International Journal of Engineering & Technology, 7 (3.20) (2018) 502-506

SPC

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Malware Analysis Using Apis Pattern Mining

Nawfal Turki Obeis and Wesam Bhaya

University of Babylon, College of Information Technology, Babil, Iraq.
**E-mail:nawfalaljumaili@yahoo.com, wesambhaya@itnet.uobabylon.edu.iq.*

Abstract

Malicious code threats cybersecurity. Malware and its detection have caught the challenges of both anti-malware industry and researchers for decades.

We use pattern mining technique to find the frequent Windows Application Program Interface (API) calls and then uses the frequent item sets to build the sequence of features for next analysis. Shingling techniques have proven effective for the problem of detecting. For verification, we use clustering processes of malware sequences based on their frequent API call sequences.

We have achieved a high detection rate of 99.029% with accuracy as high as 98.8%. Thus, proposal method improved state of the art technology in several aspects: accuracy, detection rate, and false alarm rate were decreased.

The experiment upon a big API sequence dataset demonstrated that the using frequent of API call sequences could realize a high accuracy for malware clustering while dropping the computation time.

Keywords: *Malicious Code; Malware Detection; Shingling; API Calls; Pattern Mining.*

1. Introduction

Malware program alludes to plans that purposefully mishandle vulnerabilities in preparing systems for a ruinous reason. Malware program can be isolated if the item needs or does not require a host framework to work. Another technique for classifying Malware program is by perceiving if the item creates copies of itself or not [1, 2].

Malware program creators routinely use diverse strategies to alter or change existing malware into new polymorphic adjustments to evade detection. The openness of innovative toolboxes has made it less requesting for malware creators to use procedures, for instance, dead-code expansion and enlist reassignment to play out this change. The malicious program change or jamming can be classified into transformative nature and polymorphism [3, 4].

A malware detector system is a PC program that endeavors to distinguish and identify malware using an assortment of techniques that join recognizing malware signature, utilizing heuristic standards, and perceiving malware conduct or exercises. Malware locators can work locally on the system that is being secured or give protection remotely through a PC network [5, 2].

There are two types of data are required by malware detector systems, specifically, information of the malware behavior or signature which can be expanded through a learning methodology and the framework under evaluation. Once the two wellsprings of data get the chance to be available, the malware detector uses its detection techniques to determine whether the product is benign or malware [6].

A Software program is a set of APIs. Define a k-shingle for a software program to be any subset of APIs of length k found within the software program.

2. Related Works

This segment surveys a some of the current algorithms and techniques that are utilized for detecting the malware.

Fan, Ye, and Chen (2016) provide an effective sequence mining method Called "All-Nearest-Neighbor (ANN)" to recognize the malwares in light of the found sequences. The principle aims of this article were to separate the all-around spoke to highlights from Portable Executable (PE) records, and to recognize the malwares with the of ANN technique.[7]

Fan, Hsiao, Chou, and Tseng (2015) provide tracing and analyzing the malware by distinguishing the malicious and malicious programs with the assistance of "Application Programmable Interfaces" (API) calls. The researchers chose some classification procedures, for example, Bayesian, decision tree and Support Vector Machine for malware classification.[8]

Guo and et al.(2014) prescribed a system behavior classification model to detect the portable malwares in view of its behavior characteristics. This work incorporates two phases, which incorporates analyzer training and network behavior detection.[9]

Demme and et al. (2013) provided a malware detector with identify the minor varieties in a malware programs. In this work, the fine-grained run time information was gathered without backing off the applications. The subversion of the insurance plot was avoided with the safe updating of Anti-Virus algorithms.[10]

3. Types of Malicious Detection Systems

Behavior /Statistical Detection: Behavior based identification; the behavior of normal data has been placed in the library. If there is any activity, which isn't happened previously, at that point the report of that activity is sent to the framework system. Behavior-based discovery framework can recognize the attacks, which are now dark through statistical analysis. It is furthermore called behavior based recognition strategy as it perceives the typical and irregular behavior of client [11,12].

Signature Detection: signature-based identification method, the patterns of the unusual activity are saved in database. Marks term is allude to the pattern of these unusual activities. In some time, it is called misuse-based detection. The drawback of this technique is it perceives the referred to attacks in a manner of speaking. It ideal position of this framework is it supports the fast recognizable proof and low rate of false alert [11, 12].

