on the ARFF files. Machine learning techniques were applied for the learning and classification of the ARFF files.

IV. Tests And Experimental Results

The tests and experiments were conducted using Weka [1] 3.6.2 for Windows OS version. These tests and experiments were conducted based upon four data sets:

1. Binary-weight vector model without feature selection.
2. Term frequency-weight vector model without feature selection.
3. Binary-weight vector model with feature selection.
4. Term frequency-weight vector model with feature selection.

Each data set was applied to 5 different classifier algorithms, which were *k*-Nearest Neighbor, Naïve Bayes, Support Vector Machine (SVM), J48 decision tree, and Multilayer Perceptron (MLP) neural network.

A. *Performance Metrics*

This research statistically measured the performance of the binary classification (malicious or benign) tests that were conducted. The statistical measures include true positive rate (sensitivity, recall, hit rate), false positive rate (fall-out), positive predictive value (precision), and accuracy.

B. *Without Feature Selection*

From Tab. I, Fig. 2, Tab. II, and Fig. 3, performance results were best achieved by J48 on both binary-weight and term frequency-weight data sets, although there were slightly different performance between *k*NN, SVM, and J48. In addition, Naïve Bayes achieved the poorest performance on both binary-weight and term frequency-weight data sets.

TABLE I. PERFORMANCE METRICS RESULTS (BINARY, NO FEATURE SELECTION)

Classifier	TPR	FPR	PPV	ACC
*k*NN	81.7%	8.1%	91.8%	86.5%
Naïve Bayes	58.1%	12.8%	93.2%	65.4%
SVM	90.4%	8.4%	90.4%	91.0%
J48	90.9%	3.8%	95.9%	93.6%

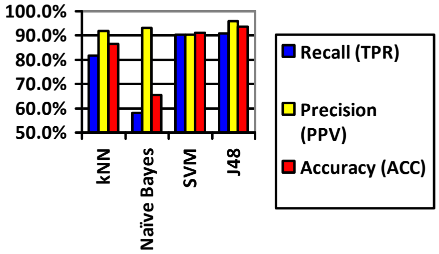

Figure 2. Classifier performance comparison (binary, no feature selection).

TABLE II. PERFORMANCE METRICS RESULTS (TERM FREQUENCY, NO FEATURE SELECTION)

Classifier	TPR	FPR	PPV	ACC
*k*NN	86.8%	8.8%	90.4%	89.1%
Naïve Bayes	56.8%	22.2%	86.3%	62.8%
SVM	90.5%	7.3%	91.8%	91.7%
J48	95.9%	2.4%	97.3%	96.8%

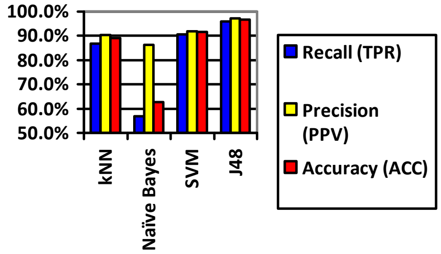

Figure 3. Classifier performance comparison (tf, no feature selection).

C. *With Feature Selection*

Based on the tests and experiments conducted for feature selection using Correlation-based Feature Selection (CFS) Subset Evaluator and *Best First* search algorithm, the feature selection results are as follows:

1. For the binary-weighted data sets, the attributes were reduced from 5191 attributes to 116 attributes (a reduction of 97.7%).
2. For the term frequency-weighted data sets, the attributes were reduced from 5191 attributes to 11 attributes (a reduction of 99.7%).

