A Data Mining Based Malware Detection Model using Distinct API Call Sequences

Om Prakash Samantray, Satya Narayan Tripathy, Susanta Kumar Das

Abstract*: Malware is a serious threat from the last decade and the threat is increasing every year with the extensive use of internet. Rigorous researches have been going on to save our important information from being stolen and damaged by the malicious software. Despite many malware detection strategies, zero-day malware detection still is a challenge for the researchers. Here, we have presented a model which picks distinct API call sequences as feature and then uses data mining classification algorithms for malware detection. Distinct API call sequences are extracted from PE files which are supplied as input to different data mining or machine learning techniques. We have selected six robust data mining classifiers, namely Decision Tree (DT), Support Vector Machine (SVM), Naive Bayes (NB), K-Nearest Neighbor (KNN), J48 and Random forest (RF) to carry out the experiment. A comparison of their performance is also presented.

***Index Terms*: API Call Sequence, Data Mining, Malware analysis, Malware detection**

I. INTRODUCTION

Malware stands for malicious software which steals sensitive information from computers or other handheld devices. Malware may be of different types like virus, Trojan, spyware, adware, ransomware, botnet, rootkit and many more. With the rapid technological growth in information technology, the need and popularity of computing devices & mobile devices has also increased rapidly. We are handling huge data in the internet which have become the targets of malware creators. According to the report by an independent IT security institute, AV-Test, the trend of developing malware is doubled by the end of 2017 and over 67 percent of all malware attacks have targeted Windows based systems[1].

The big computer security threat is that the number and type of malware is increasing day by day. Malware creators not only create malware to steal information but also focusing on penetration of the malware. They also use various obfuscation techniques to pass through the anti-malware engines.

Malware detection techniques are categorized as Signature based detection and behavior based detection. Signature based approach is the popular technique which generates "pieces of binary code" or signature by breaking the file into pieces. Signature based detection can detect malware with a higher rate as long as the signature of incoming file is known to the detection engine, i.e. it provides a low FPR (False Positive Rate) for previously seen malware. However, zero-day malware may not be detected by this technique because the antivirus software cannot recognize the new signature in time. Anomaly based detection classifies a file as malware or benign by comparing its normal behavior with anomalous behavior. It uses different algorithms to extract features like API (application programming interface) calls, dynamic linking library (DLL), PE-Miner (Portable Executable) and byte-level n-gram features to detect previously unseen malware.

A. Code Obfuscation

In order to avoid anti-malware engines, various obfuscation practices are used while creating the malware. The techniques alter a program so cleverly, so as to preserve their functionalities while making the program hard to read and understand[2]. Polymorphism, dead-code insertion, subroutine reordering, code transposition, instruction substitution, metamorphism, packing are the examples of obfuscation techniques used by both malware writers and legitimate software developers for different purposes. Malware authors use them to escape from antivirus scanners. Packing, compressing and encryption are the most common obfuscation techniques. Packer software is used for these purposes. Polymorphic malware alter the build of the file by appending data to it. It also uses encryption techniques to encrypt the file and changes the decryption function time to time along with the change in encryption keys [3]. Metamorphic malware modifies the code without using encryption. Dead-code inclusion, code inversion, registers assignment and instruction substitution is the four commonly used metamorphic obfuscation techniques [4].

B. Windows API calls

API calls are used to understand behaviors of a particular code [5]. The windows operating system API provides all essentials to develop a program. The API can be language specific or operating system specific. The OS specific API for windows operating system can further be divided into native APIs, kernel mode API and user-mode API. For example, Linux OS has GNU C library that exposes the C APIs whereas in windows based systems the C runtime library is used to expose C APIs. These libraries invoke the underlying APIs to perform a specific function.

Revised Manuscript Received on May 07, 2019.

Om Prakash Samantray, PG Department of Computer Science, Berhampur University, Berhampur, India.

Satya Narayan Tripathy, PG Department of Computer Science, Berhampur University, Berhampur, India.

Susanta Kumar Das, PG Department of Computer Science, Berhampur University, Berhampur, India.

A debugger can be used to verify it by tracing an API call during runtime, e.g. in windows operating systems, the fopen API sometimes may results in a call to the "createfile" API, which further calls the native API "Ntcreatefile" before a switch to kernel mode code is made [6].

API is internal part of any OS which enables the programs and hence malware writers use APIs as a medium to perform illegitimate actions. Similarly, malware analysts and researchers use them as an effective feature for malware analysis and detection.

C.Need for the Study

Everyday new versions of malicious software are developed by malware creators to evade detection by anti-malware engines. It became a serious threat for the information industry from the past decade. Despite so many corrective measures, the threat is increasing in an unprecedented rate which is a motivation for the researchers to work on it. In this work we have focused on API calls as a feature to detect malware because APIs can be used very cleverly by malicious software to get away from the detection systems. Some of the malware detection techniques possess high FPR (false positive rate) because benign and legitimate files are misinterpreted as malicious software. It is also possible that some of the illegitimate files or malware are left undetected which leads to high false negative rates (FNR). Traditional signature based detection systems may not identify zero-day attacks. Therefore a proficient malware detection model is required to handle this drastic growth of malware classes.

