International Journal For Technological Research In Engineering
Volume 10, Issue 2, October-2022 *ISSN (Online): 2347 - 4718*

API CALL BASED MALWARE DETECTION FRAMEWORKS USING SUPERVISED LEARNING MODEL

[1]Jyoti Kumari, [2]Mr. Mukesh Kumar
[1]Research Schalor, [2]Professor
Rabindranath Tagore University

Abstract: Malware is a severe threat from last few decades. It harms the computing system and sensitive data available with the system. Many solutions have been suggested by researchers and industries to deal with these ever going threat. Despite all these solutions, the malware attacks cannot be ignored because of the evolution of zero-day attacks. Therefore, it still stands as a challenging job to handle malware outbreaks, despite many good approaches in refining the efficiency of techniques of malicious transmission recognition, investigation and updation of signature databases and detection policies. There is a need of a consistent heuristic detection model to deal with zero-day attacks which may not be handled by signature based methods.

INTRODUCTION

Malware stands for malevolent software which contains spiteful programs that damages a system in terms of data, behavior and performance. The effect of malware may also spread across the connected networks to infiltrate the connected computing machines to carry out illicit activities. The term malware can be used to denote a single malicious program (singular) or can be used to denote a bunch of malicious programs (plural). There are different forms of malware available in today‟s information industry. These forms include; virus, worm, trojan, bot, rootkit, adware, spyware and ransomware. They differ from each other by their behavior and attacking approach.

MOTIVATION

The most important motivation towards this research is the uncontrolled growth of malware attacks in the information industry. The extensive use of internet and advanced ubiquitous computational devices make it easier for the attackers to perform attacks. Despite development of so many anti-malware systems, the number of attacks is increasing every year. Malware are designed by technically intelligent people for unethical purposes. The creators use numerous obfuscation practices to deviate from detection engines. It stands as a challenge for the malware analysts and researchers to focus on improving detection engines with an open-eye on the obfuscation techniques.

According to the statistics by AV-test (AV-test Report, 2020), they come across 3.5 lakhs of malicious programs every day. The growth rate of malware for the last ten years is shown in figure 1. This denotes the total number of malwares (in millions) identified by AV-test organization in last 10 years. These malware include known as well as unknown malware types.

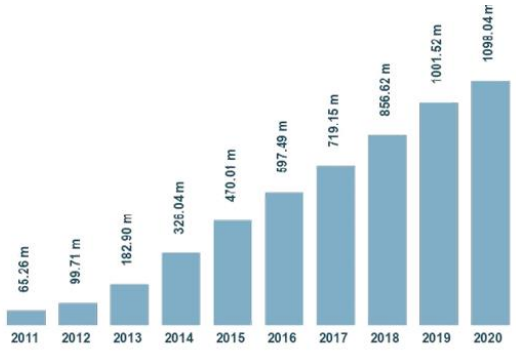

Figure 1: Statistics on Growth of malware by Av-Test (Src: (AV-test Report, 2020)

Malware Detection
These techniques are used to identify the malicious software and stop them from infecting the system, securing the system from possible data loss and system negotiation. These techniques are classified into signature-based detection, heuristic-based detection, specification-based and data mining based detection as shown in figure 4. Each of these detection approaches may use one of the three analysis methods: static, dynamic and hybrid.

Signature based Detection
It is also called as Misuse detection. It stores the database of signature and detects malware by matching pattern alongside the stored DB. The features are generated by observing the dissected code of binary executable and other samples. Then, these features are considered to create the signature of the sample files. A signature library of well-known code is maintained and renewed continuously by the anti-malware software sellers to keep the detection system accurate for known malware families. The key advantages of this method are that it can detect well-known malicious families effectively, uses fewer resources in the detection process and it essentially focuses on signature of malwares. The main weakness is that it may not detect the unseen malcode types because of the absence of signature for these types of malcode.

Heuristic based Detection
It is sometimes known as behavior-based or anomaly-based detection technique. The key purpose is to examine the behavior of seen or unseen malicious families. Behavioral

parameter includes several features like; sender or receiver addresses of the malcode, types of attachments, and other considerable numeric characteristics. It usually happens in dual stages. In the first stage the behavior of system without the malcode is recorded and a normal behavior profile is created using ML methods. In the later phase this profile is matched against the present behavior and dissimilarities are considered as possible attacks (Robiah et al., 2009). The advantage of this technique is that it can detect known as well as new, unknown instances of malcode and it emphasizes on the behavior of system to detect zero-day attack. The weakness of this technique is that it needs updation of the system behavior and other useful data in usual profile but it is likely to be outsized. More resources such as processor time, random access memory space and storage space are required by this method.

