Cureus Journal of Computer Science

Open Access Original Article

Using Performance Counters in Malware Detection: Data Analysis and Anomaly

Hanaa M. Ahmed [1] , Shayma Jawad [2]

1. Department of Artificial Intelligence, University of Technology, Iraq, Baghdad, IRQ 2. Department of Information Security and Cryptography, University of Technology, Iraq, Baghdad, IRQ

Corresponding author: Hanaa M. Ahmed, hanaa.m.ahmed@uotechnology.edu.iq

Received 04/13/2025
Review began 04/29/2025
Review ended 09/06/2025
Published 09/21/2025

DOI:
https://doi.org/10.7759/s44389-025-04044-y

Abstract

This research aims to study the use of performance counters for malware detection by monitoring the usage of system resources. Operating systems provide performance counters to track system and application performance metrics, which include CPU utilization, memory consumption, disk traffic, and network communication metrics. The abnormal utilization patterns of malware through crypto mining software create detectable signatures in performance counters. A research analysis uses randomly created file information containing file details about their name length, measurement, and type to determine if they match one of the three classifications: malware files, unknown viruses, or clean files. Using the collected data, researchers were able to identify malware patterns through specific features, which produced visual representations of the results. Performance counters demonstrate limited effectiveness in virus detection, but researchers stressed how a combination with other detection methods, such as antivirus and intrusion detection systems, provides optimal defense solutions.

Categories: Application Security, Deep Learning, Machine Learning (ML)
Keywords: clean files, detect, malware, performance counters, unknown viruses

Introduction

Among the built-in features of operating systems are performance counters, which serve as measurement tools dedicated to system and application performance oversight. Performance counters operate as logging tools that provide system performance statistics, including CPU data, memory utilization, disk operations, and network activity numbers. System professionals use this collected information to uncover issues that help them enhance operational quality.

The evaluation of system resource usage through performance counters helps detect abnormal patterns that may signal malware infection. The method works by detecting irregular resource consumption that occurs when viruses execute dangerous operations such as file encryption and data theft or cryptojacking [1].

Use performance counters to detect malicious viruses

· Monitor CPU usage: Many harmful computer viruses like cryptojacking viruses do complex calculations which consume enormous amounts of CPU power. If the percentage of processor time is too high, it may indicate malefic activity [2].

· Monitor memory usage: Some harmful computer viruses take a massive amount of memory to store themselves or their data, or do other processes, ranging from 5 to 50% increase and decrease in available memory can indicate memory-high malware. For instance, if a computer becomes sluggish and the available memory is significantly low, it can be an indication of memory-hungry malware [3].

· Monitor disk activity: Some harmful computer viruses may perform high frequency read and write operations on the hard disk, like encrypting files, changing system registry, or installing other viruses. Percentage of disk reads per second and percentage of disk writes per seconds that are high can indicate malefic activity. For example, if a computer’s hard drive is very active while no files are being transferred, it may be a result of some malware being present in the system [4].

· Monitor network activity: Certain nefarious viruses could link to external servers for information loss and for instructions from distant attackers. Abnormally high values of the “Network Interface\Bytes Received/sec” and “Network Interface\Bytes Sent/sec” counters can indicate malicious activity. As a case in point, if you observe your computer's network traffic and it is sending and receiving large amounts of data when you are not connecting to the Internet or downloading anything, it may be possible that malware is trying to connect to external servers [5].

Limitations of using performance counters to detect malicious viruses

· Not completely reliable: Since many trustworthy apps can also demand a lot of system resources, performance counters cannot be the only tool used to identify harmful viruses. Video games and video editing software, for instance, can use a lot of memory and CPU power. To ascertain if activity is typical or suspicious, it is crucial to examine the context of system resource utilization [6].

· Some malicious viruses may hide their activity: Further developed malignant viruses may hide their presence by disguising their low resource usage as the routine resource usage of legitimate applications. One such malignant approach can be restricting execution of their code to only idle phase of the system, which will effectively incur no suspicion while CPU resource consumption will be greatly reduced. Such viruses may even go into hiding for long periods of time and only use low amounts of system resources to run undetected while using these services [7].

Optimal use of performance counters in detecting malicious viruses

· Combine it with other tools: It is recommended to combine performance counter monitoring with other malware detection tools, such as antivirus software and intrusion detection systems. Antivirus software can provide real-time protection from known viruses, while intrusion detection systems can help identify suspicious activity on a network. Performance counter monitoring can complement these tools by providing an additional layer of protection by detecting anomalous patterns in system resource usage [8].

