I.J. Modern Education and Computer Science, **2017, 3, 48-55**
Published Online March 2017 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijmecs.2017.03.06

Dynamic Malware Analysis and Detection in Virtual Environment

Akshatha Sujyothi
Department of Computer Science, St Joseph Engineering College, Mangaluru, 575028, India
Email: sujyothi28@gmail.com

Shreenath Acharya
Department of Computer Science, St Joseph Engineering College, Mangaluru, 575028, India
Email: shree.katapady@gmail.com

***Abstract*—**The amount and the complexity of malicious activity increasing and evolving day by day. Typical static code analysis is futile when challenged by diverse variants. The prolog of new malware samples every day is not uncommon and the malware designed by the attackers have the ability to change as they propagate. Thus, automated dynamic malware analysis becomes a widely preferred technique for the identification of unknown malware.

In this paper, an automated malware detection system is presented based on dynamic malware analysis approach. The behavior of malware is observed in the controlled environment of the popular malware analysis system. It uses the clustering and classification of embedded malware behavior reports to identify the presence of malicious behavior. Based on the experimentation and evaluation it is evident that the proposed system is able to achieve better F-measures, FPR, FNR, TPR and TNR values resulting in accurate classification leading to more efficient detection of unknown malware compared to the traditional hierarchical classification approach.

***Index Terms*—**Static code analysis, malware, dynamic malware analysis, clustering, classification

I. Introduction

Malware threat is a kind of code designed with dangerous intentions. The major role of the malware is targeting to the privacy of users and their information which made the use of malicious activities continuously evolving. However, the high-priority threat is posed by the security researchers.

The widely deployed traditional approaches such as signature-based and static malware analysis require manual inspection of the malicious code by the human analysts. Unfortunately, tremendous growth in the generation of malicious code led the antivirus vendors to face thousands of new malware files every day.

The analysis of large volume of files by this technique evaded due to obfuscation, polymorphism etc. Hence, a reliable and automated analysis is an important point to be able to cope with this threat. With the increase in readily available and sophisticated tools, use of the new generation cyber threats/attacks is becoming more targeted, persistent and unknown. The advanced malware’s are targeted, unknown, stealthy, personalized and zero days as compared to the traditional malware which were broad, known, open and one time. Once inside, they hide, replicate and disable host protections. After getting installed, they call their command and control servers for further instructions, which could be to steal data, infect other machines, and allow reconnaissance.

The Antivirus (AV) software was introduced with an intention to prevent, detect and remove malicious software. This effort mainly focused on content-based signatures to automatically classify and analyze malware samples. Unfortunately, these techniques are essentially susceptible to inaccuracies due to polymorphic and metamorphic techniques. As a result, the Intrusion Detection System and AV are failed achieve complete success in malware analysis. Therefore, dynamic malware analysis tools are widely used to automate and classification malware and benign samples. Dynamic malware analysis involves execution of binaries in the controlled environment to extract the required information for the detection of malicious behavior.

Cloud infrastructure has been growing trend for years by providing opportunities to scale, flexibility and efficiency. Even though cloud emerging in the current scenario but there exist crucial attacks by the hackers such as VM holes and cloud specific threats related to the environment. Providing security and reliability play an important role in the virtualized environment. There are plenty of studies are taking place in the virtualized environment which has the main cause on as networks, Virtual machine manager, guest virtual machines and Operating System (OS).

Cloud data centers are used for a range of always-on services such as private, public and commercial domains. These need to be secure and resilient in terms of cyber attacks as well as component failures and misconfiguration. However, clouds have impaired the traditional detection system due to its characteristics and intrinsic operational research structures. The limitations of a priori attack signature and payload information are overcome by making use of dynamic features such as per-flow meta-statistics as derived from the packet header and volumetric information (i.e. counts of packets, bytes, etc.). In this way, dynamic analysis plays an important role in cloud infrastructure.