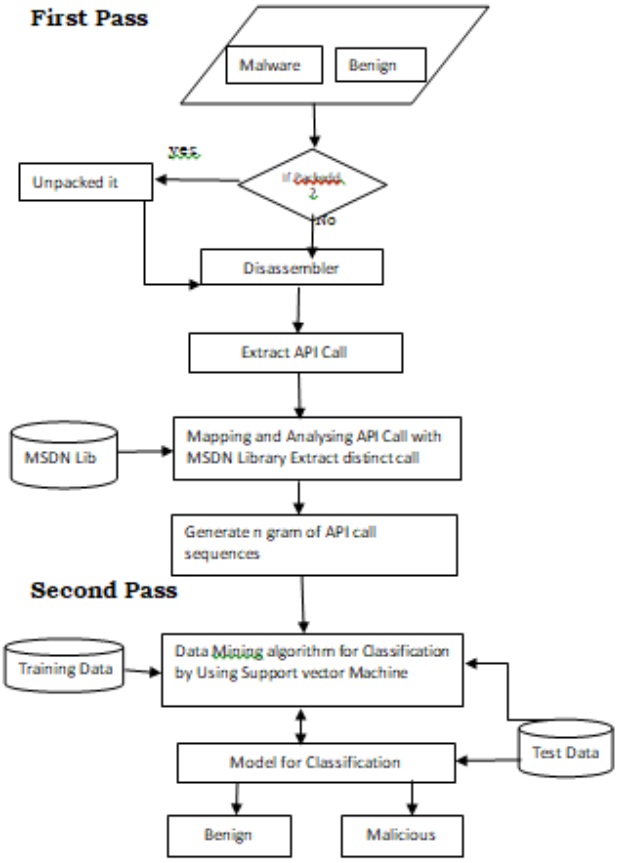

Figure 3: API-based malware detection Model

EXPERIMENTAL RESULTS

The models proposed are implemented and tested using WEKA tool for some experiments and Python based Anaconda framework (Jupiter notebook) for some other experiments. API based detection model, CAM model were tested using Weka tool whereas other models were experimented using python. Output of these experiments are presented in following subsections.

Missing Values handling: Data collected from different online sources are sometimes noisy and irregular. Incomplete data or missing data in the dataset may impact the performance of learning model which may result in incorrect predictions. Incorrect predictions eventually increases the FPR and FNR values which is practically not acceptable. Thus, proper pre-processing and cleaning need to be performed before building a training model. As per the survey on data analytics by IBM, most analysts spend approximately 80 percent of their model execution time in data preprocessing (Samantray & Narayan Tripathy, 2020). For this reason, the feature vector is scrutinized to find missing or invalid values using python instructions.

There are several techniques available in ML, to handle missing values but providentially no missing value found in the dataset. The dataset used in this experiment contains some attributes which contains textual value. This textual values are quantized and normalized before used by the classifiers. The dataset has two classes such as benign(represented as 0) and malware (represented as 1).

Feature selection: The feature selection techniques used in this experiment are; extra tree classifier and k-best feature selection. These two methods are applied individually to select a good feature selection method by comparing their results. Useful features are selected using these methods. Python based in-built libraries are used to implement these techniques.

K-best feature selection method: Arithmetic experimentations can be done to discover top features having firm relationship (strongly dependent) with the target attribute. SelectKBest class of python scikit-learn library is used to implement this method. In this case also we have selected top 20 features from the feature set. The criteria to select these best features is the Chi Square value as specified by the following formula. For any two variables, let the observed value is Oi and expected value is Ei. The chi-square method shown in equation (6) calculates the differences between these two values.

