International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor:6.887
Volume 5 Issue VIII, August 2017- Available at www.ijraset.com

Pragmatic Analysis on Malware Classification

Anjly Chanana[1], Mr Surjeet Singh[2] , Dr. Vikram Bali[3]
[1]Student [2]Assistant Professor, [3]Head of Department [2]Computer Science Department
Panipat Institute of Engineering & Technology & Samalkha

Abstract: Malware are tiny software which are created to harm our computers or to steal valuable information. Recently with the advancement in machine learning algorithms, researches also started looking to a solution which can detect the malware and classify them automatically so that necessary step can be decided. In this paper we discussed and reviewed the methods adopted and suggested by various researchers to detect and classify malwares using machine learning and clustering algorithms. Since this is not a very old field so research papers used to review are not very old.

I. INTRODUCTION

Malware are the malicious codes that are written to perform unethical tasks for example collecting sensitive information, accessing system without authorization from administrator, damaging system resources or data. Malware can be categorized into Virus, Worm, Spyware, Backdoor, Trojan, and Rootkit. The most common types of malware are Virus, Worm and Trojan. Categories of malware:

Virus: It is a type about malware that duplicate it and scatter will different Pcs. It disperses on other PC by attaching themselves of the Different executing code alternately projects the point when client executes the contaminated project. Infection replicates itself Furthermore abandons its infections concerning illustration it move starting with one framework on another. Generally, infection may be not discharged unless client executes those influenced system Yet once it may be discharged it begins recreating itself..

Worm: Worms would projects that use working framework vulnerabilities on spread over PC networks. Worms don't go by attaching itself to an existing bit from claiming code Likewise infection does. Worms could also hold numerous payloads, An bit from claiming code outlined with perform exploitative alternately particular illicit undertaking that destruct host PC alternately organize.

Spyware: Spyware, Concerning illustration the name alludes is those malware that scouts on the client action. The scouting incorporates gathering information, Around others similar to those program history, sites visited, framework alternately account information, saving money points Also money related information. This gathered majority of the data is that point conveyed to malware holders. Spyware doesn't influence host framework Anyhow Concerning illustration it enters the system; it installs itself Also gathers data over foundation Furthermore it remains undetected.

Trojan: A Trojan horse alternately known as Trojan will be a sort of malware that is non-self-replicating. It manipulates itself Likewise an ordinary executable code What's more trap client should download What's more introduce the malware. A Trojan could provide for remote get of contaminated framework to attackers or hackers, which permit hackers should take information, introduce more malware alternately change files.

Backdoor: Backdoor malware permits those unapproved entry from claiming workstation of the attackers same time remain undetected. Those malware about this sort of might have been initiated when multi-user What's more organize working framework might have been began utilizing generally. Backdoors camwood a chance to be made Toward redesigning compiler Furthermore not adjusting those code in front of or then afterward arrangement. The complier camwood be composed done such an approach that An project compiles the code regularly as well as begins backdoor..

Rootkit: Rootkit malware kind may be same Concerning illustration backdoor alternately Trojan such-and-such it tries with get entry of the PC framework without whatever consent alternately majority of the data of the client What's more tries should remain undetected. The rootkit will be unique in relation to Trojan horse Concerning illustration it may be introduced Toward the assailant then afterward getting the get of the PC framework or naturally. Dissimilar to other malware rootkit gets full get of the PC system, it might change or introduce At whatever product. It may be a standalone bit of code that tries to registry data, conceal courses system associations or files. It may be incomprehensible alternately was troublesome on recognize and uproot a rootkit starting with those influenced machine..

©IJRASET (UGC Approved Journal): All Rights are Reserved

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor:6.887
Volume 5 Issue VIII, August 2017- Available at www.ijraset.com

The point about making pernicious codes need been evolving starting with a greater amount broad should additional complex publicizing by focusing on touchy data for example, MasterCard information, passwords Also bank points which need settled on malware examination really vital. Concerning illustration the malware makes critical reduction of incredulous information What's more at times settle on harm of the network, the malware identification need get a standout amongst the The greater part basic issues in the field from claiming machine security. Thus, in place with recognize malware efficiently, malware examination assumes a paramount part.. Malware Analysis Techniques Malware analysis can be broadly classified into static and dynamic technique.

