Examine Behavioral Aspects of API calls for Malware Detection and Categorization

Praveen Yadav[1], Pooja Yadav[2]

[1]*M.Tech Scholer* , [2]*M.Tech, Computer Science & Engineering*

[1]*Govt. Engg. College, Ajmer , Mody University of science and technology Lakshmangarh India*

Abstract— Present generation scenario shows a drastic rebound in the success of the malware. According to Kaspersky Security Lab reveal, India ranks seventh in offline threats and ninth in online threats caused by malware, among top ten countries of the world. Advancement in the burial techniques appreciate code obfuscation, packing, encryption or polymorphism help malware writers to avoid detection of their malwares by Anti-Virus Scanners (AVS), as AVS to a great degree fails to identify unknown malwares. In this paper we elucidate a malware detection approach based on mining behavioral aspects of API calls, as extraction and style of API calls can promote in imperative the practice and functions of a program. We propose a feature selection algorithm to appoint unique and distinct APIs and then we have applied gear learning techniques for categorizing malicious and benign PE files.

Keywords— Behavioral Aspects, Malware, Data mining, API Call,Portable Executable (PE

I. INTRODUCTION

Malware is complete code, program or software that has harmful intentions to agree the principle of a computer route or entire other mechanical device. It boot be self-replicating and can install itself without vigorous user actions and can control or consume the sensible system functionalities. Malware attacks have increased at an alarming worth in the late years. The function for this is beautiful variants of malware are introduced everyday which are impossible to detect by the reveal anti-malware engines and available malware detection techniques. Malware detection techniques cut back be broadly classified as signature based detection and anomaly based detection techniques. Anti-malware engines exert the engross based detection schema in which the appeal is occasional into pieces and some rare conscience is picked up to stir ‘byte sequences’ or ‘signatures’. Signature based detection can provide fancy detection rate for the malwares whose signatures are disclose in the database yet the unknown or ‘zero day’ malware cannot be detected by this technique. Anomaly based detection consider a bit of code as dangerous by analyzing its levelheaded behavior and comparing it by all of its anomalous behavior. Therefore, deformity based detection can notice zero day attacks using achievement extraction, by intercepting API calls etc. A new contest nowadays, for the malware detection analysts is the handle of obfuscation techniques to inspire new varieties of malware. The obfuscated code is diverse from its previous versions and thus, it bypasses the malware scanning. Some of these concepts are discussed in this section.

A. Obfuscation

The obfuscation technique modifies a program consist of in such a process that the functionality of the program surplus the same yet makes it illegible to decipher its signature or to launch reverse engineering attempt on it. Thus, obfuscation bouncecel be secondhand by attackers to modify a malware so that its functionality garbage the same but the generated offspring copies have different virus signatures and appropriately, these beautiful copies cannot be detected by the antivirus scanner whose database contains the sign of the parent malware. Some of the obfuscation techniques are dead-code insertion, subroutine reordering, conduct transposition, proposal substitution etc. [2]

B. API Calls

Application Programming interface is the subsequent part of the windows hired system. All the programs in windows boot interact mutually the windows API [3] and access the predefined functions by making API calls to make use of the services provided by it savor base services to access resources savor file systems, processes, threads and devices, advanced services to retrieve windows registry, window attempt services, consolidate services etc. Extraction and word of API calls can bolster in determining the style and functions of a program

C. Malicious Programs

Malicious programs can conceal their behavioral aspects by per Win32 API what one is in to and can contaminate the executable. Data mining techniques and duplex static analysis bouncecel prove useful to a large extent to detect the obfuscated malwares by analyzing their attentive behavior.Portable Executable (PE) File Format

Win 32 PE claim format is the most popular charge format by Microsoft and is secondhand for executables and object codes. It is a comprise of data structure for encapsulating the important information required by the

windows loader. To consider an obfuscated malware, API request features are excerpted and this process can be automated. This requires a attentive insight into the PE structure. A PE file is impending mapped facing the memory by a forceful linker. A PE charge is plus various headers and sections which are used by this linker to explain the mapping. The executable claim consists of .text, .data and .rdata sections where .text article contains the position code and its mapping is done as execute/readonly, .data requirement comprises of writable full variables and its mapping is done as non-execute/readwrite and completely, .rdata article consists of put only data. These data structures in malware executable claim formats are made undiscoverable for detection engines by per obfuscation by all of packers or polymorphism. [4]

D. Motivation

Day by day, new malware variants are over devised by attackers to pose on up and up threats on computer systems. No base hit malware detection methodology is efficient enough to notice all the classes of malware discipline to their obfuscated nature. Also, API strings' functionalities boot be utilized very intelligently by hard softwares to conceal the detection scans. The current malware detection systems have very valuable false positive rates because benign files perchance misinterpreted as dangerous whereas, the obfuscated malware rest undetected head to valuable false negative rates. Unknown malware manage lead to zero day attacks and therefore a slim malware detection schema is required to deal with these exponentially promising encoded malware classes.

