RNN-based Classifier to Detect Stealthy Malware using Localized Features and Complex Symbolic Sequence

Sanket Shukla*, Gaurav Kolhe†, Sai Manoj P D†, Setareh Rafatirad*
*Department of Information Sciences and Technology
†Department of Electrical and Computer Engineering
George Mason University, Fairfax VA, USA 22030
Email: {sshukla4, gkolhe, spudukot, srafatir}@gmu.edu

Abstract*— Malware detection and classification has enticed a lot of researchers in the past decades. Several mechanisms based on machine learning (ML), computer vision and deep learning have been deployed to this task and have achieved considerable results. However, advanced malware (stealthy malware) generated using various obfuscation techniques like code relocation, code transposition, polymorphism and mutation thwart the detection. In this paper, we propose a two-pronged technique which can efficiently detect both traditional and stealthy malware. Firstly, we extract the microarchitectural traces procured while executing the application, which are fed to the traditional ML classifiers to identify malware spawned as separate thread. In parallel, for an efficient stealthy malware detection, we instigate an automated localized feature extraction technique that will be used as an input to recurrent neural networks (RNNs) for classification. We have tested the proposed mechanism rigorously on stealthy malware created using code relocation obfuscation technique. With the proposed two-pronged approach, an accuracy of 94%, precision of 93%, recall score of 96% and F-1 score of 94% is achieved. Furthermore, the proposed technique attains up to 11% higher on average detection accuracy and precision, along with 24% higher on average recall and F-1 score as compared to the CNN-based sequence classification and hidden Markov model (HMM) based approaches in detecting stealthy malware.

***Index Terms*—machine learning, malware detection, stealthy malware, localized features**

I. Introduction

The immense utilization of embedded hardware computing technology in computer systems, has made the system security an indispensable issue. Among multiple security threats, malware is a vital threat due to comparatively less intricacy to design, craft and disseminate into the device(s) [1]. Malicious software, ordinarily known as 'malware' is a software program or an application developed by an attacker to gain inadvertent access to the computing device(s) in order to perform unauthorized accesses as well as malicious activities such as stealing data, accessing sensitive information like credentials, and manipulating the stored information without user's permission.

Traditional and primitive software-based malware detection techniques such as signature-based and semantics-based anomaly detection techniques [2], [3] exist for more than two decades, though effective, they incur remarkable computational and processing overheads yet remain inefficient to detect hidden threats [4]. To overcome this impediment of the software-based malware detection techniques, the work in [5] proposed using the microarchitectural event traces captured through on-chip hardware performance counter (HPC) registers. Despite the better performance compared to the software-based techniques, HPC-based method fails to effectively detect stealthy malware[1] [6].

As the stealthy malware disintegrates itself into the benign code and reassembles dynamically at runtime to launch the malicious behavior, just leveraging globalized features for detecting stealthy malware is not sufficient [7], [8]. This impels us to assimilate and extract the localized features for an efficient detection of stealthy malware.

To overcome the shortcomings of existing works and address the aforementioned challenges, in this work we introduce a novel hybrid technique that uses machine learning for detecting traditional malware and stealthy malware, despite the advanced crafting techniques, with high efficiency. The major contributions of this work to achieve such high performance malware detection can be outlined in three-fold manner as follows:

- Our proposed traditional and stealthy malware detection technique uses a HPC-based method as well as localized feature-based technique. In the HPC-based solution, the HPC traces of a given application are collected during runtime and is validated through a traditional ML classifier for malware detection and classification.
- In the localized feature-based method, the application binaries are translated into image binaries from which local features are extracted and processed through long short-term memory (LSTM) recurrent neural network (RNN) for malware detection and classification.
- Depending on the confidence of the classification for a given application, the class proposed by the two approaches is considered as the final output.