Against this background, here in this paper, a framework is proposed for malware behavior analysis based on virtualization techniques. The detection of malware files in the cloud computing environment using the machine learning approaches is achieved. The main contribution is in the decrement of the False Positive rate which incorrectly classifies malicious files.

The paper is organized as follows: Section II describes the related works with the specifications of system vulnerabilities and malwares in virtualized environment, followed by the System Architecture in section III. Section IV depicts the implementation of the system with detailed descriptions about each and every module by the usage of clustering and classifications algorithm. Section V presents the experiments, results and evaluations followed by the conclusion in section VI.

II. Related Works

Malware authors have generated several ways to detect the presence of malware analysis systems but dynamic malware analysis overcome most of the hackers techniques such as evasion, obfuscation, polymorphism etc.

Several online tools available today are relying on dynamic analysis techniques that generate reports which are the human understandable format. The analysis system is required to have an appropriate representation for malware, which is then used for classification either based on similarity measure or feature vectors. However, a large number of new malware samples arriving at anti-virus vendors every day require an automated approach so as to limit the number of samples that require close human analysis. Several Artificial Intelligence techniques, particularly machine-learning based techniques have been used in the literature for automated malware analysis and classification. The different works carried out by the authors are discussed below.

Watson et al. [1] pointed out an online cloud anomaly detection system with the usage of one class support vector machine. One class SVM formulation is used at the hypervisor level with utilizing features at both system and network level. Security and resilience are major roles in the cloud infrastructure hence cloud should be able to react to the unknown threats generating in the real world. The proposed method involves anomaly detection methods to overcome the detection of unknown threats. The novel approach used here is the one class support vector machine helps in utilization of feature to identify the malicious aspect at the hypervisor level. The virtual machine used in the experiment uses per VM methods to perform the analysis which helps in detection of malwares.

Bayer et al. [2] presented a scalable clustering approach that identifies and clusters malware. Extended ANUBIS system is used with taint tracking for analyzing the behavior. The limiting factor of this approach is trace dependence. Another issue is dynamic data tainting; so the malicious binary could be injected. Some malware triggers only during some actions or events such as setting up time to explore etc. these forms major limitation for this framework. The evasion techniques are also not avoidable in this approach which is again a disadvantage.

Syarif et al. [3] pointed out the analysis of static and dynamic malware approaches. The static method is achieved without running system whereas dynamic while running. This paper highlights the major pros and cons of each technique with the different usage of it.

The static approach uses manual inspection which fails due to evasion, obfuscation and polymorphism techniques. Meanwhile, advanced static approach provides better result with the more specific details about the malicious programs. The basic dynamic analysis gives DLL infections to the system which basically gives the information about the actions of malware. Whereas, advanced dynamic malware analysis approach gives more severe actions made by the malicious code in the system such as turning off firewall etc. The integration of these two approaches results in the better solution to the detection of malware activities.

Lorenzo et al. [4] proposed a behavior-based framework for malware analysis. It improves behavior based analysis for suspicious programs allowing end-users to get the secure environment which makes them feel executing directly on the former environment. The system calls are used between security lab and potential victim programs. The behavior of malware is analyzed here.

Lakshmanan et al. [5] implemented a framework by using texture analysis method which is resilient to packing strategies and robustly classify large malware corpus. The method is flexible to packing services and classifies malware robustly.

The texture analysis is based on the images obtained during the execution of malicious codes. These images are used to represent the behavior of the malware used in the similarity measures which is useful in the process of malware classification. Compared to all the other existing dynamic approaches related to the malware analysis the texture analysis gives best result. The limitations are due to the dissimilarities of images within single family and variants of same malware.

Dilung et al. [6] presented a comparative framework with the introduction of BareCloud concept which increases the rate of detecting evasive malware but further improvement can be achieved introducing richer filesystem-level event traces.

The BareCloud make use of hierarchical clustering approaches which bitterly classifies malwares into different clusters based on the features selected. The experimental result gives the greater performance as compared to the other methods and techniques. Testing over the large number of malware samples helped to differentiate the performance and accuracy amongst other readily available tools.