Dynamic Malware Analysis Technique: Dynamic malware analysis technique is also known as behavioural analysis as it inspects the behaviour of malware by executing the binary code in the controlled environment. Behavioural analysis is quicker way of malware analysis as while doing analysis if malware is not provided the acceptable environment then there are more chances that analyst will miss the characteristics of malware. Behaviour or dynamic analysis can be further divided into two types as basic and advanced. Basic malware analysis provides suitable environment to malware, inspect their execution and collect all the information related to their runtime behaviour. This information can be related to API or system calls, files added, removed or modified, new services installed changes in the processes or registry files or any modifications in the system settings. Advanced behaviour analysis also requires knowledge of windows internals and specific programming. Analysts can load binary code into the debugger tools such as ollydbg or windbg and run malware code line by line and monitor its activity.

Static Malware Analysis Technique: Static malware analysis technique is performed by analyzing code statically without running the sample using tools such as PE Viewer, CFF explorer and more. Thus it is also known as the code analysis. This technique is safer than dynamic malware analysis technique as malware are not executed. If the malware is packed then analysis cannot be performed on it without unpacking the malware. Code analysis technique can also be further categorised into basic and advanced categories. Basic code analysis technique is not very efficient but it is easy and very quick. The goal of the static analysis is to classify the sample into benign and malicious executable code without understanding the capabilities of samples. It does not require checking the actual instructions of the sample. Basic code analysis includes identifying if antivirus detects any sample, sample is packed or unpacked, its version information, any suspicious imports by executable code or if PE field format is malformed. Advanced code analysis requires knowledge of windows internals, Assembly language and compiler code. In this type of analysis analysts are required to load the binary code into disassembler such as IDAPro to perform reverse engineering and completely analyze the executables. After performing reverse engineering analysts will understand how the code works or how malware infects system, which in turn will help to reduce infection and help to create better security and defence software. This is the most effective technique.

II. LITERATURE REVIEW

A lot of previous work is done on detection and classification of malware. This chapter discusses some attempts made for malware classification, which include usage of structured control workflow and some other data mining methods.

A. Classification of Malware using Structured Control Flow

Control flow represents the execution path that a program may take. Control flow information appears in two forms. The call graph represents the inter-procedural control flow. The intra-procedural control flow is represented as a set of control flow graphs with one graph per procedure [9]. In [9], the research has shown that malware can be efficiently characterized by its control flow. The authors have proposed a malware classification method using approximate matching of control flow graphs. The string edit distances can be calculated between the control flow signatures and the structured graphs of the malware in the database. The threshold is predefined. If the edit distance is more than the particular threshold, then the binary code can be classified as a malicious binary code, otherwise it is a benign binary code. Control flow is more invariant among metamorphic and polymorphic malware. The research shows that the proposed system could successfully identify different variants of malware [9].

B. Behavioural Malware Classification

Classification systems generally categorised into one of two categories: Those that focus on features extracted from static files, or those that execute malware and use their behavioural feature to classify malware. Static approaches sometimes use low-level features such as calls to external strings, libraries, and byte sequences for classification [23]. Other static approaches extract more information from binary codes, including the graphical representations of control flow [9], sequences of API calls. Although the