II. RELATED WORK

The work done by kephart and arnold [6], eventual the major trade done in word mining. The dealer codes of viruses are analyzed by writ by hand based detection and a if it cool for an instruction in a conscience is estimated. The mix of conscience mutually minimum false convinced is considered as polished signature and is earlier used in age detection.

In the research by wong and stamp [7], distinct metamorphic generators are analyzed. A similarity almanac is defined to speculate the intensity of modesty each generator produces. A detector based on hidden markov model (hmm) is presented. This approach presents highly unassailable results which are not convenient by at variance avs. The sooner to benefit malware detection over data mining were the authors in [8] by by the agency of three types of features: byte merger features, pe headers and strings. Various classification algorithms savor decision tree, naïve bayesian and ripper are used to regard also-ran viruses in computer. This act showed that machine learning approach is preferably superior to signature based detection.

The approaching mechanism [9], analyses the system request sequence and generates a topological tree called code graph. By for the conscience graph, happy software's and dangerous softwares are categorized. In [10], the authors exposed a graph based analogy of executables. The executables are disassembled, and earlier the control flow graphs are generated. To construct a flow graph, fundamental bocks are generated and the litany in the blocks is defined. To flash for the dreariness among the executables, graph isomorphism is employed. In the concern [11], a "phylogeny" ideal is constructed that uses n-perms to link the similarity variants of programs. The ability of n-perms and n-grams are calculated by an demonstrate which shows n-perms is more pragmatic and phylogeny tree contributes laborer in laborer to generate a transcend result.authors [12] and [13] exposed a novel clear to detect several variants of morphed malware. Both syntactic and semantic practice of a position is analyzed to regard morphed malware variants. In their expected employment malware examiner [14] advent an variety in the area of non-signature based detection. The audition was performed on 790 infected samples and it was do that the proposed approach found at the edge of about 84% of malware with 30% false positive rate. Advancement during the actual malware detector is done by the author [15], turn malware normalization that reverse engineers an obfuscated code and produced a normalized executable. Normalization eliminates the obfuscated code from the program to gain the capability of the malware detector.authors in paper [18], smartly proposed the what one is in to that gives relation between api calls and malware semantics. Instead of constructing a base signature for a hit malware, the authors created the headquarters signature for perfect class of malware, which then detects the unknown malware as well. In this paper we present a novel behave to obtain unique features of apis and consider seven different classification algorithms, get increased veracity rate.

III. PROPOSED WORK

This section elaborates the overall methodology as shown in fig. 1, which consist of two phases- training and testing phase

a) In training phase, the collected data samples are disassembled by IDA Pro disassembler [1]. Before disassembling malware samples are checked, if packed or not. If they are packed then first they are unpacked and then disassembled. IDA Pro used SQLite as plug in to generate various tables of information.

The output is then used for API extraction. The following step comprises of mapping and analyzing API calls with MSDN library.

b) In the next step proposed feature selection algorithm based on fisher score is used to select the distinct API calls. Initial set of API calls are labeled as either benign or malicious. Consider initial set of API calls is $API_i = \{API_{i1}, API_{i2}, API_{i3}\ldots\ldots\ldots API_{in}\}$, where each API_{ij} represents a feature. Hence a portable executable file F can be represented in term of feature vector μ_i.

$$\mu = \begin{cases} 1 & \text{if F import } i_{th} \text{ API call} \\ 0 & \text{if F doesn't used } i_{th} \text{ API call} \end{cases} \quad \textbf{(1)}$$