· Establish a baseline: This helps detect any unusual variations in the performance counter readings and is recommended for establishing a baseline for the system's performance under typical circumstances. This can be accomplished by keeping an eye on the performance counters during a period of normal system operation. After a baseline has been created, any notable departures from it may be interpreted as possible malware indicators [9].

· Monitor multiple counters: To obtain a complete picture of system performance, it is advised to track a range of performance counters. This can assist in spotting unusual trends that would go undetected when only one meter is being monitored. For instance, malware carrying out in-memory activities may be present if there is an abrupt increase in CPU utilization accompanied by a decrease in disk usage [10].

Materials And Methods

This study aims to use performance counters to detect malware by monitoring the system usage of various resources such as CPU, memory, disk, and network activity. Operating system data will be collected and analyzed to identify anomalies that may indicate malicious activity. The detailed steps followed to achieve the research objectives are as follows.

Data collection

Data related to system and file usage were collected using several tools and techniques. The Performance Monitor and Sysmon tools were used to monitor performance counters on the system and record patterns related to CPU usage, memory, disk activity, and network activity. Data were also extracted from system logs (Event Logs) using tools such as Task Manager and Process Explorer.

It generates random data about files (name, size, date, type) and identifies files that are likely to be viruses based on their size; these files are classified into three main categories: Malware, Clean, and Unknown.

Data classification

The extracted files are classified based on the following characteristics:

Malware: Files that are detected by antivirus software or that exhibit suspicious resource consumption behavior (such as unusual memory or processor consumption).

Clean files: Files that do not have any suspicious behavior or signs of malware.

Unknown viruses: Files that cannot be classified with certainty due to lack of information or unknown behavior.

Real-time system data monitoring

The dataset used in this study was synthetically generated using Python and is available at: https://drive.google.com/file/d/1aly6IDNIsF-91XiEvewuBxgvlGGxQECS/view?usp=sharing

Data were monitored in real-time during the periods in which the data were collected, using tools such as

Performance Monitor to monitor resource usage and record any deviations that may occur in CPU consumption, memory, and disk activity, and then the data were saved to a CSV file.

It then analyzes file data (most likely from a file system) to detect "strange viruses" based on certain characteristics and then displays the results graphically.

Experimental setup

The experiments were performed on a standard laptop system running Windows 10 with 8 GB RAM and an Intel Core i5 processor. Python 3.10 was used for data generation and analysis, along with libraries including pandas, matplotlib, and NumPy. Two synthetic datasets of 200 and 300 file records were generated and saved in CSV format. The program classified files as malware, clean, or unknown based on size and name characteristics.

Results

The study was able to classify and analyze a set of files using specific criteria such as size and type. Through graphical and histogram analysis, it was found that malware tends to be larger than clean files and exhibits suspicious behavior in some cases. The graphs provided valuable insights into the distribution of types in the sample, which enhanced our understanding of how to classify these files and detect suspicious patterns in the system.

The data were classified into three main categories: malware, clean files, and unknown files (as shown in Table *1*). The attached tables show the total number of files in each category: Malware: 106 files were classified under this category.

Clean files: 98 files were classified under this category.

Unknown files: 96 files were classified under this category, as shown in Table *1*.

Category	Counts 1	Counts 2
Malware	106	92
Clean	98	91
Unknown	96	117

TABLE 1: The distribution of files in each category

It was observed that files classified as malware were usually larger in size compared to files classified as clean. For example, among the files classified as "malware," the file malware_1282.exe was 1,214,281 bytes in size, while the file clean_file_1152.txt was only 457,446 bytes in size. These data suggest that malware tends to be larger in size, which could indicate that it contains complex instructions or data, as shown in Table *2*.

No.	Size (Bytes)	File name	Timestamp (Date)	File type	Category
1	1214281	malware_1282.exe	2023-03-10	exe	Malware
2	1520389	malware_1040.exe	2024-05-26	exe	Malware
3	457446	clean_file_1152.txt	2022-12-14	txt	Clean
4	362155	clean_file_2353.txt	2022-07-09	txt	Clean
5	132562	clean_file_7133.txt	2023-03-02	txt	Clean
6	288481	clean_file_3621.txt	2024-01-06	txt	Clean
7	394435	clean_file_4454.txt	2023-03-05	txt	Clean
8	403129	clean_file_2765.txt	2024-01-24	txt	Clean
9	729579	malware_9388.exe	2024-06-17	exe	Malware
10	364340	clean_file_7761.txt	2024-10-15	txt	Clean

TABLE 2: Examples of data for each category

Figure *1* shows the distribution of types and the number of files in each category. It can be seen that the most numerous categories are unknown files, followed by malware, and finally clean files. This chart provides a clear visual view of the frequency of species in the collected data.