D.Data Mining

Data mining techniques /algorithms play a major role in malware detection and classification. Many researchers has been using data mining algorithms for malware detection and classification. The classification algorithms that can be used for malware detection among others are; Naïve Bayes, J48, KNN, decision tree, regression, sequential minimal optimization algorithm (SMO), random forest, support vector machines, voted perceptron and so on.

The data mining techniques can be either supervised or unsupervised. Malware detection methods are also classified into two major categories such as: signature-based and behavior-based. Similarly malware analysis techniques are also of two types such as static and dynamic. All the above three i.e. malware analysis methods, malware detection approaches and classification techniques can be collectively used for malware detection. Fig 1 shows the taxonomy of malware detection approaches and possible classification techniques. The figure also depicts the popularly used malware features like API calls, binary features and assembly features used in detection systems.

II. RELATED WORK

API call data is used to know the behavior of malware. It can be collected either by using static analysis method or dynamic analysis method. A list of APIs can be extracted from Portable Executable (PE) format through static analysis approach [7] and API calls can be observed through dynamic analysis in a safe setup [8].

The API call information gathered from static and dynamic approach can be analyzed either by frequency-count of called APIs or by using data mining techniques on the collected APIs. API call sequence information extracted using static and dynamic approach can be used as malware feature and behavioral patterns respectively. [9].In addition to the above, another way of analyzing API call sequence information is through API call graph [10]. As a variation of API call graph analysis, Jang et al. used social network analysis approaches to get significant features for call graph analysis [11].

In [12], Shankarpani et al. used cosine similarity function and extended Jaccard measure to compute the resemblance among API call structures. In order to enhance the accuracy of the process. Riecket. al. [13] have considered extra information related to control flow and API argumentation. Various researches for malware detection are there in the literature. Some researches are network based and some are host based. The former detect malware by analyzing network traffic and the later utilize the behavior of system for analysis. Host based detection is focused in this work. Malware signature, dynamic behavior and malware binary code are the three ways of host based malware detection. These features can be extracted either by statically or dynamically. Static malware analysis may achieve higher detection speed but automatic static analysis is difficult because of packing techniques used in malware.

Kang et. al. [14] and Faruki et al. [15] have analyzed malware binary codes using static analysis. They used API strings of portable executables and used different algorithms to differentiate malware and benign programs. Wang et al. [16] extracted API call sequences which were compared against a database of distrustful behavior and then Bayesian classification was used to detect malware. Faraz et.al. [9] presented a tool to extract API call sequence which is then used for classification using machine learning algorithms. They have experimentally stated that analyzing windows API call sequence improves detection accuracy of the classifier. They have considered a small set of API classes for scalability analysis and achieved good accuracy when only the smaller API classes were monitored.

Yangfang ye et al. [17] proposed association mining based malware detection system in which they have analyzed windows API call sequences invoked by PE files. They have experimentally proved the correctness and effectiveness of their proposed system. Ronghua Tian et. al.[18] presented a method of malware classification in which they analyzed API call traces of PE after executing them in a controlled environment. Accuracy of their analysis result was around 97%. V. P. Nair et al. [19] in their research used the emulated environment to extract API calls. They used QEMU (quick emulator), a virtual machine monitor for analysis. They used pattern recognition method for classification. Bonfante et al. [20] in their research work, they analyzed configuration of files syntactically and semantically to identify different morphed malware. V.satyanarayan et. al.

[21] created base signature for all the malware categories rather than for a single malware which helped to identify previously seen and unseen malware. Shankarpani et al. [22] in their research shown that, malware detection techniques that use API calls result in highly accurate classifiers. As data mining is the core of our research, we have identified different data mining methods for malware classification.