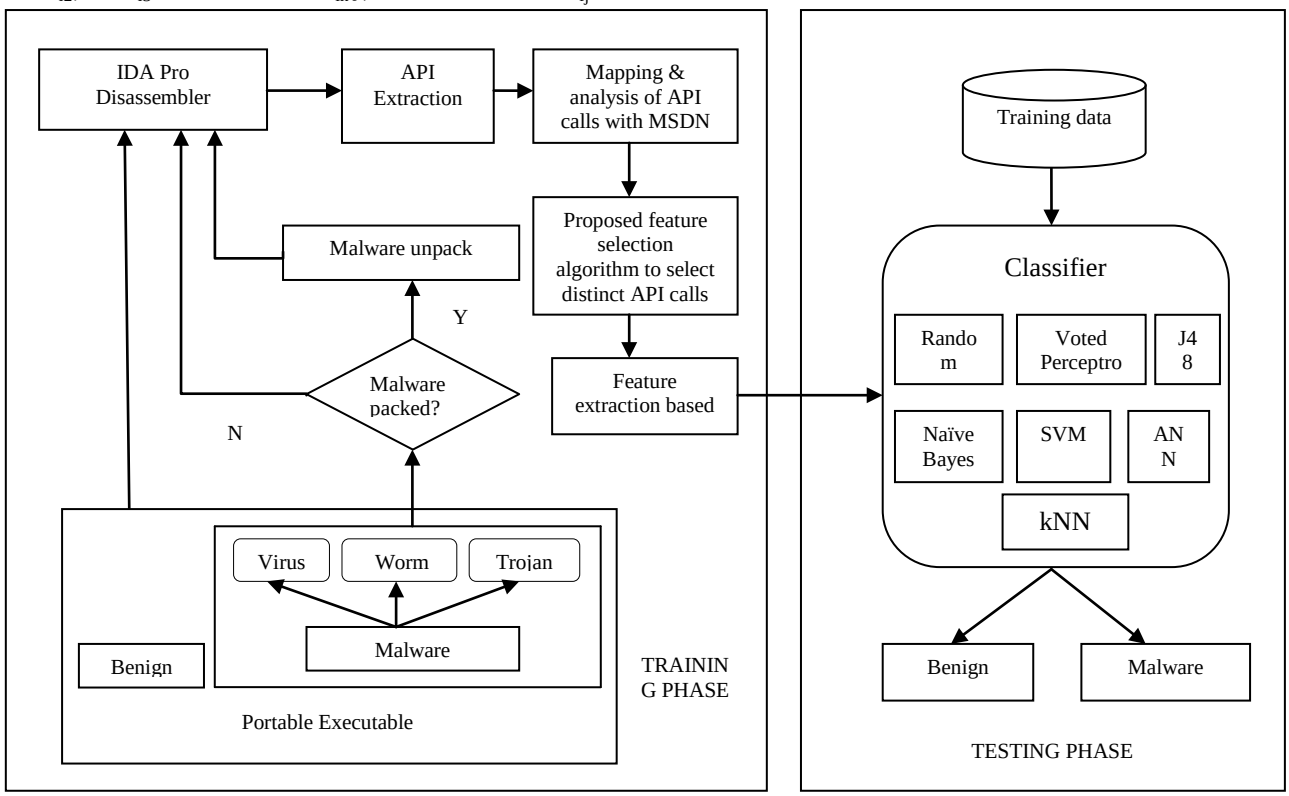

Fig. 1: Proposed Malware Diagnosis Methodology

But for classification purpose single API feature cannot be used to decide maliciousness in a PE file so sequence of API calls are needed, that are capable of diagnose malicious behavior in a PE file.

We employ an algorithm based on Fisher score to find API calls sequences from the initial set of API calls to improve the classification accuracy. The Fisher score can

$$\Phi = \frac{\sum_{k=1}^{c} n_k * (\mu_k - \mu)^2}{\sum_{k=1}^{c} n_k \sigma_k^2} \quad \ldots\ldots\ldots\ldots\textbf{(2)}$$

Where in class i

n_i is the number of data samples

μ_i is the average feature value

σ_i is the standard deviation of the feature value and μ is the average feature value in the whole dataset.

From equation 2 it could be understood that Fisher score is maximized when numerator is maximized and denominator is minimized. It shows that features with higher inter-class values and lower inter-class variance have higher Fisher scores, so these features are consider as set of API sequence capable of identify maliciousness in a PE file. Pseudo code of the proposed algorithm is shown in algorithm 1.