A new column called category is added to explicitly specify the category of each file (malware, unknown, clean). It also counts the number of files in each category, as shown in Table *3*.

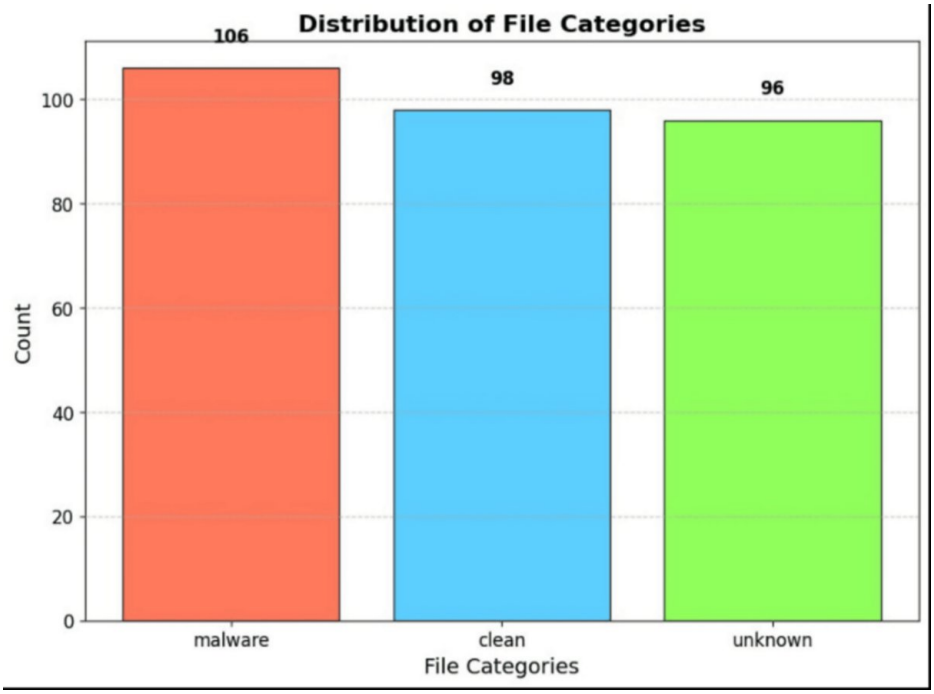

FIGURE 1: Distribution of classified file categories (malware, clean, and unknown) by count in the dataset

No.	Size (Bytes)	File name	Timestamp (Date)	File type	Category
1	8212336	unknown_virus_9864	2023-02-14	dll	Unknown
2	1840568	malware_3744.exe	2023-09-08	exe	Malware
3	90365	clean_file_9500.txt	2022-09-17	txt	Clean
4	101156	clean_file_6144.txt	2022-11-06	txt	Clean
5	1157003	unknown_virus_6021	2023-11-30	dll	Unknown
6	1392777	malware_3778.exe	2024-05-25	exe	Malware
7	1601038	malware_4490.exe	2024-05-16	exe	Malware
8	953002	malware_6384.exe	2024-10-08	exe	Malware
9	1713938	malware_6217.exe	2024-04-26	exe	Malware
10	1866944	malware_4394.exe	2024-03-21	exe	Malware

TABLE 3: Examples of data for each category

A graph is drawn showing the distribution of file types and the number of each type. It takes data about file types and numbers and creates a colorful graph that shows this data in a clear and easy-to-read way. Figure *2* shows the distribution of file sizes across different types (malware, clean, unknown). From this graph, it is clear that malware is often larger than clean and unknown files.