Dilung et al. [7] developed an automated technique called MALGENE for extracting analysis evasion signature. Malgene makes use of bioinformatics algorithms which locates evasive behavior automatically in the form of system calls.

Data mining approaches are used to determine the data events and call events to identify the behavior of the malware. These results are helpful in the construction of evasion signature, which is useful in the analysis of malware. System call alignment with parameter selection is major methods used in this approach. The experiment is carried out with different levels such as hypervisor and emulation.

Philip O'Kane, et al. [8] proposed approach for detecting malware using the N-gram analysis techniques. It uses the structure of the program by considering the characters, strings and bytes. The experiment represented in this paper uses operational code (opcode) density histogram obtained during the dynamic analysis. A reference model is created using support vector machine that is used to evaluate feature reduction methods. However, the connections between features are complex therefore usage of simple filtering approaches is not viable. To provide better result the experiment uses Eigen subspace analysis method.

Konrad et al. [9] proposed an automated analysis framework which makes use of prototype based feature vectors to cluster and classify the malware samples. The main contribution from the author is to detect unknown malware with reduced runtime. The proposed framework involves clustering, classification and incremental analysis to perform the dynamic malware analysis. As mentioned earlier most of the paper had the limitations with larger dataset. Here, the usage of incremental analysis reduced the runtime for the larger amount of data.

Vulnerabilities

Overviews of vulnerabilities which are commonly used by the malware are given in this section. Malware mainly exploits vulnerabilities in operating system design, in applications or in versions of browser plugins. The insecure design of the system can lead to the weak points wherein hackers can able to insert their code easily. Shared Resources and services are provided by the server to clients in the network which is also a point to allow attacks such as denial of services etc.

Malware's in Virtualized Environment

Virtualization has been a growing trend for years, offering opportunities for scaling, efficiency and flexibility. Despite the end users benefitted by cloud services it also comes with different threats such as holes on virtual machines (e.g. rootkit attacks on virtual machines); mutated cloud- specific Internet-based attacks that aim to compromise cloud networks (e.g. malware and DDoS attacks on cloud services). Therefore, security and resilience are playing important roles. A number of studies have addressed aspects of cloud security [16] [17] [18] at different levels such as network, hypervisor, guest VM and Operating System (OS).

III. System Architecture

The main goal of the proposed system is to detect malware in an automated way by reducing the rate of false identification of malware. The use of dynamic malware analysis is to detect the presence of malicious behavior. The system firstly captures the execution traces, thereby identifying the malicious trace.

The architectural view of the proposed system is depicted in the fig. 1 which mainly consists of analysis module and the classification module. Analysis module mainly does the preprocessing tasks such as generating data suitable for the classifier tool. Whereas, classification modules deal with the preprocessed data to perform the proper classification in order to differentiate each sample into its corresponding neighbors.

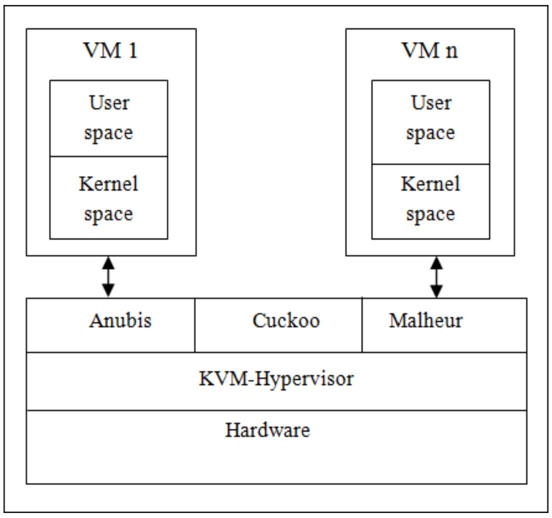


Fig.1. Basic System Architecture

The main goal of the proposed system is to detect malware in an automated way by reducing the rate of false identification of malware. The use of dynamic malware analysis is to detect the presence of malicious behavior. The system firstly captures the execution traces, thereby identifying the malicious trace. The architectural view of the proposed system is depicted in the fig. 1 which mainly consists of analysis module and the classification module. Analysis module mainly does the preprocessing tasks such as generating data suitable for the classifier tool. Whereas, classification modules deal with the preprocessed data to perform the proper classification in order to differentiate each sample into its corresponding neighbors.