Algorithm 1: Proposed feature selection algorithm

Input:

API = Initial set of API calls

($API_i = API_{i1}, API_{i2}, \ldots\ldots\ldots\ldots API_{in}$)

DB = Portable executable database

Θ = Threshold limit

Output:

API_s (I*) = Set of selected API calls

$$API_{S(I^*)} = API_{I^*(1)}, API_{I^*(2)}, API_{I^*(3)} \ldots API_{I^*(n)}$$

Procedure:

1 Calculate score of each API_i defined in equation 2.

2 Rank the API according to the calculated score in descending order.

// Select API calls sequence to form $API_{S(I^*)}$

3 for all PE sample $ in DB do

4 if count (intersection ($API_{S(I^*)}$, $) ≥ Θ

5 then

Skip iteration

[End of step 4 if statement]

6 for API call γ in API do

7 if γ Є $ and count ($API_{S(I^*)}$, $) < Θ

8 then

$API_{S(I^*)}$ = union($API_{S(I^*)}$,γ)

API = API – γ

[End of step 7 if statement]

[End of step 6 for loop]

[End of step 3 for loop]

c) After selecting distinct API calls, the final step of training phase comes into action, of extracting features based on n-grams. The extracted features are fed as input to the classifier in testing phase. In our experiment we had used WEKA software as a classifier for classification. Various classification algorithms of WEKA software is used like- Random Forest, SMO, Voted Perceptron, Naive Bayes, and KNN, ANN and J-48 algorithms.

IV. CLASSIFICATION ALGORITHM

The following section will discuss various classification algorithm of Weka tool as carried out in our experiment. WEKA is a data mining tool**,** generally used for classification. WEKA stands for Waikato Environment for Knowledge Analysis. It is a JAVA based software, developed at University of Waikato, New Zealand [17]. It contains tools for analysis like regression, clustering, classification, visualization, data preprocessing and association rules.

Our experiment performs k- fold cross validation across each algorithm of Weka software. K cross validation is based on the concept of splitting the input set into a training set and a testing set. K-folds here show the number of partitions of input set into training set and testing set. In our experiment k = 2 to 10 folds cross validation.

E. Voted Perceptron Algorithm

The VP method is based on the perceptron algorithm proposed by Rosenblate and Frank. The linearly separable data with large margins are the main advantage of this algorithm. It takes less computation time as compared to Vapnik's SVM (Support Vector Machine) algorithm. VP is simpler to implement and much more efficient.

Fig.2 shows the performance rate of VPA achieved by our experiment.

F. Random Forest Algorithm (RFA)

This algorithm as proposed by the Leo Breiman and Adele Cutler is am ensemble learning means for constructing a number of order trees at training time and then outputs the share i.e., the nodes of the classes outputted by isolated trees.

Each tree in RF takes the figure of any old way vector which have been sampled fundamentally with the agnate distribution for bodily trees in the concerning forest. It is often that single order tree has high variance. The algorithm solves this problem by averaging to see a impulsive balance mid the two extremes. It constructs the tree that considers K casual features at each node. It holds the major bulk of running efficiently over large database. Fig.2 shows the performance rate of RFA achieved by our experiment.

G. Naïve Bayes Algorithm (NBA)

The NBA is based on the Bayes' theorem of conditional probability. The Bayes Theorem says that the probability of certain event to occur given that another event has already occurred. Like,

$$P(A/B) = P(B/A).P(A) / P(B)$$

Where P (B) is constant for all classes

P (A) is relative frequency of class A

This algorithm proves itself for fast, easy, efficient and effective learning algorithm for machine learning.

Fig.2 shows the performance rate of NBA achieved by our experiment.

H. J-48 Algorithm

J48 algorithm is the implementation of WEKA software based upon J.R.Quilan C4.5 algorithm. Depending on the attribute value of the given input data (training data), the classifier, firstly, creates a decision tree for classification. In Weka, J48 class builds a C4.5 decision tree. Whenever J-48 is executed by Java Virtual Machine (JVM), an

instance of a class is created by allocating memory for storing and building a decision tree. J48 uses references of instance of classes for doing most of its work.

As this algorithm, of its own does not contain any code for building a decision tree.

Fig.2 shows the performance rate of J-48 Algorithm achieved by our experiment.

I. Sequntial Mininmal Optimization Algorithm (SMO)

SMO is an implementation of SVM (Support Vector Machine) algorithm in Weka software. SMO solved the SVM QP problem by expanding QP, as SVM classifier performs only decision boundary among two classes in an input space. SMO advantages at solving Lagrange Multiplier analytically. This algorithm is fast and is also used for regression, by constructing hyperplane(s) in an n-dimensional space.

Fig.2 shows the performance rate of SMO algorithm achieved by our experiment.