Where, c= Degree of Freedom

O= Observed values

E= Expected values

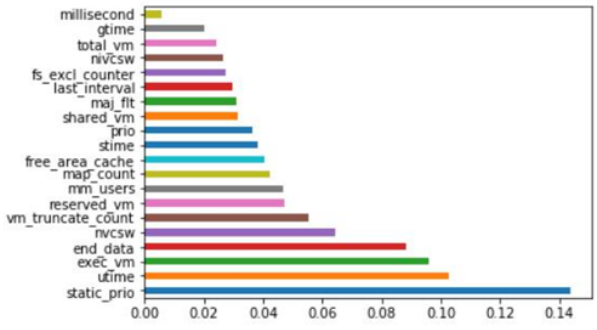

Figure 4: Twenty best features selected from the dataset using extra tree classifier

The best features along with their scores resulted from SelectKBest method are shown in figure 51.

	features	Score
1	state	2.325520e+09
13	map_count	1.201734e+07
4	static_prio	1.181309e+07
8	vm_truncate_count	8.720753e+06
28	utime	3.653671e+06
22	nvcsw	3.572497e+06
23	nivcsw	1.742535e+05
15	total_vm	1.284080e+05
18	reserved_vm	8.704946e+04
11	free_area_cache	5.921604e+04
3	prio	3.462974e+04
17	exec_vm	2.530067e+04
21	last_interval	1.346948e+04
12	mm_users	8.589701e+03
26	fs_excl_counter	8.000293e+03
16	shared_vm	8.699481e+02
20	end_data	8.699481e+02
25	maj_flt	8.699481e+02
30	gtime	1.331850e+02
24	min_flt	8.843142e+01

Figure 5: Features selected by K-best feature method based on their score

6.1. Evaluation of API call Based Supervised Learning Model

The metrics used to measure performance of this API-based detection model is as follows. Assume, the term "positive" denotes maliciousness and "negative" denotes benign.

- True Positive (TP): A malware identified as a malware.
- True Negative (TN): A benignware identified as a benignware.
- False Positive (FP): A benignware identified as a malware
- False Negative (FN): a malware identified as a benignware. Summary of these metrics are presented in the table 6.

	Actual	Identified as	Prediction Status
TP	Malware	Malware	Correct
TN	Benign	Benign	Correct
FP	Benign	Malware	Wrong
FN	Malware	Benign	Wrong

Table 6: Confusion matrix for performance metrics

It is witnessed that many among the selected algorithms achieved good accuracy when k=10. SVM (SMO) with Normalized PolyKernel has performed best in this model and Decision Tree has achieved the least among them. The result of this model with normalized polykernel SVM has better results compared to most of the related works presented in chapter 3. Fold-wise accuracy scores and mean score of the considered algorithms is given in table 7.

FOLD	DT	KNN	NB	J48	RF	SVM
2	88.13	90.43	92.12	89.42	92.86	93.12
3	88.32	94.00	91.23	91.41	93.81	93.21
4	88.63	93.02	92.36	92.13	94.22	93.21
5	88.75	94.22	90.82	93.00	94.80	93.45
6	88.91	94.13	91.73	93.45	95.51	95.00
7	89.04	94.04	91.47	93.25	95.92	97.18
8	88.59	94.00	91.50	93.08	96.48	98.10
9	88.95	94.17	91.32	93.33	96.66	97.98
10	88.95	94.42	91.11	93.33	96.83	98.20

Table 7: Fold-wise accuracy and Mean accuracy of the designated algorithms in API based malware detection

The comparative representation of the accuracies of all selected methods is depicted in figure 45. The X-axis denotes the fold number and Y-axis signifies the accuracy score. The final mean accuracies (y-axis) of the selected algorithms is depicted in figure

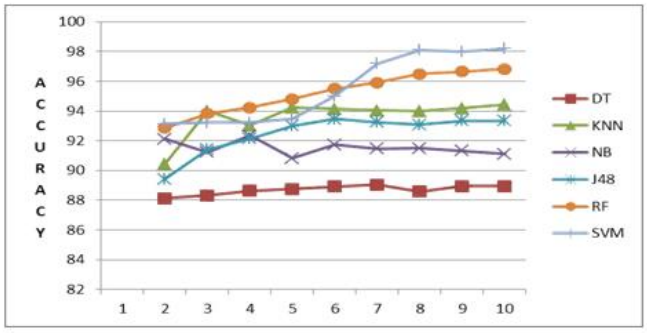

Figure 6: Fold-wise accuracy comparison of ML methods in API based detection model