©IJRASET (UGC Approved Journal): All Rights are Reserved

variants in a malware family have different static signatures, they share typical behavioural patterns resulting from their common function and heritage [8]. The authors in [8] have described preprogramed classification system that can be trained to correctly identify new mutant within known malware families, using perceived similarities in behavioural features extracted from sensors monitoring live computer hosts. In feature selection used in [8], the authors have selected a set of observable features that can be simply extracted from live computer hosts, and whose properties can be used to deduce whether a detected malware sample belongs to particular category or family. The authors have examined the following features: the frequency of calls to specific kernel functions, the data collected from performance monitors that report resource usage and the frequency of calls to specific sequences of kernel functions. The method proposed by authors uses decision trees for classification, which label each newly identified malware sample as one of the existing labels learned during training. If a new malware sample does not linked to one of the existing labels, the decision trees label it (incorrectly) with one of the existing labels. To address the issue of new malware samples that are not well-defined by an existing label, the authors have used different types of classifiers. They have evaluated the use of the nearest centroid algorithm [49], which can be used to indicate a notion of distance from the centroids that define the labels. If the malware whose distance to the closest centroid exceeded the threshold, it might be considered new a malware and required further analysis. The authors uses k-means clustering algorithm [7] to handle the behavioural diversity of malware having a specific label to establish multiple centroids for distinguishing each malware label and also investigating the use of subfamily labels for classification. The presented results shows that the behavioural classifier can accurately identify new malware variants within specific families of malware.

C. Malware Classification using the Data Mining Methods

In [23], the author has extracted the byte sequences from the executables, converting these into n-grams, and constructed several classifiers: instance-based learner, decision trees, Na¨ıve Bayes, support vector machines and boosting. They viewed each n-grams as a Boolean attribute that is either present in (i.e., T) or absent from (i.e., F) the executable. They have shown that the boosted decision trees outperformed the other methods. The following section shows the methods used in their research.

1) *Instance-Based Learner :* One of the simplest learning methods is the instance-based (IB) learner [1]. Its concept description is a collection of training examples or instances. Learning, therefore, is the addition of new examples to the collection. An instance is found in the collection which is most similar to unknown and the instances class label is returned as its prediction for unknown. The authors have used number of values the two instances have in common as the measure of similarity. In the variation of this method, such as instance-based (IB) learner, the k most similar examples are found and majority vote of their class labels are returned as the prediction. Values for k are typically odd to prevent ties [23]. These are also called as nearest neighbour and k-nearest neighbours.
2) *Support Vector Machines (SVM) :* Support Vector Machine [12] is supervised learning model with associated learning algorithms which analyze data and recognize patterns, used for classification. In [23], the authors produces a linear classifier, thus its concept description is a vector of weights, ~wg and a threshold, t. SVMs use a kernel function to map a training data into a higher-dimensioned space such that problem is linearly separable. In high dimensional space, hyper plane is build and used for classification. The method predicts the positive class if ~wg.~x−t >0 and otherwise it predicts the negative class [23]. ~x is set of points on the hyper plane. Thus, given a set of training instances belonging to one of the two categories, an SVM assigns new instances into one category or other.
3) *Na¨ive Bayes :* Na¨ıve Bayes is a probabilistic method which has long history in information recapture and text classification [29]. It stores the prior probability of each class as its concept description, P(Ci), and the conditional probability of each attribute value of given the class, P(vj |Ci). These quantities are calculated by counting in training data frequency of occurrence of the classes and the attribute values of each class. The Bayes rule is used to estimate posterior probability of each class given an unknown example, returning as its prediction class with highest value [23].
4) *Decision Trees :* The decision trees are built based on training data. The internal nodes of decision tree correspond to the attributes and leaf nodes correspond to the class labels. The performance elements use attributes and their values of an example to traverse tree from the root to leaf. It predicts class label of the leaf node [23]. It creates node, branches, and children for attribute and its values, remove the attribute from the further consideration, and distribute the instances to the suitable child node. This process repeats recursively until a node contains instances of the same class, at that point, it stores the class label [23]. In the effort to reduce over training data, most implementations also prune induced decision trees by removing the sub

©IJRASET (UGC Approved Journal): All Rights are Reserved

trees that are likely to perform imperfectly on test data [23]. The malware classification based on decision trees is very fast and accurate. The disadvantage of decision trees is that an error in higher level of the tree may cause an error in the lower part of tree [48].