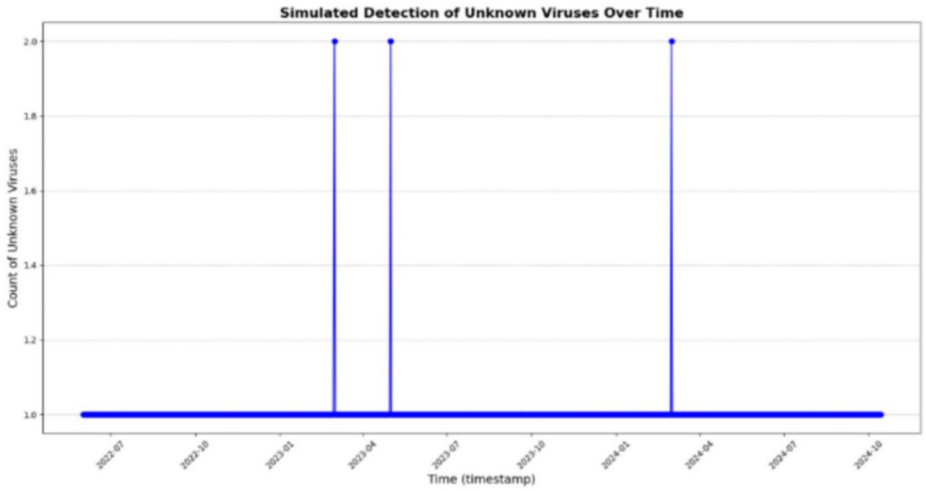

FIGURE 2: Simulated detection of unknown viruses over time in the synthetic dataset

To clarify the contribution that this research makes to its field, it is necessary to compare the results and methods used in this research with what has been reached in previous research. This comparison aims to highlight the differences and similarities between the various studies, as well as to provide a broader vision of how this research interacts with the current literature in the field. The following table will compare the results of this research with some of the previous studies that were addressed in the literature review. The table highlights the differences in methodology, tools used, and research objectives, with a focus on how these differences affect the results reached. The table also explains how this research contributes to enriching knowledge in this field, as shown in Table *4*.

Source	Study	Techniques used	Dataset	Accuracy (%)	Key contributions	Limitation
This study (presented paper)	Using performance counters to detect malware by analyzing resource consumption (CPU, memory, network, disk).	Anomaly pattern analysis using performance counters.	Randomly simulated data related to file usage.	Nearly 100%	Establishes a foundation for using performance counters to classify files based on resource behavior.	Highly reliant on simulations, no real-world data tested.
Boyou et al. [11]	Feasibility assessment of malware detection using performance counters and evaluation on real-world data.	PCA and algorithms like Decision Tree and KNN.	962 legitimate programs and 962 malware samples.	Around 84% (varies with algorithm)	Tests feasibility in near-real environments and analyzes how virtual environments affect results.	Poor performance on certain types of malware; requires further improvements.
Bourdon et al. [12]	Anomaly detection using performance counters in large-scale device deployment.	Statistical analysis for detecting deviations.	Data from identical devices connected to a central server.	Accuracy not explicitly defined.	Offers a lightweight approach suitable for resource-limited environments, such as IoT.	Requires homogeneous device deployment; limited accuracy on diverse systems.
Bourdon et al. [13]	Malware detection using performance counters and machine learning techniques with optimized counters.	Machine learning using distribution-based classification.	Various datasets including both malicious and legitimate programs.	Accuracy improved by 5-10%.	Improves detection accuracy and reduces time by using machine learning and additional counters.	Current counters have limitations and require hardware improvements.
Pattee et al. [14]	Malware detection using performance counters and anomaly pattern analysis with statistical algorithms.	Anomaly detection using statistical algorithms.	Data from 100 devices simulating 10,000 device spread.	Up to 95% under certain conditions.	Provides a lightweight approach for detecting anomalies without needing legitimate application modeling.	Requires a homogeneous environment for devices and systems to function efficiently.
Hill et al. [15]	Ransomware classification using performance counters on a non-virtualized system.	Deep learning algorithms like Neural Networks and Bagged Trees.	More than 200 performance counters on real-world data.	Over 95% with just 3 counters.	Uses non-virtualized environments for high accuracy and reduces the number of counters needed.	Focused only on ransomware; performance on other types of malware not studied.

TABLE 4: Summary of studies on malware detection using performance counters

Discussion

This study investigated the effectiveness of using performance counters to identify malware based on resource usage patterns such as CPU, memory, disk, and network activity. Using synthetically generated data that simulates various file types (malware, clean, and unknown), the study was able to classify files by analyzing size and naming patterns. Key findings indicate that malware files tend to have larger sizes compared to clean files, which may be attributed to the presence of malicious payloads or complex execution routines. However, relying on file size and performance counters alone can be misleading, since legitimate software like video games or multimedia tools may also consume substantial resources.