A. *Analysis Module*

The preprocessing of malware analysis is the execution of malicious traces in the instrumented environment. Here, wild used frameworks Anubis and Cuckoo sandbox are used to generate the behavioral reports of the malware and benign samples. These frameworks run on top of the KVM-hypervisor to avoid causes of execution to the host machines. The purpose of preprocessing step is to select the most interesting samples which are likely to have malevolent behavior. These preprocessed malware samples are then fed to the dynamic malware analysis tools used to identify the presence of malicious behaviors.

B. *Classification Module*

Malware identification is achieved through the classification of malware which is done using the Malheur tool. The reliable and accurate classification of malware will result in the effective detection of malware with the use of novel techniques described in the further sections.

IV. Implementation

The purpose of protection and resilience plays a major role to make use of dynamic malware analysis in cloud computing environment. To identify the presence of malicious code the sample must be executed and traced out the features which match the characteristics of malware samples. The system uses following methodologies to carry out the experiment.

A. *Collection Of Samples*

A set of malware binaries is collected from large malware repository namely Virusshare [10]. The input the framework is grouped into the training dataset and testing dataset. A number of known samples are listed under training dataset which is used to train the machine to understand the features of the malicious code. The testing dataset consists of unknown samples which have to be classified according to the malware characteristic.

B. *Generated Report*

Once the sample is labeled using some antivirus vendor such as virustotal initially known samples are executed in the sandbox to extract the behavior of the malware. The execution trace of the malicious code is captured with respect to system, network, file, registry activities. The generated report with all these information will be useful in the classification and detection of malware.

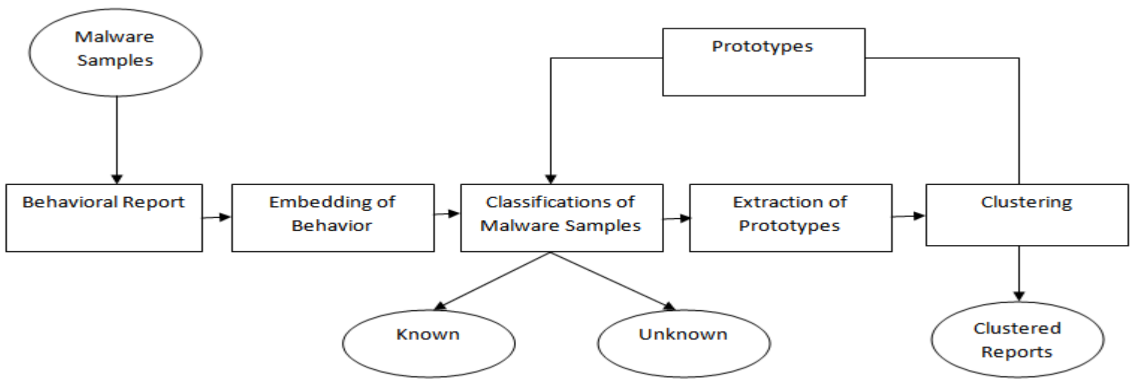


Fig.2. Flow diagram showing overall activities of the system

C. *Embedding behavior*

The raw behavioral report is not suitable for the classification process hence it must be converted to the effective format which reduces the load to the algorithms. Firstly, the system calls are converted to the MIST [19] format which consists of system call with its corresponding attribute values. MIST enables a significant characterization of behavior. Yet, this representation is inappropriate for the efficient analysis of the malware samples, as these works on vectorial representation.