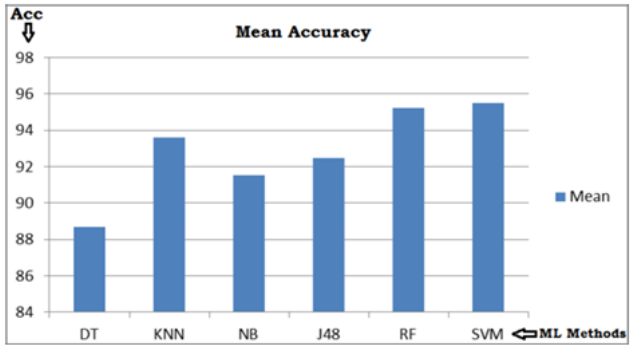

Figure 7: Mean Accuracy of API based model for the Algorithms

	mov	push	call	pop	cmp	jz	lea	test	jmp	add	...	fstcw	int	nop	pushf	rdtsc	sbb	setb	setle	shld	labels
0	3362	491	840	495	581	0	317	548	482	3477	...	0	85	340	17	0	150	1	0	0	0
1	74728	31431	14363	7866	9853	0	14338	7709	8331	45814	...	0	5453	3404	52	0	782	127	11	3	1
2	9782	12950	3422	6773	5324	0	3121	2125	1163	79531	...	0	339	1580	67	0	931	22	1	0	1
3	8224	1444	1757	1512	988	0	534	922	1203	8947	...	0	294	865	17	0	279	9	0	0	0
4	8510	2047	1241	985	1720	0	1727	505	618	2186	...	0	349	290	1	0	24	30	1	2	1

Figure 8: Snapshot of dataset after feature selection

The final feature set for the study contains 30 columns and 2736 rows. The number of samples from both classes is not in equal proportion which may create imbalance problem and

bias the model. For this reason we have applied "upsampling the minority class" method using python to make the proportion balanced.

Values are normalized in python so as to make the values fall in a specific range. Finally, the six selected algorithms are used for classification. The algorithms are DT, KNN, SVM, RF, NB and LR. The train-test-split is prepared using k-fold (k=10) cross validation method in python for all the selected algorithms. Performance of the models are measured using the following metrics.

CONCLUSION

Malicious software or malware has been a big threat to the field of information security from the last decade. As the use of technological resources and internet is growing, malware attacks are also growing with the same rate. For this purpose malware research is a continuous and ever-demanding research domain in computer science. In this thesis an attempt is made to contribute some important aspects of malware analysis and detection for for societal benefits.

Study of the state-of-the-art of this malware analysis and detection is carried out to understand the knowledge of this domain. A study is done by considering the type of malware analysis method and the kind of features used in the previous works. Based on the knowledge gained in the study, important features like API calls, opcodes, change-in system behavior and hybrid features are considered for the model designs and experiments. Experiments are conducted using WEKA and Python based frameworks to implement feature selection methods and machine learning algorithms.

The API call based detection model used 4-gram sequences of distinct. API calls which are selected using odd"s ratio method. The model is tested using six data mining algorithms such as DT, KNN, NB, J48, RF and SVM with default parameter values. This model has achieved 95.49% accuracy score with SVM algorithm which is highest among the selected algorithms.

REFERENCES

[1]. Abou-Assaleh, T., Cercone, N., Kešelj, V.,& Sweidan, R. (2004). N-gram-based detection of new malicious code. Proceedings International Computer Software and Applications Conference.

[2]. Ahmed, F., Hameed, H., Shafiq, M. Z., & Farooq, M. (2009). Using spatio-temporal information in API calls with machine learning algorithms for malware detection. Proceedings of the ACM Conference on Computer and Communications Security.

[3]. Alqurashi, S., & Batarfi, O. (2018). Acomparison between API call sequences and opcode sequences as reflectors of malware behavior. 2017 12th International Conference for Internet Technology and Secured Transactions, ICITST 2017.

[4]. Anderson, B., Quist, D., Neil, J., Storlie,C., &Lane,T. (2011). Graph-based malware detection using dynamic analysis. Journal in Computer Virology. .

[5]. Bazrafshan, Z., Hashemi, H., Fard, S. M. H., & Hamzeh, A. (2013). A survey onheuristic malware detection techniques. IKT 2013 - 2013 5th Conference on Information and Knowledge Technology.

[6]. Beniwal, S., & Arora, J. (2012). Classification and Feature Selection Techniques in Data Mining. International Journal of Engineering Research & Technology (IJERT).

[7]. Y., Tawbi, N.,Bergeron, J., Debbabi, M., Desharnais, J., Erhioui, M., Lavoie, Y., & Tawbi, N. (2001).Static Detection of Malicious Code inExecutable Programs. Control.Bilar, D. (2007).