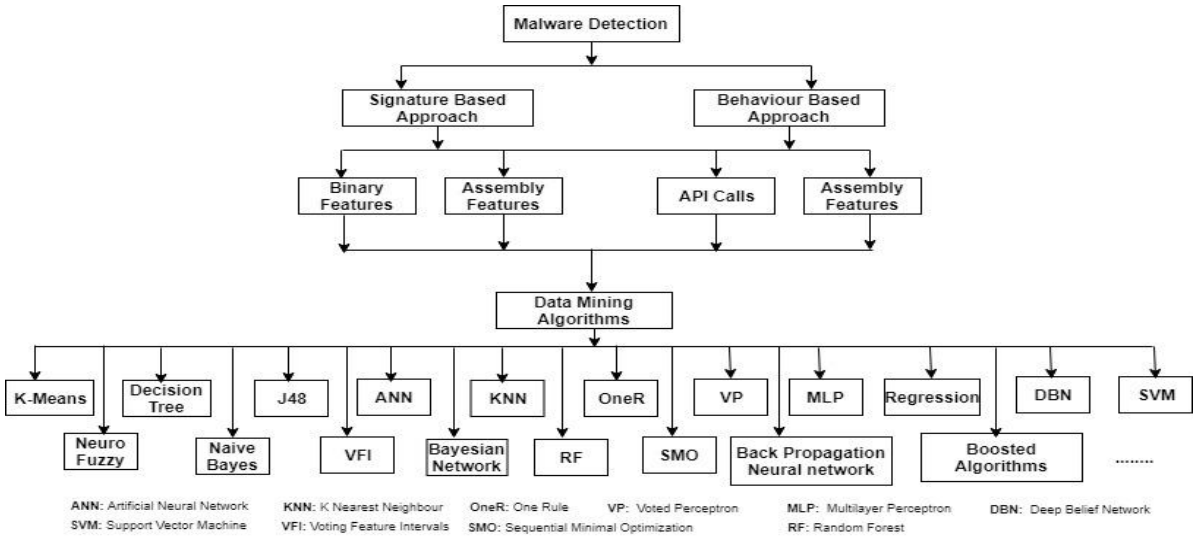

Fig 1 Taxonomy of data mining techniques for malware detection

Table I: Survey of data mining based malware detection approaches

SL. No.	Ref. No	Name of the Data Mining technique used	Analysis method	Type of detection	Accuracy claimed
1	[23]	K-Means	Dynamic	Signature Based	99
2	[24]	K-Means	Hybrid	Signature Based	89.8
3	[25]	SVM	Dynamic	Signature Based	94
4	[26]	SVM	Dynamic	Signature Based	98
5	[27]	SVM	Dynamic	Signature Based	98
6	[28]	SVM	Dynamic	Signature Based	97
7	[29]	SVM & NB	Dynamic	Signature Based	95.9
8	[30]	SVM	Hybrid	Signature Based	98.7
9	[31]	SVM, RF, J48, KNN & DT	Hybrid	Signature Based	99.8
10	[32]	SVM, MLP, J48, NB	Hybrid	Signature Based	98.6
11	[33]	SVM, NB, J48	Dynamic	Behavior Based	99
12	[34]	SVM, NB, DT, LR	Dynamic	Behavior Based	94
13	[35]	SVM, J48, Regression	Dynamic	Behavior Based	98.3
14	[36]	SVM, J48, IBK, NB	Static	Behavior Based	98.9
15	[37]	DT,SVM, Boosting	Static	Behavior Based	99.6
16	[38]	SVM, DT, RF	Dynamic	Behavior Based	96.9
17	[39]	SVM, NB, LR,MLP	Hybrid	Behavior Based	95.05
18	[40]	DT, SVM,NB	Dynamic	Signature Based	95
19	[41]	DT, NB	Hybrid	Signature Based	97.3
20	[42]	DT	Dynamic	Behavior Based	98
21	[43]	DT, KNN, RF	Dynamic	Behavior Based	99.3
22	[44]	J48,SVM,KNN,ANN	Hybrid	Signature Based	95.2
23	[45]	SVM, KNN	Hybrid	Signature Based	92.9
24	[46]	NB, BN	Hybrid	Behavior Based	95.27
25	[36]	RF	Dynamic	Behavior Based	86
26	[47]	ANN	Hybrid	Signature Based	98.9
27	[48]	KNN, LR, BN	Static	Behavior Based	97.6

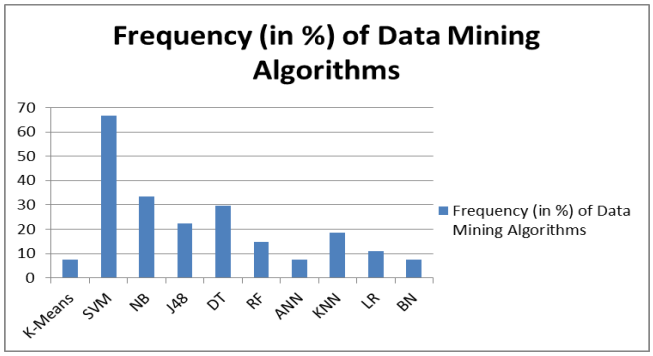

Fig 2 Frequency (in %) of Data mining algorithms related to Table 1.

As data mining is the core of our research, we have identified different data mining methods for malware classification. We have also identified the type of detection approach such as; signature or behavior based. Table 1 summarizes the survey. Based on the frequency of the algorithms used in those studied research works we have selected a few data mining algorithms to use in this research. Fig 2 shows the frequency (in %) of these data mining algorithms.

In this work we have presented a model for unique API features selection from PE executables and use of different data mining algorithms for classification. A comparative study on accuracy of these algorithms is also presented.