J. K Nearest Neighbor Algorithm (KNN)

It is simplest of all algorithms of supervised machine learning algorithm. It classifies the object depending on the majority of vote of its K nearest neighbor at closest distant from the object.

Fig.2 shows the performance rate of KNN algorithm achieved by our experiment.

K. Artificial Nural Network (ANN) Algorithm

Artificial Neural Network, in short, also named as Neural Network (NN), is biologically inspired by nervous system of animals. It is a computational model of distributed computing. The model comprises of input nodes, some hidden nodes and output nodes with weights assigned to nodes (Li & Geo, 2008). As data is given to the network during the training process, the weights of the nodes are adjusted accordingly. NN consists of three steps of classification procedure- preprocessing of data, training of data and data testing.

Fig.2 shows the performance rate of ANN achieved by our experiment.

V. EXPRIMNETAL RESULTS

Experimental tests were performed on the 90 malicious samples and 120 benign files. Benign samples are collected from the freshly installed windows including application software like spreadsheet word processing, mathematical software, browsers, internet and other softwares. In this experimentation process, samples of malware were gathered from [16] and other relevant sources. The behavioral aspects of malicious code were extracted by analyzing the set of distinct API sequences. The novel feature selection algorithm which is put forward in this paper, first picks out the distinct API calls from the APIs drawn out in the previous step and then these unique API calls act as input for the classifiers which are used to determine whether a PE file is malicious or not. When the k cross validation technique was used with all of the classification algorithms discussed above, it can be inferred that the best accuracy for all the algorithms is achieved when k=10.

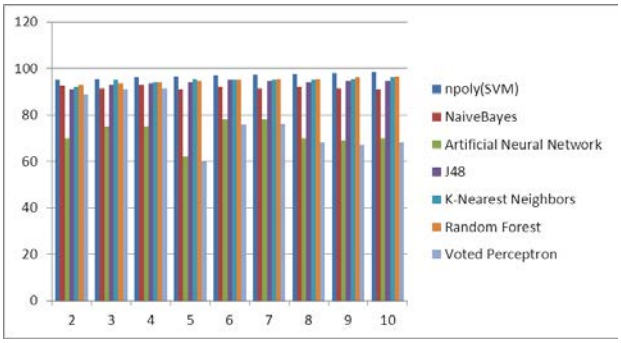

Fig. 2: Performance rate of algorithms.

Figure 2 lists various classification algorithms along with their performance rates when k cross validation is applied. On analyzing the above results, we have noted that SVM yields the best output as compared to the other classifiers.

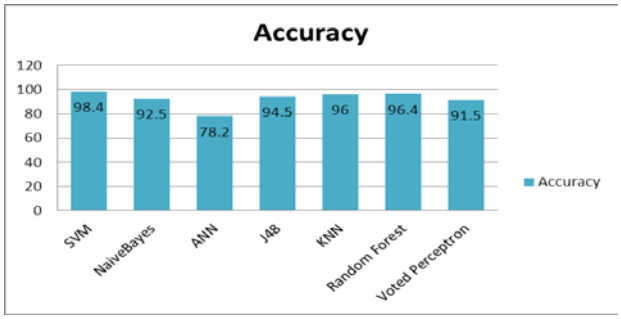

Fig.3.

TABLE I. Comparison of various malware detection methods

Authors	Techniques	Features	ACC, DR, TPR
Fatemeh Karbalaie et al.[19]	Technique based on graph mining	Behavior graph for malware diagnosis	DR-96.6% FP-3.4%
Mojtaba Eskandari et al.[20]	Technique based on Control Flow Graph	Detection of metamorphic malware using behavioral aspects	ACC-97.77% DR-97.53%
Kyoochang	Static	Malware	67%

Jeong et al.[21]	Analysis	identification using code graphs	
Yoshiro Fukushima et al.[22]	Behavioral Analysis	Behavior based detection of suspicious process	DR-60% FP-0%
Faraz Ahmed et al.[23]	Dynamic Analysis	Uses spatial and temporal information present in API calls for malware detection API calls	ACC-97%
Ammar Ahmed E. Elhadi et al.[24]	Combination of static and dynamic analysis	Combine static and dynamic analysis with behavior and signature based approach	Not specified
Ronghua Tian et al.[25]	Behavioral Analysis	Uses behavioral features of API calls to distinguish malicious and benign files	ACC-97%
Our method	Behavioral Analysis	Exploring Behavioral Aspects of API calls for Malware Identification	ACC-98.4%

VI. CONCLUSION

In this work, we have used active machine learning techniques with seven different classifier resulted in increased accuracy rate. In this method, proposed feature selection algorithm is used to select the unique and distinct API calls which are then classified using different data mining methods. On comparing the obtained evaluated results, we observed that among all classifier, SVM exhibited the best results in all measures.

