Towards Run-Time Hardware-Assisted Stealthy Malware Detection

Hossein Sayadi[1], Yifeng Gao[2], Hosein Makrani[3], Avesta Sasan[2],
Jessica Lin[2], Setareh Rafatirad[2], Houman Homayoun[3]
[1]Department of Computer Engineering and Computer Science, California State University, Long Beach, CA
[2]Department of Computer Science, George Mason University, Fairfax, VA
[3]Department of Electrical and Computer Engineering, University of California, Davis, CA
[1]{hossein.sayadi}@csulb.edu, [2]{ygao12,asasan,jessica,srafatir}@gmu.edu, [3]{hmakrani,hhomayoun}@ucdavis.edu

***Abstract*—Malware detection using low-level hardware features, e.g. Hardware Performance Counters (HPCs) information, has emerged recently as an effective alternative for traditional signature-based methods to enhance the security of computer systems. Prior works on Hardware-assisted Malware Detection (HMD) have limited their study on detecting malicious applications that are spawned as a separate thread during application execution, hence detecting embedded malware patterns at run-time still remains an important challenge. Embedded malware refers to harmful stealthy attacks in which malicious code is hidden within benign applications and remains undetected by traditional malware detection approaches. In response, in this paper, we propose a customized time series machine learning-based approach to accurately detect embedded malware at run-time using branch instructions feature, the most prominent HPC feature for distinguishing embedded malware from benign applications. With our novel solution, the embedded malware can be detected at run-time with nearly 94% detection performance on average with only one HPC features, outperforming the detection performance of state-of-the-art HMD and general time series classification methods by up to 42% and 36%, respectively.**

***Keywords:* Stealthy Malware; Machine Learning; Run-time Malware Detection; Time Series Classification**

I. Introduction

Malware, a broad term for any type of malicious software, is a piece of code designed by cyber attackers to infect the computing systems without the user consent. It primarily serves for harmful purposes such as stealing sensitive information, unauthorized data access, destroying files, running intrusive programs on devices to perform Denial-of-Service (DoS) attack, and disrupting essential services to perform financial fraud mostly relying on exploiting weaknesses in authentication. The rapid development of information technology has made malware a serious threat to computer systems. Given the exceedingly challenging detection of new variants of malicious applications, malware detection has become more crucial in modern computing systems. The recent proliferation of computing devices in mobile and Internet-of-Things (IoT) domains further exacerbates the malware threats calling for effective malware detection solutions.

Conventional signature-based and semantic-based malware detection methods [1, 2, 3] mostly impose significant computational overhead to the system and more importantly do not scale well. Furthermore, they are unable to detect unknown threats making them unsuitable for devices with limited available computing and memory resources [4]. The emergence of new malware threats requires patching or updating the software-based malware detection solutions (such as off-the-shelf anti-virus) that needs a vast amount of memory and hardware resources which is not feasible for emerging computing systems specially in embedded mobile and IoT devices [5, 6, 7]. In addition, most of these advanced analysis techniques rely on the underlying hardware which makes the existing traditional malware detection techniques hard to import onto emerging embedded computing devices.

In order to address the traditional malware detection shortcomings, Hardware-assisted Malware Detection (HMD), by employing low-level features captured by Hardware Performance Counters (HPCs), have emerged as a promising solution. HMD solutions methods reduce the latency of detection process by order of magnitude with small hardware overhead [8]. The HPCs are basically special-purpose registers implemented into modern microprocessors to capture the trace of hardware-related events such as number of instructions, cache-misses, etc. [9, 10, 11]. While HPCs have been typically used for performance and power tuning of applications [12, 13, 14], in this work we leverage HPCs for security. Recent studies on HMD have demonstrated that malware can be differentiated from normal programs by classifying anomalies using Machine Learning (ML) techniques applied on HPC features [4, 8, 10, 15, 16, 17].

Due to ever-increasing complexity of malware attacks and financial motivations of attackers, malware trends are recently shifting towards *stealthy* attacks [18]. Stealthy attack is a type of cybersecurity attack in which the malicious code is hidden inside the benign application for performing harmful purposes. The main purpose of stealthy attacks is to remain undetected for a longer period of time in the computing system. The longer the threat remains undiscovered in the system, the more opportunity it has to compromise computers and/or steal information before suitable detection mechanism can be deployed to protect against it.

Stolfo et al. discovered a new type of stealthy threat referred as *embedded malware* [19] in which the attacker embeds the malicious code inside a benign file on the target host such that the benign and malicious applications are executed as a single thread on the target system. It has been shown that

traditional signature-based antivirus applications are unable to detect embedded malware even when the exact signature of malware is available in the detector database [19]. As a result, embedded malware is potentially a serious security threat and accurate anomaly detection techniques must be developed to mitigate it.