III. METHODOLOGY

A.API sequence extraction and analysis

The proposed model for malware detection is depicted in fig 3. The model takes portable executables (PE) as input in First pass. The PE files can be either malware files or benign files. The packed malware files are unpacked before being sent to a disassembler. On the contrary, the benign files are directly disassembled. IDAPro can be used as a disassembler [49]. APIs are then extracted from the disassembled files. The API calls can be mapped and analysed with MSDN library to get API sequence from the binaries.

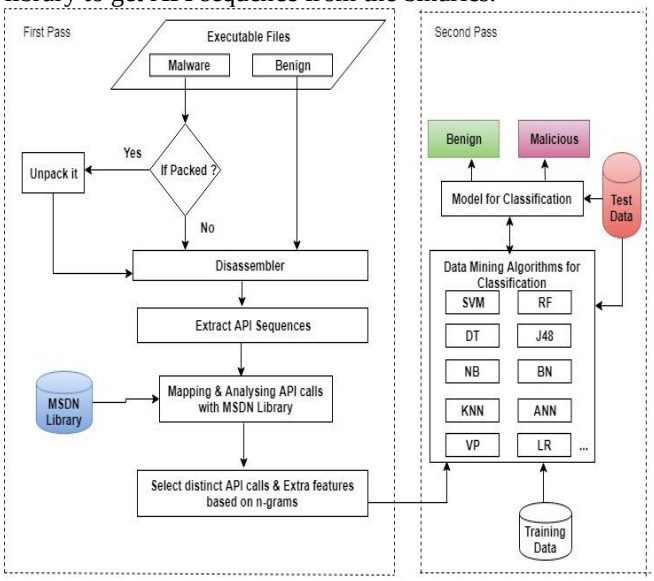

Fig 3: Methodology for API based malware Detection through data mining

B.Distinct API sequence selection

We have used a feature selection algorithm which generates n-grams of API calls and then finds Odds Ratio (OR) of all to get appropriate API call-gram for feature vector creation. Sliding window operations were used iteratively to generate n-grams of API calls. We have observed that 4-grams outperform 1-grams, 2-grams and 3-grams. The odds ratio is defined below.

$$OR(F) = \log \frac{P(F|b)\left(1 - P(F|m)\right)}{P(F|m)(1 - P(F|b)}$$

Where, P(F|b) and P(F|m) are the possibility for a feature to appear in benign class (b) and malicious class(m).

After calculating odds ratios, a feature vector are constituted using features having larger odds ratio.

C.Data mining algorithms for classification

Distinct API sequences are given as input to various data mining and learning algorithms. We have used Data Mining algorithms in our experiment. These algorithms are selected based on the higher percentage in frequency of use as depicted in fig 2. The algorithms used are, support vector machine (SVM), Naive Bayes (NB), decision tree (DT), J48, K-Nearest Neighbor (KNN) and Random forest (RF).

IV. EXPERIMENT AND EVALUATION

A. Sample Collection

We have conducted this experiment using 430 malware and 200 benign executables. The benign samples are collected from windows operating system and other application software, immediately after their fresh installation. The malware sample are collected from VX heaven virus collection dataset and other sources from internet and verified by virustotal. The malware dataset contains Trojans, viruses, worms and rootkits as verified by virustotal. We have used classification algorithms which require two datasets; one to train the classification model and another to test the model. Hence, we have used K-fold cross-validation technique. Here, the sample dataset is divided as k-partitions. Then the training and validation are done repeatedly for k times. In each iteration, a partition is used for testing and other partitions are used for training. In this way, the entire dataset is trained and tested through the selected data mining algorithms.

B. Performance Metrics

Metrics used for evaluation is shown in table II.

Where, TP is True Positive, TN is True Negative, FP is False Positive and FN is False Negative.

Table II: Performance Metrics

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

Accuracy is measured as follows.

$$\text{Accuracy} = \frac{\textit{Sum of True positive and True Negative}}{\textit{Sum of all metrics}}$$

C.Data Mining algorithms and results.

We have used WEKA tool to carry out the experiment for the following algorithms.

i. The Naïve Bayes algorithm (NB)

This algorithm used Bayes' theorem and it finds chance of occurrence of one event based on occurrence of other event which has already occurred. This algorithm is simple and efficient data mining algorithm. Fig 4(a) shows the accuracy of malware classification for Naïve Bayes with 10-fold cross validation.

ii. K-nearest neighbor algorithm (KNN)

It is a simple object-based classification algorithm. The classification is done on the basis of votes by K-nearest neighbors of the object. Fig 4(b) below shows the overall accuracy of this algorithm with 10-fold cross validation.

iii. Support Vector Machine (SVM)

SVM is a supervised learning method that outputs an optimal hyper plane to categorize linear and non-linear data. In this technique, n-dimensional space contains a number of points representing data Items. Here 'n' is the number of attributes in the dataset. Then to perform classification, it finds the hyper plane which differentiates the two classes [37]. Sequential Minimal Optimization (SMO) algorithm in WEKA is selected which is a fast implementation of SVM. We have chosen normalized polynomial kernel in SMO to perform the classification. Fig 4(c) below shows the overall accuracy of SMO with 10-fold cross validation.