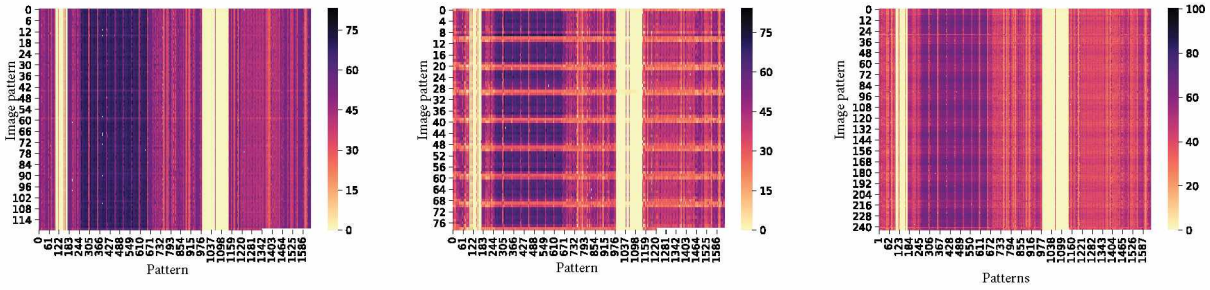
We have evaluated the proposed two-prong method with novel localized feature extraction technique on over 6000 traditional malware samples (backdoor, rootkit, trojan, virus and worm), 2500 stealthy malware samples and 1500 benign samples. The accuracy which we attained on traditional malware samples was 94% and accuracy for stealthy malware was nearly 90% with a F1-score of 92% and recall score of 91%.

II. Motivation

Here, we discuss the key findings which motivated and impelled us to propose the hybrid two-pronged malware detection technique. Figure 1a, Figure 1b and Figure 1c visualizes the

[1]Stealthy malware is a malware which is embedded inside a benign application through sophisticated malware obfuscation techniques, thereby making it complex to detect with traditional approaches as well as HPC-based approach and image processing approaches.

978-1-7281-4550-1/19/$31.00 ©2019 IEEE
DOI 10.1109/ICMLA.2019.00076



(a) (b) (c)

Fig. 1: Heatmap for (a) benign applications; (b) stealthy malware; and (c) malware applications

features for benign, stealthy malware and traditional malware applications respectively in the form of heatmaps.

The y-axis (image pattern) represents the patterns in an application's executable and the x-axis (patterns) represents the observed patterns across the whole dataset corpus. The intensity of each pixel in the heatmap is the cosine similarity percentage between the individual pattern in the file for which the heatmap is generated. High intensity indicates ample presence of a given pattern in the particular application, which shows a close match between the various overall patterns.

We draw the following observations from the plotted heatmaps: (1) In Figure 1a, which is a heatmap for benign application, one can observe high intensity (dark) regions from pattern number 183 to 671. However, the same region appear with less intensity in case of malware, as in Figure 1c. Moreover, across the malware heatmaps (Figure 1b and Figure 1c), no such intense regions can be observed. (2) In the heatmap for stealthy malware (Figure 1b), one can observe that for a given overall patterns, the heatmap across stealthy image patterns are not uniform i.e. one can observe equidistant horizontal light intensity regions and similar faded horizontal regions, indicating uneven pattern occurrence in stealthy malware; and (3) the intensity of patterns are spread across the patterns for stealthy malware heatmap, whereas localized in the case of traditional malware, as observed in Figure 1c and Figure 1b. Altogether, it makes stealthy malware harder to detect. Based on the above observation, we propose extracting and utilizing localized features to distinguish stealthy malware, traditional malware and benign applications.

III. Proposed Malware Detection Methodology

A. Overview of the Proposed Methodology

First, we present the overview of our proposed two-prong methodology depicted in Figure 2 for an efficient malware detection, followed by in-depth details. The incoming application (traditional malware or benign or stealthy malware) is fed to both, HPC-based method and localized feature extraction based computer vision approach simultaneously as shown in Figure 2. In the HPC-based approach, the prominent HPCs information is collected during runtime, which is then fed to ML classifier for malware detection. The prominent HPCs that are needed, are determined offline by obtaining all feasible HPC values and feeding to principal component analysis (PCA) for feature reduction. While, the HPC-based technique performs the dynamic analysis on incoming file, the localized feature based approach is a static approach that utilizes computer vision-based processing for malware detection. In this method, the incoming binary file is converted to a gray-scale image. The patterns are extracted from this gray-scale image and are further labelled to compare with the stored patterns of stealthy and traditional malware by employing a RNN-LSTM. Depending on the classification confidence from both the techniques, the class predicted by the technique with higher confidence is considered as the output class for input application. We describe the details of individual approaches below.