The embedding of a report into vector form with instruction q-grams involves set of instruction with a fixed length of instruction pattern. It can be best explained with (1).

$$S = \{(a_1, . . . , a_q) \mid a_i \in A \text{ with } 1 \leq i \geq q\} \tag{1}$$

Where, S denotes the set of all q-grams and A denotes the set of all possible instructions. The measures of A and S may vary based on the different MIST level. A report can be embedded in the Vector space |S| making use of embedded function which acts as a pointer to the q-gram in the instruction set. Equation (2) represents the embedding function which is useful in the embedding of reports.

$$\varphi(x) = (\varphi_s(x))_{s \in S} \text{ with } \varphi_s(x) = \begin{cases} 1 & if\ report\ x\ contains\ q-grams \\ 0 & otherwise \end{cases} \tag{2}$$

D. *Distance Matrix*

The embedded reports are compared with each other to compute distance matrix using Euclidian Distance formula. It compares behaviors of reports and the value ranges from 0 to $\sqrt{2}$ where 0 represents similar behavior.

E. *Clustering and Classification*

The conduct of the produced report is separated and utilized for the further steps, for example, clustering strategy. Machine learning methodologies are helpful in the analysis of behavioral reports including the methods such as "*clustering of behavior*" and "*classification of behavior*".

Clustering and classification for analysis of malicious behavior with the use of hierarchical clustering and SVM approach [9] are used. Initially, a report is chosen as a prototype, randomly or fixed. Using this prototype the malware embedded reports are clustered. The clustering of embedded reports partitions the given data into meaningful groups namely the clusters. Every cluster split a common characteristic for each object within it whereas it varies from other clusters. It helps in the finding of the characteristics to the unknown malware data for the larger amount of samples.

The clustering algorithm mentioned here represents the flow of control between the prototype generation and comparison with the embedded reports. The initial point of taking one report as a prototype and comparing with the other reports, results in the nearest neighbors having similar values together. However, this helps in the generation of clusters.

Algorithm for Clustering:

```
1. Initialize by considering each prototype as a
   cluster
2. Iteratively find and merge the close pair of
   clusters
3. While (min(distance) < $d_c$) (i.e., the Maximum
   distance between Individuals)
   do
      Cluster the nearest prototypes
      Update distance using complete linkage
4. for (x ∈ reports )
    do
       Assign nearest prototypes to x
       Reject Clusters with lesser than
              minimum members
```

The machine learning concept helps system to learn the features of the malware at the time of clustering. The traces of characteristics are saved and used for the classification method. The procedure is used further in the detection of unknown classes of malware by making use of the prototypes obtained in the preliminary steps.

Classification of unknown malware will require proper identifications of the given features in the behavioral pattern which is achievable using clustering and classification methods but it is limited as the process is a batch process. Therefore, the incremental analysis method uses already classified behavioral report with the stored prototypes with a new behavioral report of unknown malware. The generation of malware increasing this method helps to perform classification by reduced runtime.

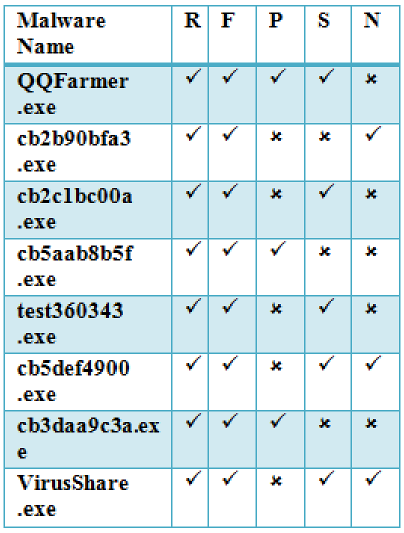
The classification algorithm given below is used in the classification of embedded reports into unique clusters. Classification results in the clear differentiation of malware into its similar behaviors of malware. This is useful in the identification of malware traces.