TABLE III. PERFORMANCE METRICS RESULTS (BINARY, FEATURE SELECTION)

Classifier	TPR	FPR	PPV	ACC
*k*NN	94.3%	8.1%	90.4%	92.9%
Naïve Bayes	94.2%	9.2%	89.0%	92.3%
SVM	94.3%	8.1%	90.4%	92.9%
J48	94.2%	9.2%	89.0%	92.3%
MLP	94.0%	11.2%	86.3%	91.0%

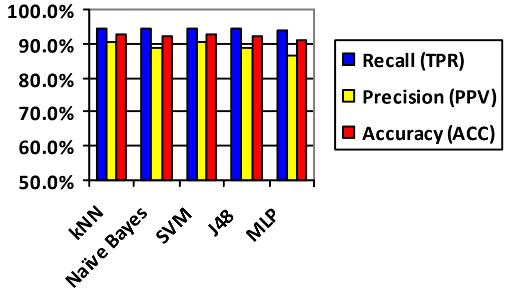

Figure 4. Classifier performance comparison (binary, feature selection).

TABLE IV. PERFORMANCE METRICS RESULTS (TERM FREQUENCY, FEATURE SELECTION)

Classifier	TPR	FPR	PPV	ACC
*k*NN	94.3%	8.1%	90.4%	92.9%
Naïve Bayes	58.7%	19.1%	87.7%	65.4%
SVM	94.1%	10.2%	87.7%	91.7%
J48	94.5%	4.8%	94.5%	94.9%
MLP	94.4%	6.0%	93.2%	94.2%

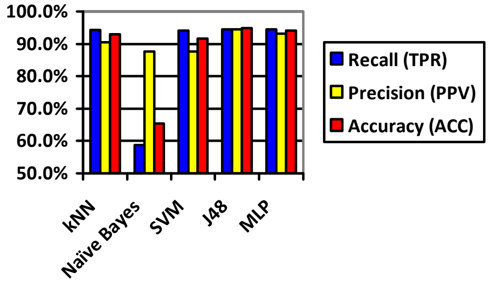

Figure 5. Classifier performance comparison (tf, feature selection).

From Tab. III, Fig. 4, Tab. IV, and Fig. 5, performance results were best achieved also by J48 on both binary-weight and term frequency-weight data sets, although there were also slightly different performance between *k*NN, SVM, J48, and MLP. Furthermore, although the attributes (features) were reduced, good performance results could still be achieved by performing feature selection.

V. Conclusion

In conclusion, it can be stated that this research has developed a proof-of-concept of an alternative malware detection method.

Feature selection was presented in this research using *Best First* search algorithm. By performing feature selection or feature reduction, the features were reduced drastically. Hence, the time taken to train and build the model becomes shorter at the cost of the performance decreases slightly. In some cases, the performance can also increase slightly.

The performance comparison of 5 different classifiers was also presented. The overall best performance was achieved by J48 using the term frequency-weight without feature selection data set, with a recall (true positive rate) of 95.9%, a false positive rate of 2.4%, a precision (positive predictive value) of 97.3%, and an accuracy of 96.8%. The analysis of the tests and experimental results concluded that this proof-of-concept is quite effective and efficient in detecting malware.

References

[1] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten, “The WEKA Data Mining Software: An Update”, SIGKDD Explorations, Volume 11, Issue 1, 2009.

[2] P. Trinius, C. Willems, T. Holz, and K. Rieck, “A Malware Instruction Set for Behavior-Based Analysis”, 2009.

[3] K. Rieck, T. Holz, C. Willems, P. Duessel, and P. Laskov, “Learning and Classification of Malware Behavior”, DIMVA, LNCS 5137, pp. 108–125, Berlin Heidelberg: Springer-Verlag, 2008.

[4] K. Rieck, P. Trinius, C. Willems, and T. Holz, “Automatic Analysis of Malware Behavior using Machine Learning”, 2009.

[5] M. Christodorescu, S. Jha, and C. Kruegel, “Mining Specifications of Malicious Behavior”, Proceedings of the 6th joint meeting of the ESEC and the ACM SIGSOFT Symposium on the FSE, September 3–7, Dubrovnik, Croatia, ACM, 2007.

[6] U. Bayer, C. Kruegel, and E. Kirda, “TTAnalyze: A Tool for Analyzing Malware”, 15th Annual Conference of the European Institute for Computer Antivirus Research, Hamburg, Germany, pp. 180–192, 2006.