B. HPC-based Detection

In the HPC-based detection technique, we require the microarchitectural event traces captured through HPCs for malware detection. One of the challenges is that there is only a limited number of available on-chip HPCs that one can extract at a given time-instance. However, executing an application generates few tens of microarchitectural events. Thus, to perform real-time malware detection, one needs to determine the non-trivial microarchitectural events that could be captured through the limited number of HPCs and yield high detection performance. To achieve this, we use principal component analysis (PCA) for feature/event reduction on all the microarchitectural event traces captured offline by iteratively executing the application. Based on the PCA, we decide the most eminent events and monitor them during runtime. The ranking of the events is determined as follows:

$$\rho_i = \frac{cov(App_i, Z_i)}{\sqrt{var(App_i) \times var(Z_i)}} \quad (1)$$

where ρ_i is pearson correlation coefficient of any i^{th} application. App_i is any i^{th} incoming application. Z_i is an output data contains different classes (backdoor, rootkit, trojan, virus and worm in our case). $cov(App_i, Z_i)$ measures covariance between input and output. $var(App_i)$ and $var(Z_i)$ measure variance of both input and output data respectively. Based on the ranking, we can select most eminent HPCs and monitor them during runtime for efficient malware detection. These

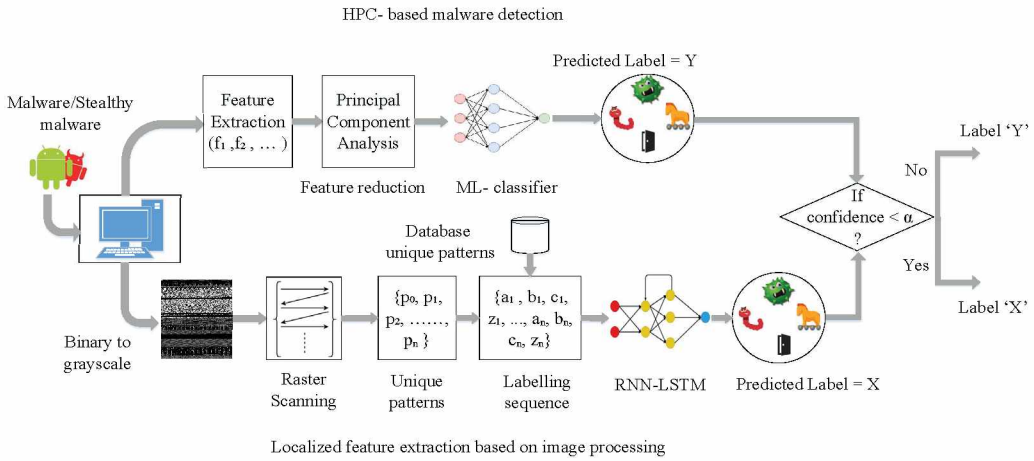


Fig. 2: Proposed hybrid approach for detecting stealthy malware

reduced features collected at runtime are provided as input to ML classifiers which determine the malware class label ($\widehat{Y} \Rightarrow$ backdoor, rootkit, trojan, virus and worm) with higher confidence (α). This HPC-based malware detection technique is fast, robust and accurate in detecting and classifying the traditional malware but it has overhead in terms of area, power and latency. Despite the benefits achieved, this approach does not yield higher performance on stealthy malware due to contamination of HPC when malware is embedded into the benign application. To address this critical issue, a computer vision-based approach is adopted in parallel.

C. Detection based on Localized Feature Extraction

In the computer vision-based detection technique, the application binary is converted into a gray-scale image for localized feature extraction. The incoming binary file is read as a vector of 8 bit unsigned integers and then structured into a 2D array. This can be visualized as a gray-scale image in the range [0,255] (0: black, 255: white) [7]. The width of the image is fixed to 256 and the height is allowed to vary depending on the file size. Since, all the files vary in size from 60 kB - 100 kB, the recommended width is 256 [7]. A raster scanning is performed on the converted binary images as shown in Figure 2, to find the image patterns. Each pattern is of 32×32 block size. We utilize a cosine similarity to distinguish between multiple patterns i.e., if the cosine similarity of two patterns is higher than a threshold (0.75 in this work based on conducted experiments), they are considered to be same. When more than one matched patterns are found in the database, then the one with the highest cosine similarity is considered. Once the image patterns are recognized for a given binary file, the whole image binary is converted into a sequence of patterns (Each pattern is provided with a label). This sequence of labels is fed to a long short-term memory (LSTM) recurrent neural network (RNN). RNN can be fed with the sequences of same length. We perform the padding of zeros to sequence in order to make its length uniform.