5) *Boosted Classifiers:* Boosting [15] is a method of merging multiple classifiers. A set of weighted models are produced by iterative learning of a model from a weighted dataset. The generated model is then analysed. The dataset is reweighted based upon the performance of model [23]. The authors have provided a method for detecting unknown malicious code in executable code using machine learning. They extracted byte sequences from the executable code , converted these sequences into n-grams, and then constructed several classi - fiers: na¨ıve Bayes, boosted decision trees and boosted SVMs. The results of their experiments have shown that the boosted decision trees outperformed other methods and achieved a true-positive rate of 0.98 and a false-positive rate of 0.05.

D. VILO: A Rapid Learning Nearest-Neighbour Classifier for Malware Classification

There are two different types of malware classification possible: familial and binary: In binary malware classification problem, an unknown executable code is classified as either being benign or malicious. Conversely, in familial malware classification problem, a malicious executable code is classified as belonging to a specific group of malware [25]. In [25], the work is based on the familial malware classification. VILO makes use of three components: N-perm feature vectors, Term Frequency X Inverse Document Frequency (TFIDF) weighting of features [20], and nearest-neighbour algorithm. N-perms are obtained by sliding the window of size s over bytes, opcodes of n-grams. They are robust against some code obfuscations like instruction reordering [11]. VILO implements the nearest neighbour algorithm with similarities evalueated over TFIDF weighted opcode mnemonic permutation features (N-perms). The results in [25] showed that VILO is the quick and efficacious learner of the real-world malware. TFIDF weighting of features ensures that the features that are common across many categories of executable code are not overly emphasized [25]. This is also suitable for constantly changing malware population. Nearest neighbour search does not require any construction of the classification model and hence it is very effective and simple[25]. The authors have also stated that VILO is not acceptable for binary malware classification.

E. K means Clustering

Clustering is a in which process of objects are categorised into large number of subsets based upon similarity in the context of a particular problem. The objects square measure thereby organized into a illustration that characterizes the gathering being sampled. The following section discusses different methods of clustering [19].

F. Exclusive versus Non-Exclusive

An exclusive classification is that the partition of set of objects. every object forever belongs to precisely one set, or cluster. The Nonexclusive classification, conjointly referred to as overlapping classification, is another within which associate object is assigned to the many clusters [19].

G. Intrinsic versus Extrinsic

Depending on whether or not class labels are gift on the objects being classified, there are 2 classes (intrinsic and extrinsic classification).Intrinsic classification uses a proximity matrix to perform classification. Intrinsic classification is additionally referred to as “unsupervised learning” as a result of no labels are used for the classifying the item [19]. Extrinsic classification uses the category labels on the object [19]. Extrinsic classification is also called “supervised learning.”

H. Hierarchical versus Partitioned

Hierarchical bunch algorithmic rule divide knowledge into the hierarchy of clusters, whereas partitioned algorithmic rule divide knowledge set into reciprocally disjoint partitions [18]. Hierarchical clustering algorithm produces the hierarchy of clusters called dendrogram either by dividing larger clusters to smaller ones or merging smaller clusters into larger ones [18]. Hierarchical clustering algorithms are further divided into following two categories:

I. Agglomerative

©IJRASET (UGC Approved Journal): All Rights are Reserved

Initially, each point is considered as a cluster. The two "nearest" clusters are combined into one cluster repeatedly until all clusters are merged into the single cluster . This is a "bottom up" approach.

J. Divisive

All observations start from one cluster, and splits are performed recursively as moves down the hierarchy. This is "top down" approach.