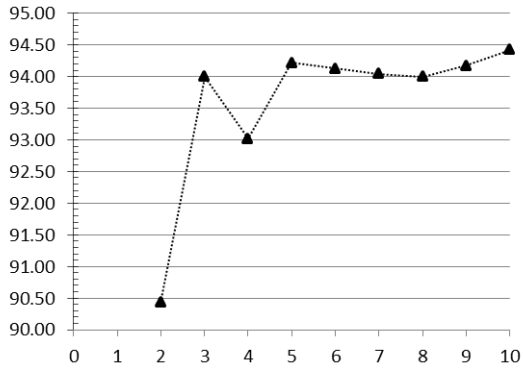

Fig 4(a): Performance of Naive Bayes algorithm

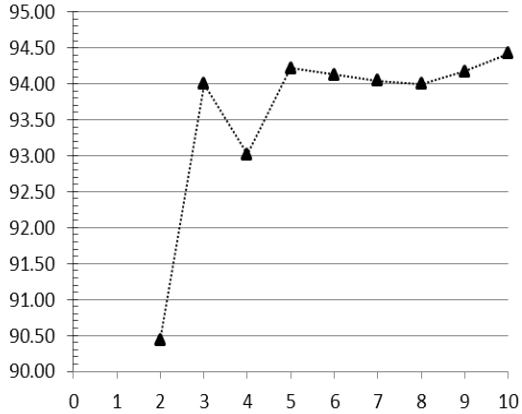

Fig 4(b): Performance of KNN algorithm

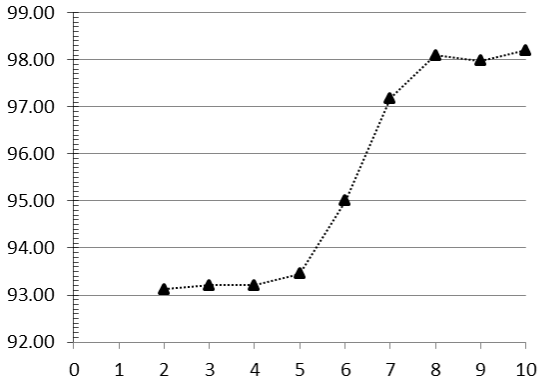

Fig 4(c): Performance of SVM / SMO algorithm

iv. Decision Tree (DT):

Decision tree is a powerful and widely used method for classification. We have used two different implementations of Decision Tree (ID3 and J48) in our experiment as we identified them as frequently used algorithms in our literature survey. The accuracy of ID3 with 10-fold cross validation is shown in fig 4(d).

v. J48 algorithm

It is based on C4.5 DT algorithm. In WEKA, J48 builds a C4.5 DT. Accuracy of J48/C4.5 with 10-fold cross validation is shown in fig 4(e).

vi. Random Forest algorithm (RF)

This is a learning method which constructs a number of DTs and then produces class as output. In other words, RF builds multiple decision trees and combines them together to get an accurate prediction. Accuracy of RF with 10-fold cross validation is shown in fig 4(f).

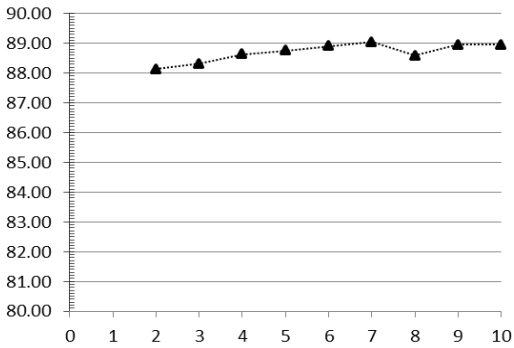

Fig 4(d): Performance of Decision Tree (ID3) algorithm

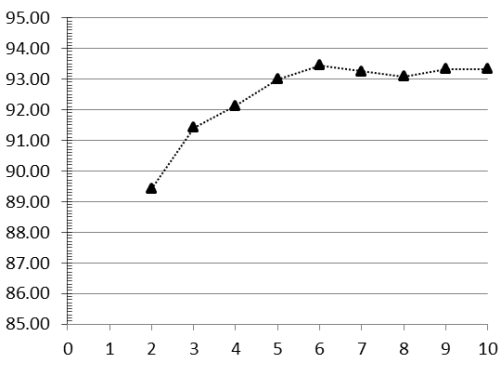

Fig 4(e): Performance of J48 algorithm

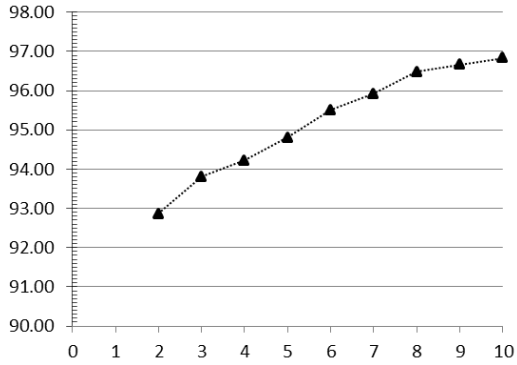

Fig 4(f): Performance of Random Forest algorithm

We have observed that most of the algorithms perform well with k=10. SVM (SMO) with Normalized PolyKernel has the highest accuracy and Decision Tree has the lowest accuracy among these six selected classification algorithms. Accuracy of all these six classification algorithms are shown in fig 5(a) and 5(b).