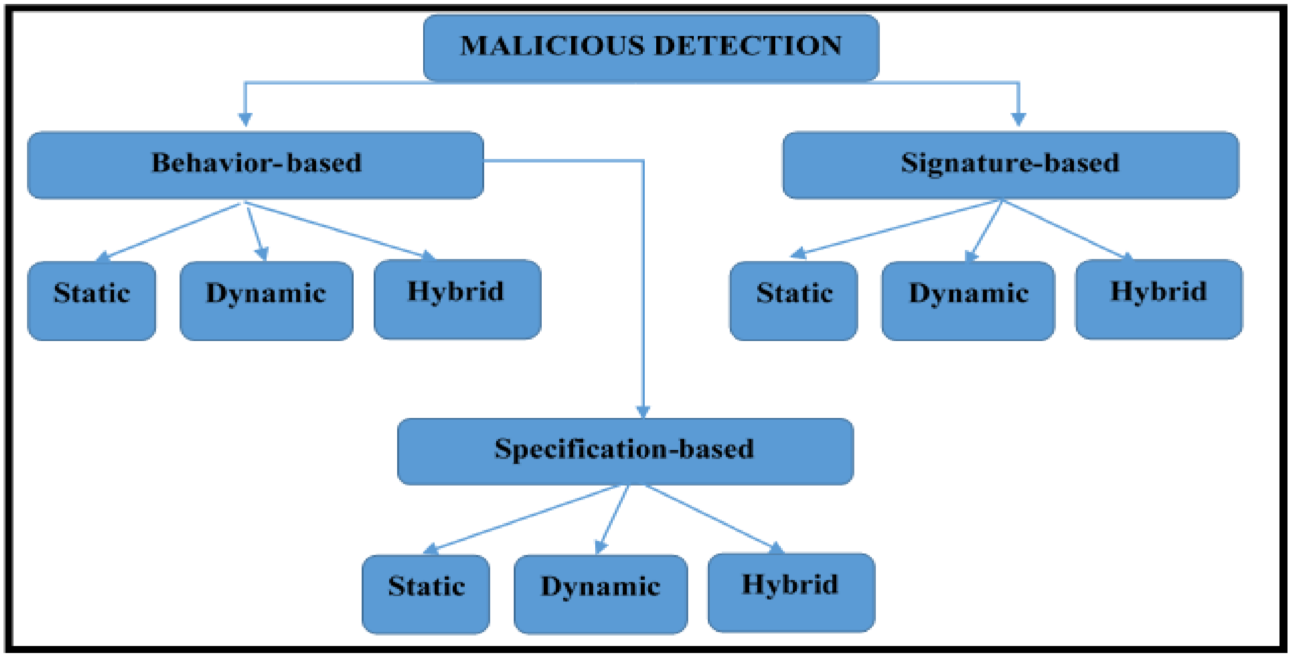

Figure (1): Malicious Detection Organization [2]

Table (1): Comparison between signature detection and behavior detection

Techniques	Advantages	Disadvantage
Signature-based detection	• Higher Accuracy for known behaviors and location rate. • Simplest and powerful strategy. • Very Low false alarm rate.	• Can recognized just known malware. • regular update is very important of the standards, which are utilized.
Behavior-based detection	• Can test unknown and more complicated malicious • Detect new and unforeseen vulnerabilities.	• Need to be trained and tuned mod carefully, otherwise and tend to false-positives. • High false alarm rate and Low detection rate.

4. Some Types of Malicious

In the following, the portion of the key classifications of malicious are briefly surveyed [13,14]:

- *Spyware*: is any innovation that guides in social event data around a man or organization without their insight.
- *Virus*: is a project or programming code that reproduces by being replicated or starting its duplicating to another system, PC boot sector or archive.
- *Worm*: is a self-recreating virus that does not change records but rather copies itself.
- *Logic bomb*: is modifying code, embedded surreptitiously or purposefully, that is intended to execute (or "detonate") under conditions, for example, the omission of a specific measure of time or the disappointment of a project client to react to a system charge.
- *Trapdoor*: is a technique for accessing some a part in a framework other than by the typical strategy (e.g. obtaining entrance without supplying a watchword).
- *Trojan horse*: is a system in which malignant or destructive code is contained inside obviously innocuous programming or information in a manner that it can gain power and do its select type of harm.
- *RATs (Remote Admin Trojans)*: are an uncommon type of Trojan Horse that permits remote control over a machine.

- *Malware*: short to "**mal**icious soft**ware**" is any system or document that is unsafe to a PC client.
- *Mobile Malicious Code*: web archives frequently have server-supplied code connected with them that executes inside the web browser.
- *Malicious Font*: site page text that endeavors the default strategy used to de-compress Embedded in Windows

(Open Type Fonts) based projects including Internet Explorer and Outlook.