Partitioned clustering algorithm generates various partitions and then computes them by some pre - specified criterion [18]. They are also called non-hierarchical as each of the data point is placed in one of k mutually exclusive clusters. One of the most popular clustering algorithms is k-means clustering algorithm. The number of clusters k is pre-decided. The algorithm initializes centroids for k partitions, and then assigns each member to the cluster based on its distance from each centroid. It then recomputed the centroids based on newly formed clusters. This process is repeated until the difference between the initial and the recomputed centroids is negligible. There are key differences between the hierarchical and the partitioned clustering [18]:

1) Partitioned clustering is faster than the hierarchical clustering.

2) Hierarchical clustering requires only the similarity measure, whereas the partitioned clustering requires the initial centres/centroids and the number of clusters.

a) *k-Means Clustering Algorithm :* In this project, k-means clustering algorithm is used for classification of malware. This algorithm is one of the simplest unsupervised learning algorithms that resolves the clustering problem [27]. It classifies dataset into specific number of clusters (say, k), which is pre-decided. k centroids are defined at the time of initialization, one for each cluster. These centroids are placed as far as possible from each other. Once this initialization is completed, each of the data point in the dataset is associated with centroid "nearest" to it and placed in corresponding cluster. In the next step, centroids for each cluster are recomputed depending upon the data points that have been assigned to it. These steps are repeated until the centroids no longer change, or until distance between the initial centroids and the recomputed centroids is negligible. The Euclidean distance formula is used for distances calculation. The algorithm used in this project can be described in 7 steps. Steps 1 and 2 contains of the pre-work. Step 3 is the initialization step. Steps 4, 5 and 6 are the looping steps. Step 6 is the analysis step.

3) Collect malware dataset.

4) Identify number of clusters (k).

5) random distribution on each dimension of the centroid or by selecting one of the data points as a centroid.

6) Determine distance of each malware from each centroid using the Euclidean distance, and then assign each malware to the cluster with centroid closest to it.

7) Recompute the centroids for each cluster.

8) Repeat the steps 4 and 5 until there is no change in cluster centroids

9) If formed clusters do not look reasonable, repeat the steps 1-6 for different number of clusters.

a) *Advantages of the K-Means Clustering Algorithm :* The k-means clustering algorithm is simple to implement. Since the distance of the point from centroid does not depend on the distance of any other point from any centroid, these calculations can be performed in parallel, thus increasing the overall speed of execution. Other advantages of k-means clustering algorithm are as follow: 1. It is a intuitive and simple clustering algorithm. 2. This algorithm also works well for globular clusters.

b) *Disadvantages of the K-Means Clustering Algorithm :* Although k-means clustering algorithm is fast and simple and produce tighter clusters than hierarchical clustering, it has some limitations. Following are the limitations of k-means clustering algorithm:

a) The number of clusters k is required to be defined at the beginning. If this k is incorrectly assumed, the clusters may not reflect the natural classification of the data.

b) Different initial values of the centroids may produce different clusters. It may therefore be necessary to try clustering with different random initializations for the cluster centroids.

c) k means does not work well with non-globular clusters.

©IJRASET (UGC Approved Journal): All Rights are Reserved

III. CONCLUSION

This paper provides a review to previously used algorithms to classify the malwares. It has been noticed that many researchers used machine learning concepts which requires supervised learning which means a collected database of previous attacks of malwares on system and information about their type is required to built or suggest our own algorithm. K-means algorithm is widely used recently along with machine learning algorithms to classify the type, but it suffers from the drawback of uncertain cluster heads i.e. how many certain type of malware can be present in the provided data. In our next paper we will focus on this problem.