The existing studies on hardware-based malware detection have primarily assumed that the malware is spawned as a separate thread while executing on the target host. However, in real-world scenarios malicious programs attempt to hide themselves within a benign application to bypass the detection mechanisms. In HMD methods, the HPC data is directly fed to a detector, therefore, for embedded malicious code hidden inside the benign application, the HPC data becomes contaminated, as the collected events include the combined benign and malware microarchitectural events.

In response, in this work we propose an effective time series machine learning-based approach to accurately detect the embedded malicious patterns inside the benign programs using only one HPC feature (branch instruction). To the best of our knowledge, this is the first work that addresses the challenge of detecting stealthy/embedded malware using hardware performance counters features at run-time. The main objective of this work is to accurately detect the malicious application embedded inside the benign program using least number of microarchitectural events (only one HPC features) in which the traditional machine learning-based solutions are unable to detect them with even 8/16 features. Using an effective feature reduction technique, we first identify the most prominent low-level feature for embedded malware detection. Next, we propose a lightweight scalable time series-based Fully Convolutional Neural Network (FCN) model that automatically identifies potentially contaminated samples in HPC-based time series to distinguish the stealthy malware from benign applications at run-time using only branch instructions as the most significant low-level microarchitectural event.

II. Related Work and Background

A. Embedded Malware Detection

Stolfo et al. first [19] proposed the first study on introducing a new type of stealthy malicious attack referred as embedded malware. The authors introduced a new type of stealthy threat referred as embedded malware in which the attacker embeds the malicious code or file inside a benign file on the target host such that the benign and malicious application are executed as a single thread on the target system.

They further introduced a method referred as file-print analysis in which they calculated 1-gram byte distribution of a file to identify the file type among PDF and DOC files. However, their approach is not capable of identifying the exact location of the embedded malware inside a benign file making it unfeasible for effective stealthy malware detection. The work in [20] proposed static and run-time dynamic methods for detecting malware embedded in Word documents. In static analysis, they deployed an open source application to decompose files and produced a similarity score for final classification decision. In their dynamic approach, they employed sandbox-based tests to check OS crashes and unexpected changes to the underlying environment. However, it is acknowledged by the authors that their approach is not practical to be used as an independent malware detection scheme.

The research in [21] used conditional markov n-grams techniques to propose an anomaly detection scheme to detect embedded malware. The rationale for using this type of n-grams is that it provides a more meaningful representation of a file's statistical properties than traditional n-grams methods. They deployed entropy rate, an information-theoretic measure, to quantify changes in Markov n-gram distributions of a file and demonstrated that the entropy rate gets significantly disturbed at malware embedding locations indicating its robustness for embedded malware detction. Their results indicate that the proposed Markov n-gram detector provides better detection and false positive rates than the previous work on embedded malware dettection in [19].

B. Malware Detection using Hardware Performance Counters

Demme et al. [17] was the first study that proposed to deploy HPCs information for malware detection and demonstrated the effectiveness of using traditional ML models for hardware-based malware detection. They showed high detection accuracy result for Android malware by applying complex ML algorithms like Artificial Neural Network (ANN) and K-Nearest Neighbour (KNN). Tang et al. [22] further discussed the feasibility of complex unsupervised learning on low-level features to detect buffer overflow attacks that incurs large overhead and sophisticated analysis. Ozsoy et al. [15] used sub-semantic features to detect malware using Logistic Regression (LR) and ANN algorithms. Moreover, they suggested changes in microprocessor pipeline to detect malware in truly real-time nature which increases the overhead and complexity.

The research in [8] proposed ensemble learning techniques for effective run-time hardware-assisted malware detection and improved the performance of HMD by accounting for the impact of reducing the number of HPC features on the performance of malware detectors. In [23], a machine learning-based HMD is proposed that uses various traditional classifiers, but requires 8 or more features to achieve high accuracy, which makes it less suitable for online malware detection. In addition, a recent work in [4] proposed a two-stage machine learning-based approach for run-time malware detection in which in the first level classifies applications using a multiclass classification technique into either benign or one of the malware classes (Virus, Rootkit, Backdoor, and Trojan). In the second level, to have a high detection performance, the authors deploy a machine learning model that works best for each class of malware and further apply effective ensemble learning to enhance the performance of malware detection.