Learning of RNN for the patterns happens as follows. Let u_t and h_t denote the input and state vectors, respectively, at time instance t. Let W_{in}, W_{rec}, b, W_{out}, b_{out} be the input to hidden layer weight matrix, recurrent weight matrix, bias, output weight matrix and output bias respectively. Let ω and ϵ be the activation function of the hidden layer and output layer respectively. In our proposed work, tanh is used for hidden layer and softmax is used for the output. The recurrent models are then described by the following equations:

$$h_t = \omega \times (W_{in} \times u_t + W_{rec} \times h_{t-1} + b) \tag{2}$$

$$\widehat{X_t} = \epsilon \times (W_{out} \times h_t + b_{out}) \tag{3}$$

This RNN-model is finally used to classify the incoming stealthy malware binary based on equation (2) and equation (3) and predicts the corresponding class label $\widehat{X_t}$. The rationale to utilize RNN is to exploit the temporal as well as spatial dependencies that attackers utilize to craft stealthy malware.

For the given input application, we have 2 labels (same or different) predicted through HPC-based and computer vision-based approaches. In our work, we consider the confidence of the prediction from both the approaches and the label from predictor with confidence higher than threshold (α = 75%, which is determined through trial-and-error method) is considered to be the associated class for the input application.

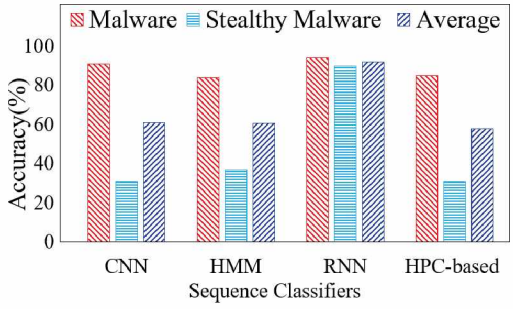


Fig. 3: Performance comparison of sequence classifiers

Classifier	Backdoor			Rootkit			Trojan			Virus			Worm		
	Precision	Recall	F-1	Precision	Recall	F-1	Precision	Recall	F-1	Precision	Recall	F-1	Precision	Recall	F-1
CNN	0.88	0.9	0.87	0.88	0.93	0.9	0.88	0.81	0.8	0.88	0.85	0.83	0.88	0.82	0.78
HMM	0.81	0.8	0.72	0.81	0.85	0.79	0.81	0.76	0.8	0.81	0.82	0.83	0.81	0.7	0.67
RNN(Proposed)	**0.93**	**1**	**0.94**	**0.93**	**0.98**	**0.97**	**0.93**	**0.9**	**0.94**	**0.93**	**1**	**0.97**	**0.93**	**1**	**0.93**
HPC-based	0.8	0.8	0.83	0.8	0.76	0.72	0.8	0.79	0.77	0.8	0.71	0.68	0.8	0.7	0.66

Fig. 4: Comparison of precision, F-1, Recall score

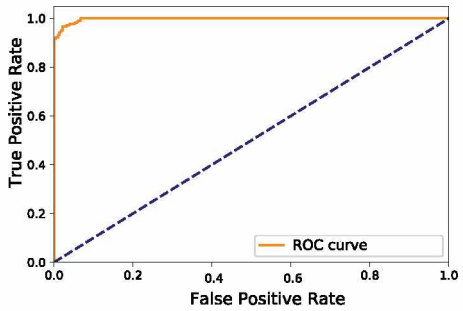


Fig. 5: Receiver operating characteristic (ROC)-curve

IV. Experimental Results

A. Experimental Setup

The proposed methodology is implemented on an Intel core i7-8750H CPU with 16GB RAM. We have obtained malware applications from VirusTotal [9] . We utilized benign applications such as documents (.pdf, .txt, .docx) inside which the binaries of above mentioned malware classes are integrated through code obfuscation (code relocation [10]) process to create 2500 stealthy malware samples. We have randomly placed the malware code into a benign file to increase stealthiness. This process was utilized to create 1600 stealthy malware. The HPC-based mechanism leverages a single layer neural network with 10 neurons in hidden layer. The RNN model has 1 dimensional dropout layer with threshold of 0.7, LSTM layer with 64 neurons and 0.7 recurrent dropout followed by a dense layer with softmax activation. It uses ADAM optimizer and loss is calculated based on the categorical cross entropy.