Algorithm for Classification:

```
1. for x ∈ reports do
2. Determine the nearest prototype in the training
   data
3. If (nearest prototype is within the radius $d_r$)
   (i.e., the Radius of a cluster),
      Assign reports to the particular cluster
   else
      Reject as unknown
```

V. Result Analysis

The experiment involves two set of data such as training and testing. Firstly training samples are given to the algorithms for learning the different features of malware samples. It is carried out in the controlled environment such as a sandbox. The execution traces are collected in the form of a textual report. However, the dynamic analysis of these malware samples in the sandbox gives descriptive details about the actions took place during the analysis, clearly depicting the different aspects such as network, file system, registry and so on activities.



Malware Name	R	F	P	S	N
QQFarmer .exe	✓	✓	✓	✓	✗
cb2b90bfa3 .exe	✓	✓	✗	✗	✓
cb2c1bc00a .exe	✓	✓	✗	✓	✗
cb5aab8b5f .exe	✓	✓	✓	✗	✗
test360343 .exe	✓	✓	✗	✓	✗
cb5def4900 .exe	✓	✓	✗	✓	✓
cb3daa9c3a.ex e	✓	✓	✓	✗	✗
VirusShare .exe	✓	✓	✗	✓	✓

Fig.3. List of system activities from Anubis sandbox

A number of input samples are executed in the sandbox environment and the reports are listed. The analysis time required by the execution of different volumes of samples is compared by taking time as a measure.

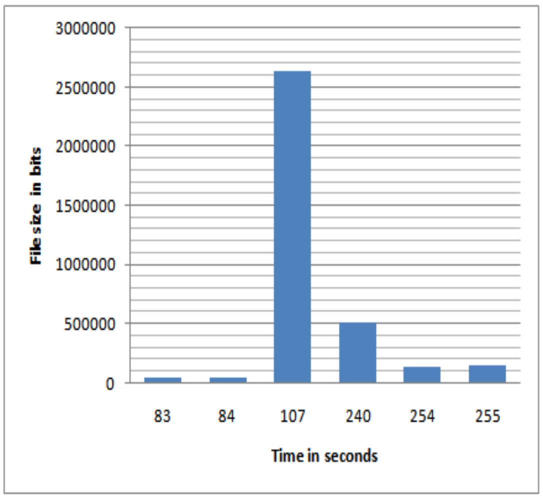


Fig.4. Execution time for files in Anubis framework

The time required for the execution of malware files in Anubis framework is shown in the graph which clearly indicates that even if the size is large it requires greatly less amount of time for execution.

A. Evaluation

The evaluation of components used in the proposed method namely extraction of prototypes, clustering and classification is checked first to see the performance of the framework. In order to achieve this precision and recall, metrics are used.

The precision P defines "how well individual clusters agree with malware classes" and the metric recall R compute "to which extent classes are scattered across clusters". Formally, these two parameters for the number of clusters C and a group of malware classes Y are defined as

$$P = \frac{1}{n}\sum_{c\in C} \#_C \quad \text{and} \quad R = \frac{1}{n}\sum_{y\in Y} \#_y \qquad (3)$$

Where, #c is the biggest number of reports in cluster c which share same class and #y the biggest number of reports labeled y inside one cluster. However, an experimental setup is done for maximizing precision and recall . The entire performance measure for the evaluation is considered and defined as F-measure,

$$F = \frac{2.P.R}{P+R} \qquad (4)$$

This combines precision and recall. A great detection of classes yields F = 1, whereas either a low precision or recall result in a low F-measure.

Table 1. Comparison of clustering methods

Clustering Method	F-measure
Proposed Method with MIST 1	0.950
Proposed Method with MIST 2	0.953
Hierarchical Clustering	0.881

Table 2. Comparison of classification methods

Classification method	F_k	F_u
Proposed Method with MIST 1	0.981	0.967
Proposed Method with MIST 2	0.972	0.954
SVM with text feature	0.807	0.564

Results for the relative valuation of proposed approaches are shown in Table 6.2 and 6.3 taking F-measure as a major performance metric. The analysis of the components in the framework gives the best result for the related methods. The "prototype-based clustering" gives an F-measure = 0.95 for MIST =1 and 0.936 for MIST= 2. For unknown it reached F-measure is 0.96.