The work in [24] evaluated the suitability of HPCs for HMD. Though the presented experimental results in [24] are mostly in-favor of efficient malware detection through HPCs, they claim that if HPC traces of malware and benign applications are similar, it is hard to detect malware. However, the robustness of malware detection highly depends on the type of classifier employed. Moreover, it is likely to mislead the HMD methods, if the malware is crafted adversarially to perturb HPC patterns look similar to benign applications patterns, similar to adversarial attack in CNNs for image processing [25]. However, no details on crafting such adversarial applications nor real-world samples are provided. In addition, this work has performed limited analysis on embedded malware and only shows that one benign program infused with ransomware

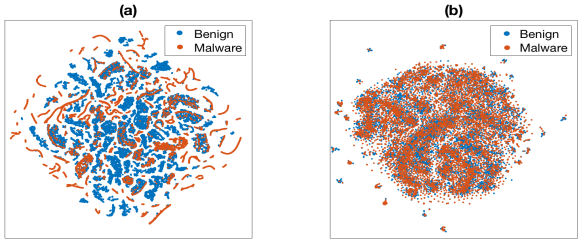

Fig. 1: Visualizing the complete benign and malware dataset using t-SNE algorithm: a) malware spawned as separate thread b) malware embedded inside benign applications

cannot be detected by traditional machine learning-based HMD without providing any effective solution to tackle the challenge of detecting stealthy/embedded malware.

Collectively, the prior works on HMD focus on detecting a threat model in which the malware is spawned as a separate thread within the application execution time. Furthermore, they used traditional ML-based algorithms with more than 4 HPC features to detect the malware with high accuracy. In particular, they have ignored assuming the malicious code embedded inside benign program which is a more threatening attack for today's computing systems. Our work is different, as it targets a more harmful attack, embedded malware, running within the same thread and execution binary of benign programs. It also proposes a lightweight machine learning-based approach which is capable of detecting pattern of embedded malware in the benign application using only one low-level microarchitectural feature.

III. Proposed Malware Detection Framework

A. Challenge of Detecting Stealthy Malware using HPCs

Figure 1 illustrates the challenge of detecting embedded malware. Figure 1-(a) visualizes the complete benign and malware HPC data (described in details in Section III), when the malware spawned as a separate thread, via t-distributed Stochastic Neighbor Embedding (t-SNE) algorithm. As seen, the marginal area between malware and benign program is large when malware spawned as a separate thread indicating that by using traditional ML models (prior works) the malware can be easily detected. However, the converted points of embedded malware data are mixed in Figure 1-(b) depicting the impact of embedding malcode inside benign applications. The figure highlights the challenge of embedded malware detection indicating that due to the dense distribution of malware and benign applications features, traditional classification approaches are not able achieve a high accuracy in detecting embedded malware. As a case study, by applying nearest neighbor classifier on both complete and embedded malware dataset, the classifier can achieve an accuracy of 90% in detecting the malware as a separate thread. However, the classifier can only achieve nearly 60% accuracy in embedded malware detection task when the malicious code is hidden inside the normal program.

B. Data Collection & Feature Reduction

The benign and malware applications are executed on an Intel Xeon X5550 machine (4 HPC registers available) running Ubuntu 14.04 with Linux 4.4 Kernel and HPC features are captured using *Perf* tool. We executed more than 3500 benign and malware applications. Benign applications include real world applications comprising MiBench, SPEC2006, Linux

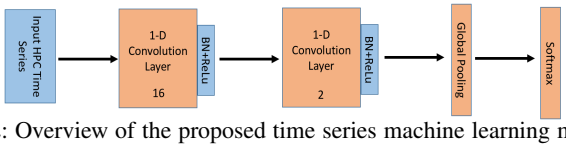

Fig. 2: Overview of the proposed time series machine learning model for embedded malware detection

system programs, browsers, and text editors. Malware applications collected from Virustotal and Virusshare include 850 Backdoor, 640 Rootkit, and 1460 Trojan samples. After collecting the required data, by applying Principle Components Analysis (PCA) the critical HPC features are identified for effective HMD [4, 26]. The proposed time series-based detection approach using only the most significant HPC feature, branch instructions, is able to detect the embedded malware inside benign application with high detection accuracy. Branch operations are one of the non-trivial events as most malwares rely on branching operations for executing the malicious activity. Also, branch related counters can be accessed even in most of the low-end embedded and IoT devices, therefore making this type of microarchitectural events appealing to use for HMD.