V. CONCLUSION

In this paper, we have suggested a malware detection framework based on API call sequences which are extracted using a disassembler and then mapped with MSDN library of Microsoft OS. APIs based on n-grams are selected using an algorithm. Distinct API call sequences based on Odds ratio are selected to constitute a feature vector. These selected API sequences are supplied to various classification algorithms with k-fold cross validation where, k is from 2 to 10.

We have observed in our experiment that, SVM with normalized polynomial kernel outperforms all the other classification methods. SVM has an accuracy of 98.2 when k va1ue is 10. Among the six classification algorithms chosen for this experiment, Decision tree (ID3 in Weka) possesses lowest accuracy value of 89.04.

As an extension to this work, we will consider few more features like DLL name, PE header along with API calls before applying classification algorithms. We believe that, number of samples may also affect the accuracy of these classification algorithms because more samples may increase the number of features for classification. Hence, the experiment can be carried out with different volumes of samples and a comparative analysis of the same can be done with this work in the future.

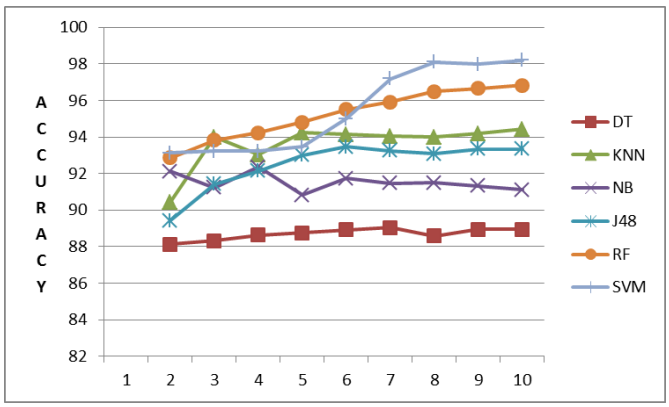

Fig 5(a): Accuracy of classification methods with k=2 to 10

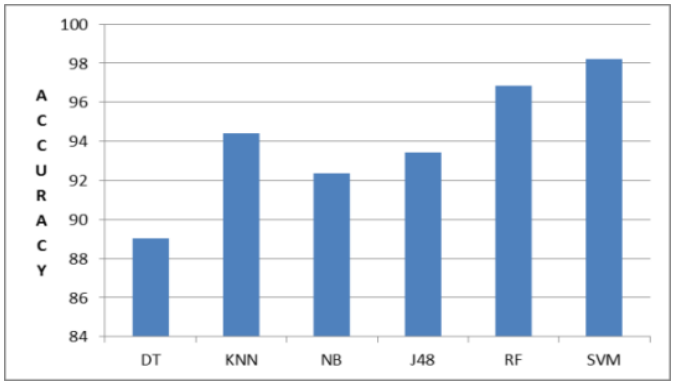

Fig 5(b): Accuracy comparisonof classification methods

REFERENCES

1. https://www.av-test.org/en/news/the-av-test-security-report-20172018-the-latest-analysis-of-the-it-threat-scenario/
2. Linn, C &Debray, "Obfuscation of executable code to improve resistance to static disassembly", in 10th ACM conference on computer and communications security, Washington, USA, 2003, pp. 290-299.
3. M Christodorescu, S Jha, S Seshia et al, "Semantics-aware malware detection" in IEEE Symposium ,Security and Privacy, 2005, pp. 32–46.
4. M. Alazab, Robert Layton, SitalakshmiVenkataraman and Paul Watters, "Malware Detection Based on Structural and Behavioural Features of API Calls".
5. Wang. C, Pang. J, R. Zhao and Liu. X, "Using API Sequence and Bayes Algorithm to detect Suspicious Behaviour", International Conference on Communication Software and Networks, 2009, pp. 544-548.
6. Sd. ZainudeenMohd. Shaid&Mohd. AizainiMaroof, "In Memory Detection of Windows API Call Hooking Technique", in IEEE international conference on computer, communication and control technology(I4CT), 2015, Malaysia, pp. 294-298.
7. A. Sami, B. Yadegari, H. Rahimi, N. Peiravian, S. Hashemi, and A. Hamze, "Malware detection based on mining API calls," in Proceedings of the 25th Annual ACM Symposium on Applied Computing (SAC '10), ACM, 2010, pp. 1020–1025.
8. Y.Qiao, Y. Yang,L. Ji, andJ.He, "Analyzing malware by abstracting the frequent itemsets in API call sequences," in Proceedings of the 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom '13), 2013, pp. 265–270.
9. F. Ahmed, H. Hameed, M. Z. Shafiq, and M. Farooq, "Using spatio-temporal information in API calls with machine learning algorithms for malware detection," in Proceedings of the 2nd ACM Workshop on Security and Artificial Intelligence, November 2009, pp. 55– 62.
10. J. Bergeron,M.Debbabi, J.Desharnais,M.M. Erhioui, Y. Lavoie, and N. Tawbi, "Static detection of malicious code in executable programs," in Proceedings of the Symposium on Requirements Engineering for Information Security (SREIS '01), 2001.
11. J.-W. Jang, J. Woo, J. Yun, and H. K. Kim, "Mal-netminer: malware classification based on social network analysis of call graph," in Proceedings of the Companion Publication of the 23rd International Conference on World Wide Web Companion (WWWCompanion '14), pp. 731–734, InternationalWorldWide Web Conferences Steering committee, 2014.
12. M. K. Shankarapani, S. Ramamoorthy, R. S. Movva, and S. Mukkamala, "Malware detection using assembly and API call sequences," Journal in Computer Virology, vol. 7, no. 2, pp. 107– 119, 2011.
13. K. Rieck, P. Trinius, C. Willems, and T. Holz, "Automatic analysis of malware behavior using machine learning," Journal of Computer Security, vol. 19, no. 4, pp. 639–668, 2011.
14. B. Kang, T. Kim, H. Kwon, Y. Choi, and E. G. Im, "Malware classification method via binary content comparison," in Proceedings of the 2012 ACM Research in Applied Computation Symposium, 2012, pp. 316–321.
15. P. Faruki, V. Laxmi, M. Gaur, and P. Vinod, "Mining control flow graph as API call-grams to detect portable executable malware," in Proceedings of the Fifth International Conference on Security of Information and Networks, 2012, pp. 130–137.