REFERENCES

[1] D. Aha, D. Kibler, and M. Albert (1991). Instance-Based Learning Algorithms. Machine Learning, 6, 37–66.
[2] learning. Artificial Intelligence Research, 1(1).
[3] T. Austin, E. Filiol, S. Josse, and M. Stamp (2013). Exploring Hidden Markov Models for Virus Analysis: A Semantic Approach. System Sciences (HICSS), 2013 46th Hawaii International Conference, 5039–5048.
[4] J. Aycock (2006). Computer Viruses and Malware. Fairfax, VA: Springer-Verlag New York.
[5] J. Baltazar, J. Costoya, and R.Flores (n.d.). The Real Face of KOOBFACE: The Largest Web 2.0 Botnet Explained. http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/ white-papers/wp_the-real-face-of-koobface.pdf
[6] J. Bezdek, R. Ehrlich, and W. Full (1984). FCM: The Fuzzy c-Means Clustering Algorithm. Computers & Geosciences, 10(2-3), 191–203
[7] C. Bishop. (2006). Pattern Recognition and Machine Learning. Springer.
[8] R. Canzanese, M. Kam, and S. Mancoridis (n.d.). Toward an Automatic, Online Behavioral Malware Classifiction System.
[9] S. Cesare and Y. Xiang (2010). Classification of Malware Using Structured Control Flow. 8th Australasian Symposium on Parallel and Distributed Computing, 107, 61–70.
[10] K. Chen (n.d.). A constant Factor Approximation Algorithm for K-median Clustering with Outliers. http://faculty.cs.tamu.edu/chen/courses/cpsc669/2009/reading/xu1.pdf
[11] F. Cohen (1993). Operating system protection through program evolution. Computer & Security, 12(6), 565–584.
[12] C. Cortes and V. Vapnik(1995). Support-Vector Networks. Machine Learning, 20, 273–297. AT&T Bell Labs, Holmdel, NJ
[13] S. Deshpande (2012). Eigenvalue Analysis for Metamorphic Detection, Masters Thesis, San Jose State University.
[14] Difference between a computer virus and a computer worm (n.d.). http://scienceline.ucsb.edu/getkey.php?key=52
[15] Y. Freund and R. Schapire (1999). A Short Introduction to Boosting. Journal of Japanese Society for Artificial Intelligence, 14(5), 771–780.
[16] F. Howard (2012). Exploring the Blackhole exploit kit: 4.1 Distribution of web threats. http://nakedsecurity.sophos.com/exploring-the-blackhole-exploit-kit-14/
[17] N. Idika and A. Mathur (2007). A Survey of Malware Detection Techniques. http://www.flickr.com/photos/panda_security/5198720136/lightbox
[18] Indika (2011). Difference between Hierarchical and Partitional Clustering. http://www.differencebetween.com/difference-between-hierarchical-and-vs-partitional-clustering
[19] A. Jain and R. Dubes (1988). Algorithms for Clustering Data. Englewood Cliffs, New Jersey: Prentice Hall. \
[20] K. Jones (1972). A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation, 28(1), 11–21.
[21] L. Kaufman and P. Rousseeuw (1987). Clustering by means of Medoids, edited by Y. Dodge. Statistical Data Analysis Based on the L1-Norm and Related Methods, 405–416. North Holland.
[22] S. Kazi and M. Stamp (2013). Hidden Markov Models for software piracy detection. Information Security Journal: A Global Perspective, 22(3), 140–149.
[23] S. Kolter and M. Maloof (2006). Learning to Detect and Classify Malicious Executables in the Wild. Journal of Machine Learning Research, 7, 2721–2744.
[24] A. Krogh (1998). An Introduction to hidden Markov models for biological sequences. Computational Methods in Molecular Biology, 45–63. Lyngby, Denmark: Elsevier.
[25] A. Lakhotia, A. Walenstein, C. Miles, and A. Singh (2013). VILO: a rapid learning nearest-neighbor classifier for malware traige. Journal in Computer Virology, 9(3), 109–123. Secaucus, NJ: Springer-Verlag.
[26] M. Landesman (n.d.). Boot sector virus repair.http://antivirus.about.com/od/securitytips/a/bootsectorvirus.htm
[27] J. MacQueen (1967). Some Methods for Classification and Analysis of Multivariate Observations. Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, 1. 281–297. Berkeley, University of California.
[28] Malware Protection Center (n.d.). Win32/Zbot. http://www.microsoft.com/security/portal/ threat/encyclopedia/entry.aspx?Name=Win32%2fZbot
[29] M. Maron and J. Kuhns (1960). On relevance, probabilistic indexing, and information retrieval. Journal of the Association for Computing Machinery, 7, 216–244.
[30] P. Mullins (n.d.). Malware. http://cs.sru.edu/~mullins/cpsc100book/ module05_SoftwareAndAdmin/module05-04_softwareAndAdmin.html
[31] A. Nappa, M. Zubair Rafique, and J. Caballero (2013). Driving in the Cloud: An Analysis of Drive-by Download Operations and Abuse Reporting of viruses. Proceedings of the 10th Conference on Detection of Intrusions and Malware & Vulnerability Assessment. Berlin, DE.
[32] Panda Security (n.d.). Virus, worms, trojans and backdoors: Other harmful relatives of viruses.http://www.pandasecurity.com/homeusers-cms3/security-info/about malware/generalconcepts/concept-2.htm
[33] Panda Security (n.d.). Malware Evolution. http://www.flickr.com/photos/panda_security/5198720136/lightbox
[34] S. Priyadarshani (2011). Metamorphic Detection via Emulation, Masters Thesis, San Jose State University. http://scholarworks.sjsu.edu/cgi/viewcontent.cgi?article=1176&context=etd_projects
[35] L. Rabiner (1989). A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE, 77(2), 257–286.
[36] N. Runwal, R. Low, and M. Stamp (2012). Opcode graph similarity and metamorphic detection. Journal in Computer Virology, 8, 37–52.
[37] M. Stamp (2011). Information Security: Principles and Practice, second edition, Hoboken, NJ: Wiley-Interscience.