- *Rootkits*: are an arrangement of programming instruments utilized by a gatecrasher to pick up and keep up access to a PC framework without the client's knowledge.

5. Proposed Methodology

This section shows the point-by-point depiction of the proposed malware detection system. The fundamental goal of this work is to distinguish the malwares that happened in API calls, and to group its write, as normal or abnormal. The majority of the customary strategies don't center to recognize the malwares in the windows executable, so this work for the most part centered on the malware detection in an API call pattern. Here, the oddity is actualized in the feature extraction and feature selection stages. Figure (2) shows the general of the proposed system, which incorporates the following stages:

- Transformation
- Remove redundancy
- Features extraction
- Features selection
- Pattern Matcher
- Cluster Module

•

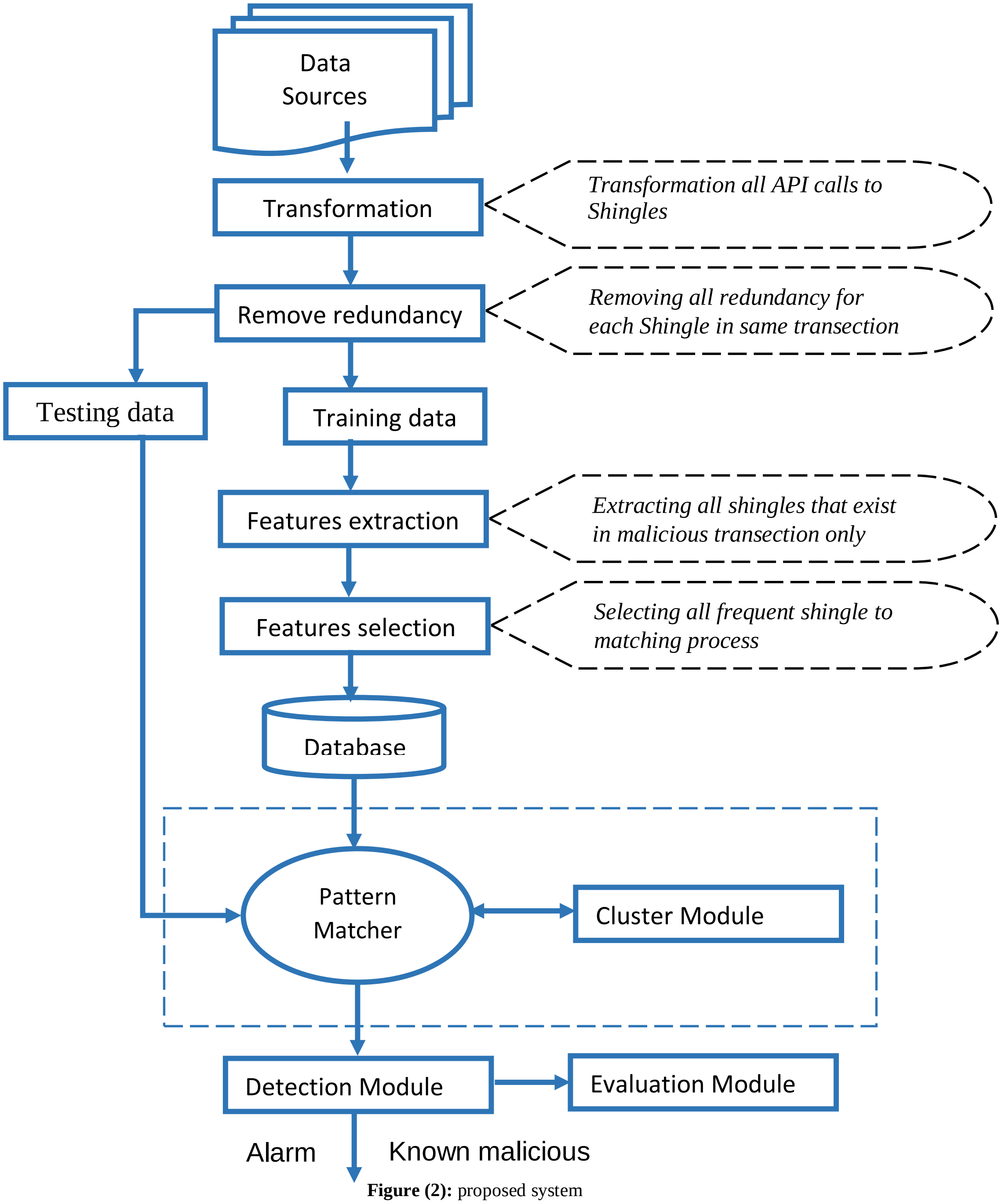
Data Sources
Transformation
Transformation all API calls to Shingles
Remove redundancy
Removing all redundancy for each Shingle in same transection
Testing data
Training data
Features extraction
Extracting all shingles that exist in malicious transection only
Features selection
Selecting all frequent shingle to matching process
Database
Pattern Matcher
Cluster Module
Detection Module
Evaluation Module
Alarm
Known malicious

Figure (2): proposed system

6. Evaluation Measurement of Malicious Detection System

There are some essential variables which are utilized to evaluate measurement of malicious detection system [15,14].