REFERENCES

[1] https://www.hexrays.com/products/ida/support/download.shtml

[2] Ilsun You and Kangbin Yim "Malware Obfuscation Techniques: A Brief Survey" in BWCCA '10 Proc. of the 2010 Int. Conf. on Broadband, Wireless Computing, Commun. and Applicat., 2010.

[3] API calls: http://en.wikipedia.org/wiki/Windows_API

[4] http://msdn.microsoft.com/en-us/magazine/cc301805.aspx

[5] Nikita Jain and Vishal Srivastava "DATA MINING TECHNIQUES: A SURVEY PAPER", Int. J. of Research in Eng. and Technology.

[6] J.O.Kephart and B.Arnold, "A Feature Selection and Evalution of Comput. Virus Signatures", in Proc. of the 4th Virus Bulletin Int. Conf., 1994, pp 178-184.

[7] W.Wing and S.Mark, "Hunting for metamorphic malware" in Journal in Comput. Virology.

[8] M.G.Schultz, E.Eskin, E.Zodak and S.J.Stolfo. "Data Mining Methods for Detection for New Malicious Executables", in proc. of the IEEE Symp. on Security and Privacy, Washington, 2001, pp 38-49.

[9] Kyoochang et al., "Code Graph for Malware Detection", Int. Conf. on Inform. Networking, ICOIN, 2008, pp 1-5.

[10] R.T. Dullien and B. Universitaet, "Graph Based Comparison of Executable Objects", University of Technology, Florida, 2005.

[11] W.Andrew et al., "Malware Phylogeny Generation using Permutation of Code", in J.l in Comput. Virology, ACM, 2010, pp 13-23.

[12] G. Bonfante et al., "Architecture of a Morphological Malware Detector", Comput. Virology, 2009, pp. 263-270.

[13] G. Bonfante et al., "Control Flow Graphs as Malware Signatures", 2007.

[14] SeonYoo and Ulrich Ultes-Nitsche, "Towards Establishing a Unknown Virus Detection Technique using SOM", J. in Comput. Virology, 2006, pp.163-186.

[15] M. Christodorescu et al, "Malware normalization", Techn. report, University of Wisconsin, November 2005.

[16] VX Heavens: http://vx.netlux.org

[17] Weka: http://www.cs.waikato.ac.nz/~ml/weka/

[18] V. Satyanarayan et al., "Signature generation and detection of malware families", ACISP 2008.

[19] Fatemeh Karbalaie et al., "Semantic Malware Detection by Deploying Graph Mining", In Proc. of the Student Int. Conf. for IT Security for the next generation, University of HongKong and Kasperky Academy,2010, pp 1020-1025.

[20] Mojtaba Eskandari and Sattar Hashemi, "Metamorphic Malware Detection using Control Flow Graph Mining", Int. J. of Comput. Sci. and Network Security, IJCSNS, vol. 11,12, Dec. 2011.

[21] Kyoochang et al., "Code Graph for Malware Detection", In Int. Conf. on Inform. Netorking, ICOIN'08, 2008, pp 1-5.

[22] Yoshiro Fukushima et al.," A Behavior Based Malware Detection Scheme for Avoiding False Positive", In Proc. of

6th IEEE ICNP Workshop on Secure Network Protocols, (NPSec), Oct., 2010, pp.79-84.

[23] Faraz Ahmed, et al.," Using Spatio-Temporal Information in API Calls with Machine Learning Algorithms for Malware Detection", In Proc. of the 2nd ACM workshop on Security and artificial intelligence, ACM, March 2009, pp 55-62.

[24] Ammar Ahmed E. Elhadi et al.," Malware Detection Based on Hybrid Signature Behaviour Applicat. Programming Interface Call Graph", American J. of Appl. Sci., 2012, pp 283-288.

[25] Ronghua Tian et al.," Differentiating Malware from Cleanware Using Behavioural Analysis", In Proc. of the 3rd Int. Conf. on Security of Inform. and Networks, SIN'10, IEEE, March 2010, pp 23-30.