©IJRASET (UGC Approved Journal): All Rights are Reserved

[38] M. Stamp (2012). A Revealing Introduction to Hidden Markov Models. http://www.cs.sjsu.edu/faculty/stamp/RUA/HMM.pdf
[39] Symantec (2010). Trojan.Zbot. http://www.symantec.com/security_response/writeup.jsp?docid=2010-011016-3514-99
[40] Symantec (2009). What is the difference between viruses, worms, and Trojans? http://www.symantec.com/business/support/index?page=content&id=TECH98539
[41] Symantec (2004). Trojan Horse. http://www.symantec.com/security_response/writeup.jsp?docid=2004-021914-2822-99
[42] Symantec Security Response (2011). Trojan.Zeroaccess. http://www.symantec.com/security_response/writeup.jsp?docid=2011-071314-0410-99
[43] A. Vasudevan. (2008). MalTRAK: Tracking and Eliminating Unknown Malware. Computer Security Applications Conference 311-321.
[44] N. Villeneuve with a foreword by R. Deibert &R. Rohozinski (2010). KOOBFACE: Inside a Crimeware Network. http://www.trendmicro.com/cloud-content/us/pdfs/ security-intelligence/white-papers/wp_the-real-face-of-koobface.pdf
[45] Virus Bulletin (n.d.). Last-minute paper: An indepth look into Stuxnet. http://www.virusbtn.com/conference/vb2010/abstracts/LastMinute7.xml
[46] Virus Removal Services (n.d.). Beware of FAKE Antivirus - Winwebsec. http://virus.myfirstattempt.com/2012/11/beware-of-fake-anti-virus-winwebsec.html
[47] W. Wong and M. Stamp (2006). Hunting for metamorphic engines. Journal in Computer Virology. 2(3), 211–229.
[48] M. Yusoff and A. Jantan (2011). Optimizing Decision Tree in Malware Classifi- cation System by using Genetic Algorithm. http://sdiwc.net/digital-library/web-admin/upload-pdf/00000060.pdf \
[49] Q. Zhang and S. Sun (2012). A centroid k-nearest neighbor method. ADMA'10 Proceedings of the 6th international conference on Advanced data mining and applications: Part I, 278–285. Berlin, Heidelberg: Springer-Verlag.

©IJRASET (UGC Approved Journal): All Rights are Reserved