16. C. Wang, J. Pang, R. Zhao, W. Fu, and X. Liu, "Malware detection based on suspicious behaviour identification," in Proceedings of First International Workshop on Education Technology and Computer Science, 2009., 2009, pp. 198–202.
17. Yangfang Ye, Dingding Wang, Tao Li, Dongyi Ye. And Qingshan Jiang, " An Intelligent PE-malware detection system based on association mining", in Journal in Computer Virology, Vol 4, Feb 2008, pp. 323-334.
18. Ronghua Tian et al.," Differentiating Malware from Cleanware using Behaviour analysis",in proceedings of 3rd International Conference on Security of Information and Networks, SIN 10, IEEE, March 2010, pp. 23-30.
19. V. P. Nair et al., "MEDUSA: Metamorphic Malware Dynamic Analysis using Signature from API," in 5th International Conference on Malicious and Unwanter Software", ACM, 2010, pp. 263-269.
20. G. Bonafante et al.,"Architecture of Morphological Malware Detector", Journal in Computer Virology,2009, pp. 263-270.
21. V. Satyanarayana et al., "Signature Generation and Detection of Malware Families", ACISP, 2008.
22. Shankarpani et al.,"Kernel Machines for Malware Classification and Similarity Analysis", in the International Joint Conference on Neural Networks, Barcelona, 2010, pp. 1-6.
23. Fraley JB, Figueroa M," Polymorphic malware detection using topological feature extraction with data mining", in Southeast Conference 2016, pp. 1–7.
24. Wu B, Lu T, Zheng K, Zhang D, Lin X ," Smartphone malware detection model based on artificial immune system", China Comm. 11,2014, pp. 86–92.
25. Sun L, Li Z, Yan Q, Srisa-an W, Pan Y, "SigPID: significant permission identification for android malware detection", in 11th international conf. on malicious and unwanted software 2016, pp. 1–8.
26. Hashemi H, Azmoodeh A, Hamzeh A, Hashemi S," Graph embedding as a new approach for unknown malware detection", J ComputVirol Hacking Tech 13, 2017, pp. 153–166.
27. Li Z, Sun L, Yan Q, Srisa-an W, Chen Z ," DroidClassifier: efficient adaptive mining of application-layer header for classifying android malware", in proceedings of 12th international conference, securecomm, China, 2016,Springer, pp. 597–616.
28. Boujnouni ME, Jedra M, Zahid N , " New malware detection framework based on N-grams and support vector domain description", in 11th international conference on information assurance and security, 2015, pp. 123–128.
29. Bat-Erdene M, Park H, Li H, Lee H, Choi MS, " Entropy analysis to classify unknown packing algorithms for malware detection". International Journal of Information Security, 2017, pp. 227–248.
30. Wang P, Wang Y-S, "Malware behavioural detection and vaccine development by using a support vector model classifier", J ComputSystSci, 2015, pp. 1012–1026.
31. Rehman Z-U, Khan SN, Muhammad K, Lee JW, Lv Z, Baik SW, Shah PA, Awan K, Mehmood I, "Machine learning assisted signature and heuristic-based detection of malwares in Android devices", ComputElectrEng, 2017.
32. Palumbo P, Sayfullina L, Komashinskiy D, Eirola E, Karhunen J, "A pragmatic android malware detection procedure", Computer Security 70, 2017, pp. 689–701.
33. Boukhtouta A, Mokhov SA, Lakhdari N-E, Debbabi M, Paquet J, " Network malware classification comparison using DPI and flow packet headers", J ComputVirol Hacking Tech 12, 2016, pp. 69–100.
34. Miao Q, Liu J, Cao Y, Song J, " Malware detection using bilayer behavior abstraction and improved one-class support vector machines. IJIS, 2016, pp. 361–379.
35. Norouzi M, Souri A, SamadZamini M, " A data mining classification approach for behavioral malware detection", J ComputNetw Communication, 2016, pp. 6:9.
36. Sheen S, Anitha R, Natarajan V, "Android based malware detection using a multifeature collaborative decision fusion approach", Neuro computing 151(Part 2), 2015, pp. 905–912.