B. Performance of Malware Detection

For the traditional malware i.e., malware spawned as separate thread, an accuracy of nearly 90% is achieved with HPC-based malware detection. However, for stealthy malware, the HPC-based malware detection accuracy is shattered to 54% on an average (These individual results are not plotted for the purpose of brevity). However, with the proposed hybrid approach, an accuracy of 94% is achieved respectively despite of code obfuscation.

Performance comparison of the sequence classifiers can be illustrated from Figure 3. We observe that for a traditional malware application, RNN sequence classifier based on our localized feature extraction method, classifies traditional malware with the highest accuracy of 94% accuracy. The performance of traditional sequence classification technique deteriorate against stealthy malware, while our localized feature extraction based method achieved an accuracy of nearly 90% against stealthy malware. The most important observation is that the accuracy was definite while detecting stealthy malware which was embedded in benign application using various obfuscation techniques such as randomized code obfuscation, code relocation and polymorphism [10].

In addition to accuracy, we evaluate and compare other performance metrics for malware detection, as shown in Figure 4. Precision score of 0.93 is achieved with the proposed methodology with an average F-1 score and recall score of 0.94 and 0.96 respectively. We conclude that the recall score and F-1 score attained with proposed methodology is approximately 24% higher and 23% higher respectively, compared to others.

From Figure 5, it is evident that with the proposed malware detection, the area under the curve is very close to 1, which indicates a higher robustness. Considering the evaluation evidences, we can substantiate that our proposed hybrid malware detection technique outperforms other mechanisms in detecting traditional and stealthy malware.

V. Conclusion

We propose a hybrid approach of utilizing architectural (trace) as well as code properties, consisting of HPCs and extracted localized features, which are used for stealthy malware detection. In the HPC-based approach, we determine the most prominent HPCs for malware detection and feed them to ML classifier for malware detection. In parallel, we provide the incoming application to the devised image processing technique to convert application binary to a gray-scale image and extract patterns over spatial distribution. For sequence classification, we utilize a RNN to extract and process the localized features to attain the highest average accuracy of 90% over stealthy malware and 94% over traditional malware application. Thus, we conclude that our proposed methodology is robust in detecting stealthy malware and traditional malware.

References

[1] K. Xiao and et. al., "Hardware trojans: Lessons learned after one decade of research," *ACM Trans. Des. Autom. Electron. Syst.*, 2016.

[2] G. Jacob and et.al., "Behavioral detection of malware: a survey towards an established taxonomy," *Journal in Computer Virology*, 08 2008.

[3] N. Patel, A. Sasan, and H. Homayoun, "Analyzing hardware based malware detectors," in *Design Automation Conf.*, 2017.

[4] Q. Chen and R. A. Bridges, "Automated behavioral analysis of malware: A case study of wannacry ransomware," in *Int. Conf. on Machine Learning and Applications (ICMLA)*, 2017.

[5] J. Demme and et al., "On the feasibility of online malware detection with performance counters," in *ISCA'13*, 2013.

[6] S. J. Stolfo and et al., "Towards stealthy malware detection," 2007.

[7] L. Nataraj and et.al., "Malware images: Visualization and automatic classification," in *Symposium on Visualization for Cyber Security*, 2011.

[8] S. Wang and et. al., "High-throughput CNN inference on embedded ARM big.little multi-core processors," *CoRR*, 2019.

[9] G. Sood, "virustotal: R client for the virustotal api," 2017.

[10] I. You and et. al., "Malware obfuscation techniques: A brief survey," in *Int. Conf. on Broadband, Wireless Computing, Communication and Applications*, 2010.