C. Stealthy Malware Threat Models

For modeling the embedded malware threats, we have considered persistent malicious attacks which occur once in the benign application with notable amount of duration attempting to infect the system. Persistent malicious codes are primarily a subset of Advanced Persistent Threat (APT) which is comprised of stealthy and continuous computer hacking processes, mostly crafted to perform a specific malfunction activities. For the purpose of thorough analysis, we deployed various malware types for embedding the malicious code inside the benign application including Backdoor, Rootkit, Trojan, and Hybrid (Blended) attacks. For per-class embedded malware analysis, malware traces taken from one category of malware, are randomly embedded inside the benign applications and the proposed detection approach attempts to detect the malicious pattern. Furthermore, the Hybrid threat combines the behavior of all classes of malware.

D. Proposed Framework & Evaluation

The overview of proposed machine learning-based malware detectors is depicted in Figure 2. Intuitively, the network is a simplified version of neural network inspired from previous general convolutional neural network-based time series classification models. As shown in Figure 2-(a), the proposed malware detector is based on the least number of HPC features and targets detecting stealthy attacks that have been ignored in prior studies on hardware-based malware detection. Furthermore, as seen in Figure 2-(b), the network is created by stacking two 1-D convolution layers with 16 and 2 kernels, respectively. The size of kernel in these two convolution layers is 2 and 3, respectively. These convolution layers aim at selecting the subsequence of HPC time series for identifying the malware. Then a global average pooling layer is applied to convert the output of the convolution layer into low dimension features. These features are then fed into a fully connected neuron network to distinguish the embedded malware from benign applications.

For a comprehensive and fair comparison of proposed malware detector with state-of-the-arts, we implemented different

ML-based HMD techniques including JRip, J48, and Logistic Regression, and and time series classification including 1NN a classical time series classification method and Bag-of-Pattern-Features (BOPF) the latest proposed scalable time series classification approach that have all demonstrated high accuracy for detecting malware (spawned as separate thread) in recent works [8, 17, 23, 27].

Table I presents the evaluation results of malware detection for different classes of embedded malware for validation set analysis. The results show that our proposed lightweight neural network-based solution can achieve average accuracy, precision, recall and F-score of nearly 0.9 across all types of experimented embedded malware only by using the most prominent HPC feature (branch instructions). This makes the run-time detection of stealthy malware feasible which is primarily eliminating the need to execute applications multiple times to capture various low-level features suitable for HMD.

TABLE I: Evaluation results for validation set

Type	Precision	Recall	F-score	Accuracy
Hybrid	0.85	0.89	0.88	0.87
Rookit	0.93	0.88	0.91	0.91
Trojan	0.91	0.87	0.89	0.89
Backdoor	0.88	0.94	0.91	0.91
Average	0.89	0.9	0.9	0.89

TABLE II: Comparison of AUC values of testing set for proposed malware detector abd prior works

Attack Type / Method	**Proposed**	JRIP	J48	LR	1NN	BOPF
Hybrid	**0.92**	0.64	0.62	0.53	0.6	0.7
Rookit	**0.98**	0.77	0.62	0.5	0.54	0.53
Trojan	**0.93**	0.85	0.69	0.57	0.65	0.79
Backdoor	**0.91**	0.73	0.54	0.51	0.6	0.68
Average	**0.94**	0.75	0.62	0.52	0.58	0.67

The Area Under the Curve (AUC) values for each embedded malware category are presented in Table II. A higher AUC value means that the classifier is performing better in terms of identifying the stealthy malware and classification of malware and benign applications. As seen, the proposed malware detector achieves an average AUC value of nearly 0.94 across all experimented categories of embedded malware. Furthermore, it significantly outperforms the traditional ML algorithms used in recent HMD works, JRip, J48, and LR, by up to 0.48, and further outperforms tested time series classifications approaches by up to 0.45 (for embedded Rootkit).

IV. Conclusion

Embedded malware is a category of stealthy security threats that allows malicious code to be hidden inside a benign application on the target host and remains undetected by traditional signature-based methods and commercial antivirus software even when the malware signature is present in the detector database. In malware detection using low-level features, when the HPC data is directly fed into a machine learning classifier, embedding malicious code inside the benign applications leads to contamination of HPC information, as the collected features combine benign and malware microarchitectural events together. In response, in this work we proposed lightweight a time series-based Fully Convolutional Neural Network framework to effectively detect the embedded malicious code that is concealed inside the benign applications. Our novel approach, using only the most significant HPC, branch instructions, can detect the embedded malware with 94% detection performance on average at run-time outperforming the detection performance of state-of-the-art HMD methods by up to 42%.

V. Acknowledgment

This research was supported in part by DARPA SSITH program under the award number 97774952.