Finally, the evaluation of performance metrics such as TPR, FPR, TNR, and FNR are measured. These metrics compares the number of given samples with the number of output reports which results in the correct identification of malware samples. Following formulas are used to calculate these measures:

$$FPR = \frac{n_{ben\to sus}}{n_{ben\to sus} + n_{ben\to ben}} \qquad (5)$$

$$FNR = \frac{n_{sus\to ben}}{n_{sus\to sus} + n_{sus\to ben}} \qquad (6)$$

$$TPR = \frac{n_{sus\to sus}}{n_{sus\to ben} + n_{sus\to sus}} \qquad (7)$$

$$TNR = \frac{n_{ben\to ben}}{n_{ben\to sus} + n_{ben\to ben}} \qquad (8)$$

The use of TPR, TNR, FPR and FNR helps in determining the rate of correct classification of given samples. The reduction in the FPR rate increases the efficiency of the given approach. As shown in the Fig. 5 the rate of FNR is gradual decreased and TPR has obtained high rate representing the good classification

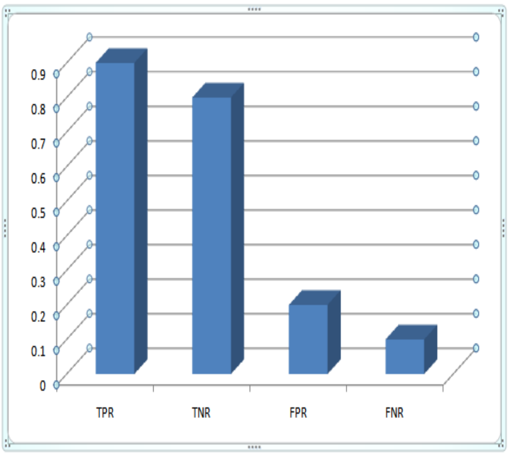


Fig.5. Comparison of Performance metrics

rate. Therefore, this study has given a good result in terms of classification and identification of malware.

The performance can be maintained same with a large number of datasets as there is the usage of preprocessing and filtering. However, the algorithm performs well with the large datasets and also the usage of incremental analysis i.e. comparing the rejected data again with the prototypes results in the classification of unknown malware as well.

VI. Conclusion and Future Scope

Malicious activities in the virtualized environment are becoming more severe. The Internet is posed to vulnerabilities leveraging attackers to access the confidential data. The basic and the static approaches have failed in this area due to the techniques used by the hackers and the lack of identifying the new unknown signature of the malware. Among the several techniques used so far in the development of automated malware analysis, the dynamic malware analysis resulted in better performance.

In this proposed method the extraction of malware behavior, retrieving the best feature, clustering to the related prototype as well as assigning to the corresponding class is achieved which is useful in the detection of malware sample in the virtualized environment. This work is mainly concerned to provide better clustering and classification of malware samples by depicting it as malicious or benign. It has achieved reduced rate of FPR and FNR as compared to the existing approaches. It can be used to overcome the problem of virtual machine holes and malicious activities by a hacker. Since the virtual machines are connected to the host, it is secured at the hypervisor level, thereby ensuring that the protection of data will be at a high rate.

In future, the entire approach can be automated by integrating the modules of the proposed system with better machine learning approaches to have the efficient clustering and classifications of malware data.