True positive (TP): The aggregate number of typical data that are recognized as an ordinary data amid malicious detection process.

True negative (TN): In malicious recognition, number of identified anomalous data which are really irregular data in dataset.

False positive (FP): false alarm, all out number of recognized ordinary data yet they are real malicious.

False negative (FN): Number of distinguished strange cases however, in genuine they are normal data.

Evaluation measurement of malicious detection system is measured in terms of detection rate, accuracy and false alarm rate:

Detection Rate (DR) = (TP/TP+FN) x 100%.

False Alarm Rate (FAR) =FP/Number of Malicious.

Accuracy = (TP+TN/TP + TN + FP + FN) x 100%.

7. Cluster Analysis and Experimental Results

A Dynamic malware detection system is implemented in two experiments: the first experiment implements CSDMC2010 [16] using modified k-means cluster analysis with 70% for training and 30% for testing. The second experiment implements APIMDS [17] dataset using modified k-means cluster analysis without training and used all dataset for testing.

7.1. The Modified K-Means Cluster Analysis for Csdmc2010 Dataset

Picking the right values of (k) shingle in k-means algorithm can measure the quality of clustering process. In table (2), we start by implement the modified k-means algorithm for k=2, 4, and 6. The following results show the performance evaluation for cluster quality during the training stage using the equations in section (6).

Table (2): Choosing the Right (K- shingle) Values for CSDMC2010

K- shingle	**ACC**	**DR**	**FPR**
2	***98.8%***	***99.029%***	***0.081%***
3	91.25%	96.17%	3.83%
4	85%	88%	12%
5	75.94%	75.66%	24.44%
6	70.6%	74.1%	25.9%

In table (2 the best (K- shingle) values are six the highest accuracy with the regarding of higher detection rate and the smallest alarm rate was conducted. The case (1) when k=2 gives a higher accuracy, higher detection rate and zero alarm rate.

7.2. The Modified K-Means Cluster Analysis for Apimds Dataset

Picking the right values of (k) shingle in k-means algorithm can measure the quality of clustering process. In table (3), we start by implement the modified k-means algorithm for k=2, 4, and 6. The following results show the performance evaluation for cluster quality during the training stage using the equations in section (6).

Table (3): Choosing the Right (K- shingle) Values for APIMDS

K- shingle	**ACC**	**DR**	**FPR**
2	***99%***	***99.09%***	***0.01%***
3	83.45%	76.37%	23.63%
4	62%	68%	32%
5	55.22%	55.33%	44.77%
6	50.8%	44.1%	55.9%

7.3. Confusion Matrix Results

When is training 70% of the dataset CSDMC2010 (272) transection and 30% testing (116) transection from all dataset (388) transection and we have results shows in table (4).

Table (4:) Confusion Matrix Results for the dataset CSDMC2010

Actual	**Predicted**	
	Malware	**Normal**
Malware	**TP(111)**	**FP(1)**
Normal	**FN(0)**	**TN(12)**

When is testing 100% of the dataset APIMDS (٢٣١٤٦) transection and we have results shows in table (5).

Table (5): Confusion Matrix Results for the dataset APIMDS

Actual	**Predicted**	
	Malware	**Normal**
Malware	**TP(٢٢٩٣٥)**	**FP(٢١1)**
Normal	**FN(0)**	**TN(٠)**

8. Results Comparison

It is clearly shown that the best classifier is the modified K-means algorithm. The detection system using modified K-means algorithm can be used in an early detection of malware. Generally, the time complexity based on shingling is less than one minute (< 1 minute) for the APIMDS dataset and it is less than (< 5 seconds). It can be applied in an early detection. Table (3.5) presents the overall performance based on accuracy, detection rate, false alarm rate, for each detection model.