References

[1] A. A. Elhadi and et al., "Malware detection based on hybrid signature behaviour application programming interface call graph," *American Journal of Applied Sciences*, vol. 9, no. 3, p. 283, 2012.

[2] A. Moser and et al., "Limits of static analysis for malware detection," in *ACSAC 2007*, Dec 2007, pp. 421–430.

[3] M. D. Preda and et al., "A semantics-based approach to malware detection," in *POPL'07*, 2007, pp. 377–388.

[4] H. Sayadi and et al., "2smart: A two-stage machine learning-based approach for run-time specialized hardware-assisted malware detection," in *DATE'19*, March 2019, pp. 728–733.

[5] H. Sayadi and et al., "Customized machine learning-based hardware-assisted malware detection in embedded devices," in *17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications (TrustCom'18)*, 2018, pp. 1685–1688.

[6] S. M. P. Dinakarrao and et al., "Lightweight node-level malware detection and network-level malware confinement in iot networks," in *DATE'19*, March 2019, pp. 776–781.

[7] A. Mosenia and N. K. Jha, "A comprehensive study of security of internet-of-things," *IEEE TETC*, vol. 5, no. 4, pp. 586–602, Oct 2017.

[8] H. Sayadi and et al., "Ensemble learning for effective run-time hardware-based malware detection: A comprehensive analysis and classification," in *Design Automation Conference (DAC)*. IEEE, 2018, pp. 1–6.

[9] X. Wang and et al., "Confirm: Detecting firmware modifications in embedded systems using hardware performance counters," in *ICCAD'15*, Nov 2015, pp. 544–551.

[10] H. Sayadi and et al., "Comprehensive assessment of run-time hardware-supported malware detection using general and ensemble learning," in *ACM International Conference on Computing Frontiers (CF'18)*, 2018.

[11] F. Brasser and et al., "Special session: Advances and throwbacks in hardware-assisted security," in *International Conference on Compilers, Architectures and Synthesis for Embedded Systems (CASES)*, 2018, pp. 1–10.

[12] H. Sayadi and et al., "Machine learning-based approaches for energy-efficiency prediction and scheduling in composite cores architectures," in *ICCD'17*, Nov 2017, pp. 129–136.

[13] M. Malik and et al., "Ecost: Energy-efficient co-locating and self-tuning mapreduce applications," in *Proceedings of the 48th International Conference on Parallel Processing (ICPP'19)*, 2019, pp. 7:1–7:11.

[14] H. M. Makrani and et al., "A comprehensive memory analysis of data intensive workloads on server class architecture," in *MEMSYS'18*. ACM, 2018, pp. 19–30.

[15] M. Ozsoy and et al., "Malware-aware processors: A framework for efficient online malware detection," in *HPCA'15*, Feb 2015, pp. 651–661.

[16] B. Singh and et al., "On the detection of kernel-level rootkits using hardware performance counters," in *ASIACCS'17*, 2017, pp. 483–493.

[17] J. Demme and et al., "On the feasibility of online malware detection with performance counters," in *ISCA'13*, 2013, pp. 559–570.

[18] H. Zhang and et al., "Detection of stealthy malware activities with traffic causality and scalable triggering relation discovery," in *ASIACCS*, 2014.

[19] S. J. Stolfo and et al., "Towards stealthy malware detection," in *Malware Detection*. Springer US, 2007, pp. 231–249.

[20] W.-J. Li and et al., "A study of malcode-bearing documents," in *DIMVA'07*. Berlin, Heidelberg: Springer, 2007, pp. 231–250.

[21] M. Z. Shafiq and et al., "Embedded Malware Detection Using Markov n-Grams," *DIMVA'08*, pp. 88–107, 2008.

[22] A. Tang and et al., "Unsupervised anomaly-based malware detection using hardware features," in *RAID'14*. Springer, 2014, pp. 109–129.

[23] N. Patel and et al., "Analyzing hardware based malware detectors," in *DAC'17*. ACM, 2017, pp. 25:1–25:6.

[24] B. Zhou and et al., "Hardware performance counters can detect malware: Myth or fact?" in *ASIACCS '18*, 2018, pp. 457–468.

[25] I. J. Goodfellow and et al., "Explaining and harnessing adversarial examples," in *arXiv:1412.6572*, 2015.

[26] E. A. et al., "Host-based misuse intrusion detection using pca feature extraction and knn classification algorithms," in *Intelligent Data Analysis*, 2018.

[27] X. Li and et al., "Linear time complexity time series classification with bag-of-pattern-features," in *ICDM'17*, 2017, pp. 277–286.