References

[1] Ulrich Bayer, Imam Habibi, Davide Balzarotti, Engin Kirda, Christopher Kruegel, “A View on Current Malware Behaviors”, In USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET), 2009

[2] Michael R. Watson, Noor-ul-hassan Shirazi, Angelos K. Marnerides, Andreas Mauthe, David Hutchison,” Malware Detection in Cloud Computing Infrastructures”, IEEE Transactions on Dependable and Secure Computing, DOI. 10.1109/TDSC.2015.2457918,2015

[3] Ulrich Bayer_,Paolo Milani Comparetti_,Clemens Hlauschek_,Christopher Kruegel§, Engin Kirda,” Scalable, Behavior-Based Malware Clustering”, In Proceedings of the Network and Distributed System Security Symposium, 2009

[4] Syarif Yusirwan S, Yudi Prayudi, Imam Riadi, ” Implementation of Malware Analysis using Static and Dynamic Analysis Method”, International Journal of Computer Applications, Vol 117 – No. 6, May 2015

[5] Lorenzo Martignoni, Roberto Paleari, Danilo Bruschi, “A framework for behavior-based malware analysis in the cloud”, 2009

[6] D. Kirat, G. Vigna, and C. Kruegel, “BareCloud: Bare-metal Analysisbased Evasive Malware Detection,” in 23rd USENIX Security Symposium (USENIX Security 14). USENIX Association, 2014

[7] Dhilung Kirat, Giovanni Vigna, “MalGene: Automatic Extraction of Malware Analysis Evasion Signature”, CCS’15, October 12–16, 2015

[8] Philip O’Kane, Sakir Sezer, Kieran McLaughlin, “SVM Training Phase Reduction Using Dataset Feature Filtering for Malware Detection”, IEEE Transactions on Information Forensics and Security, Vol. 8, No. 3, March 2013

[9] Konrad Rieck, Philipp Trinius, Carsten Willems,Thorsten Holz,” Automatic Analysis of Malware Behavior using Machine Learning”,Journal Of Computer Security, Vol. 19, pp. 639-668, 2011

[10] Virusshare. https://virusshare.com/

[11] ANUBIS. https://anubis.iseclab.org/

[12] CWSandbox. http://cwsandbox.org/

[13] Ether. http://ether.gtisc.gatech.edu/

[14] Cuckoo https://www.cuckoosandbox.org/

[15] Malheur http://www.mlsec.org/malheur/

[16] A. K. Marnerides, M. R. Watson, N. Shirazi, A. Mauthe, and D. Hutchison, “Malware analysis in cloud computing: Network and system characteristics,” IEEE Globecom 2013, 2013.

[17] A. K. Marnerides, P. Spachos, P. Chatzimisios, and A Mauthe, “Malware detection in the cloud under ensemble empirical model decomposition,” in Proceedings of the 6th IEEE International Conference on Networking and Computing, 2015.

[18] M. Christodorescu, R. Sailer, D. L. Schales, D. gandurra, and D. Zamboni, “Cloud security is not (just) virtualization security: A short paper,” in Proceedings of the 2009 ACM Workshop on Cloud Computing Security, ser. CCSW ’09. New York, NY, USA: ACM, pp. 97–102. 2009

[19] P. Trinius, C.Willems, T. Holz, and K. Rieck. A malware instruction set for behavior-based analysis. In Proceedings of 5th GI Conference “Sicherheit, Schutz und Zuverl¨assigkeit”, Berlin, Germany, 2010

Authors’ Profiles

Akshatha Sujyothi received B.E degree from V.T.U in 2013 and M.Tech in 2016 in the field of Computer Science and Engineering Currently working as Asst. Professor in the department of computer science & engineering at BIT Mangaluru. Her area of interest is computer security.

Shreenath Acharya received B.E from Mysore University and M.Tech from VTU. Currently serving as Asst. Professor in Computer Science & Engineering Department at St Joseph Engineering College, Mangaluru. He has over 18 years of experience in education sector and 17 publications in international conferences/journals. His areas of interest are cloud computing, computer communication networks and security.

How to cite this paper: Akshatha Sujyothi, Shreenath Acharya,"Dynamic Malware Analysis and Detection in Virtual Environment", International Journal of Modern Education and Computer Science(IJMECS), Vol.9, No.3, pp.48-55, 2017.DOI: 10.5815/ijmecs.2017.03.06