Table (3.5): Comparison between a Proposed Detection Model and Other Models

Paper	**ACC**	**DR**	**FPR**
Fan, Ye, and Chen (2016) [7]	95.25%	96.17%	3.83%
Fan, Hsiao, Chou, and Tseng (2015) [8]	95%	96%	4%
Guo and et al.(2014) [9]	85.94%	80.66%	19.44%

Demme and et al. (2013) [10]	96.6%	84.1%	15.9%
Work of the CSDMC2010 dataset	***98.8%***	***99.029%***	***0.081%***
Work of the APIMDS dataset	***99%***	***99.09%***	***0.01%***

9. Conclusion

This paper proposed a significance detection system for malware using an API call pattern. The fundamental point of this paper is to increase the accuracy of malware recognition. For this reason, diverse strategies are actualized in this work. At first; it transforms all API calls to Shingles. Then, it has been removing all redundancy for each Shingle in the same transactions. Furthermore, it has been extracting all shingles that exist in the malicious transaction only to classify the malware. The significant advantages of this strategy are reducing the time consumption, computational complexity and increasing the detection rate.

References

[1] Assif Assad, A., & Deep, K. (2016). Applications of Harmony Search Algorithm in Data Mining: A Survey (pp. 863–874). Springer, Singapore.

[2] Elhadi, A. A., Maarof, M. A., & Barry, B. (2013). IMPROVING THE DETECTION OF MALWARE BEHAVIOUR USING SIMPLIFIED DATA DEPENDENT API CALL GRAPH. International Journal of Security and Its Applications, 7(5), 29–42.

[3] You, I., & Yim, K. (2010). Malware Obfuscation Techniques: A Brief Survey. In 2010 International Conference on Broadband, Wireless Computing, Communication and Applications (pp. 297–300). IEEE.

[4] Christodorescu, M., Jha, S., Maughan, D., Song, D., & Wang, C. (Eds.). (2007). Malware Detection (Vol. 27). Boston, MA: Springer US.

[5] Ravi, C., & Manoharan, R. (2012). Malware Detection using Windows API Sequence and Machine Learning. International Journal of Computer Applications, 43(17), 12–16.

[6] Bhaya, W., & Ali, M. (2017). REVIEW ON MALWARE AND MALWARE DETECTION USING DATA MINING TECHNIQUES. Journal of University of Babylon, 25(5), 1585 - 1601.

[7] Fan, Y., Ye, Y., & Chen, L. (2016). MALICIOUS SEQUENTIAL PATTERN MINING FOR AUTOMATIC MALWARE DETECTION. EXPERT SYSTEMS WITH APPLICATIONS, 52, 16–25.

[8] Fan, C.-I., Hsiao, H.-W., Chou, C.-H., & Tseng, Y.-F. (2015). MALWARE DETECTION SYSTEMS BASED ON API LOG DATA MINING. Paper presented at the Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual, Taichung, Taiwan.

[9] Guo,D.-F., Sui, A.-F., Shi, Y.-J.,Hu, J.-J., Lin,G.-Z.,&Guo, T. (2014). BEHAVIOR CLASSIFICATION BASED SELF-LEARNING MOBILE MALWARE DETECTION. Journal of Computers, 9(4), 851–858.

[10] Demme, J., Maycock, M., Schmitz, J., Tang, A., Waksman, A., Sethumadhavan, S., … Stolfo, S. (2013). On the feasibility of online malware detection with performance counters. In Proceedings of the 40th Annual International Symposium on Computer Architecture - ISCA ’13 (Vol. 41, pp. 559–570). New York, New York, USA: ACM Press.

[11] Shah, K., & Singh, D. K. (2015). A survey on data mining approaches for dynamic analysis of malwares. In 2015 International Conference on Green Computing and Internet of Things (ICGCIoT) (pp. 495–499). IEEE.

[12] Egele, M., Scholte, T., Kirda, E., & Kruegel, C. (2012). A survey on automated dynamic malware-analysis techniques and tools. ACM Computing Surveys, 44(2), 1–42.

[13] Wressnegger, C., Yamaguchi, F., Arp, D., & Rieck, K. (2016). Comprehensive Analysis and Detection of Flash-Based Malware (pp. 101–121). Springer, Cham.

[14] Obeis, N. T., & Bhaya, W. (2016). Review of data mining techniques for malicious detection. Research Journal of Applied Sciences, 11(10).

[15] Miller, B., Kantchelian, A., Tschantz, M. C., Afroz, S., Bachwani, R., Faizullabhoy, R., … Tygar, J. D. (2016). Reviewer Integration and Performance Measurement for Malware Detection (pp. 122–141). Springer, Cham.

[16] DATASET-1: http://www.csmining.org/index.php/malicious-software-datasets-.html

[17] DATASET-2: http://ocslab.hksecurity.net/apimds-dataset