37. Siddiqui M, Wang MC, Lee J,"A survey of data mining techniques for malware detection using file features", in Proceedings of the 46th annual southeast regional conference, 2008. ACM.
38. Galal HS, Mahdy YB, Atiea MA, "Behavior-based features model for malware detection", J ComputVirol Hacking Tech 12, 2016, pp. 59–67.
39. Dali Z, Hao J, Ying Y, Wu D, Weiyi C, "DeepFlow: deep learning-based malware detection by mining Android application for abnormal usage of sensitive data", in IEEE symposium on computers and communications (ISCC), 2017, pp. 438–443.
40. Fan CI, Hsiao HW, Chou CH, Tseng YF, "Malware detection systems based on API log data mining", in IEEE 39th annual computer software and applications conference, 2015, pp. 255–260.
41. Cui B, Jin H, Carullo G, Liu Z, "Service-oriented mobile malware detection system based on mining strategies", Pervasive Mob Comput 24, 2015, pp. 101–116.
42. Mohaisen A, Alrawi O, Mohaisen M, "AMAL: high-fidelity, behavior-based automated malware analysis and classification. ComputSecur 52, 2015, pp. 251–266.
43. Wuechner T, Cislak A, Ochoa M, Pretschner A, "Leveraging compression-based graph mining for behaviour based malware detection", IEEE Trans Dependable SecurComput 2017.
44. Fan Y, Ye Y, Chen L, "Malicious sequential pattern mining for automatic malware detection", Expert SystAppl 52, 2016, pp. 16–25.
45. Santos I, Brezo F, Ugarte-Pedrero X, "Opcode sequences as representation of executables for data mining- based unknown malware detection", InfSci 231, 2013, pp. 64–82.
46. Nikolopoulos SD, Polenakis I, "A graph-based model for malware detection and classification using system-call groups", JComputVirol Hacking Tech 13, 2016, pp. 29–46.
47. Malhotra A, Bajaj K, "A hybrid pattern based text mining approach for malware detection using DBScan", CSI Trans ICT 4, 2016, pp.141–149.
48. Wu S, Wang P, Li X, Zhang Y, " Effective detection of android malware based on the usage of data flow APIs and machine learning", Information Software Technology 75, 2016, pp. 17–25.
49. https://www.hex-rays.com/products/ida/index.shtml.

AUTHORS PROFILE

Om Prakash Samantray got the M.Tech degree in Computer Science & Engineering from Biju Patnaik University of Technology, Odisha, India in 2010. Currently, he is pursuing Ph.D. in Computer Science from Berhampur University, Odisha, India. He has 11 years of teaching experience with different Engineering Institutes in Odisha and Andhra Pradesh. His research interests include Information Security, Computer network security, Data warehousing & Mining, Machine Learning, IoT and big data.

Dr. Satya Narayan Tripathy received his M.C.A. and Ph.D. degrees in Computer Science from Berhampur University, Berhampur, Odisha, India in the years 1998 and 2010, respectively. He has been teaching in the Department of Computer Science, Berhampur University since 2011. Currently, he is a Lecturer in the Department of Computer Science, Berhampur University. Dr. Tripathy serves on the advisory boards of several organizations and conferences. He is a Life Member of Computer Society of India (LMCSI), Life Member of Orissa Information Technology Society (LMOITS) and Member of several professional bodies. His research interests include computer network security, wireless ad hoc network, network security in wireless communication and data mining.

Dr. Susanta Kumar Das received his Ph.D. degree from Berhampur University, Odisha, India in 2006. Dr. Das is currently a Reader at the Department of Computer Science. He is a life member of IEEE, ISTE, SGAT, OITS and member of several professional bodies. His research interests include Data Communication & Computer Networks, Computer Security, Internet & Web Technologies, Database Management Systems and Mobile Ad-Hoc Networking & Applications.