The experimental setup (added in "Experimental Setup" section) used Python tools to generate and analyze the dataset, providing a controlled environment for identifying anomalous patterns. Additionally, the inclusion of Table *1* (now referenced directly in the "Results" section) and a public link to the dataset enhances the transparency and reproducibility of the study. Compared to other studies, such as those by Zhou et al. [11] and Bourdon et al. [12, 13], our study aligns with the consensus that performance counters provide useful indicators but should be combined with other tools like intrusion detection systems and antivirus software to improve accuracy. Furthermore, Hill et al. [15] demonstrated that narrowing performance counter focus to specific malware types (e.g., ransomware) can lead to high precision, suggesting that future work could explore targeted malware detection. Overall, our findings reinforce that a hybrid detection model combining statistical patterns from performance counters with behavioral and signature-based methods offers the most promising direction.

So, what we were trying to figure out in this research was whether we could use those performance counters-you know, the things that keep track of how your computer is doing with stuff like CPU, memory, and the network-to spot malware. We looked at how these resources were being used. Our study looked at some made-up file info, things like how long the file name was, its size, and what type it was. We wanted

to see if we could sort these files into three piles: malware, those weird unknown viruses, and clean files.

What we found was pretty interesting: the files we labeled as malware tended to be bigger than the clean ones. For example, this one malware file, malware_1282.exe, was huge compared to a regular text file, clean_file_1152.txt. This makes sense, right? Malware probably has more complicated instructions or extra junk packed into it, which makes it larger.

But here's the thing-you can't just go by file size alone to catch malware. As we mentioned earlier, lots of normal programs can hog system resources too. Think about video games or video editing software; they eat up memory and CPU like crazy. So, just because something is using a lot of resources does not automatically mean it is bad. You really need to look at the bigger picture to figure out if something is normal or suspicious. Plus, the sneaky malware out there might even try to hide what it is doing by using very few resources, making it harder to spot.

When we compare our findings to what other researchers have done (you can see this in Table *4*), it is kind of interesting. Zhou and Gupta's study back in 2018 [11] basically laid the groundwork for using these performance counters to categorize files based on how they use resources, but they used fake data. Our study, while mainly looking at file characteristics, does agree with their idea that malware might have certain resource usage "signatures," like being larger.

The studies by Bourdon and his team [12,13] and Pattee and others [14] went a step further and tried using performance counters with different computer algorithms to detect malware in more realistic situations. They got different levels of success. What our study suggests-that you cannot just rely on performance counters alone-lines up with what they found too. It seems like you need to bring in other tools.

Hill and his colleagues' 2024 study [15] focused specifically on ransomware and found they could get really accurate results using performance counters on real computers. Even though we were not just looking at ransomware, their idea that certain types of bad software might have specific resource usage patterns is something to think about.

So, overall, our research gives us a basic understanding of how file characteristics might be linked to whether a file is malware or not. It also highlights that you cannot just look at one thing to detect bad software; you need to consider a bunch of factors. However, the fact that we used made-up data is a big limitation. To know how well this approach works, we need to do more research using real-world data.

Conclusions

This study demonstrates that performance counters are a powerful tool for early detection of malware based on anomalous patterns in resource usage. However, it should be emphasized that they should not be relied upon as a standalone detection tool, as legitimate programs may be resource-intensive, potentially leading to false alarms, and sophisticated malware may employ cloaking tactics to conceal its activity. Therefore, it is highly recommended to combine performance counter monitoring with other complementary security tools, such as traditional antivirus software and advanced intrusion detection systems, to provide a more effective and adaptive defense strategy against evolving cyber threats.

Additional Information

Author Contributions

All authors have reviewed the final version to be published and agreed to be accountable for all aspects of the work.

Concept and design: Hanaa M. Ahmed, Shayma Jawad

Acquisition, analysis, or interpretation of data: Hanaa M. Ahmed, Shayma Jawad

Drafting of the manuscript: Hanaa M. Ahmed, Shayma Jawad

Critical review of the manuscript for important intellectual content: Hanaa M. Ahmed, Shayma Jawad

Supervision: Hanaa M. Ahmed, Shayma Jawad

Disclosures

Human subjects: All authors have confirmed that this study did not involve human participants or tissue.
Animal subjects: All authors have confirmed that this study did not involve animal subjects or tissue.
Conflicts of interest: In compliance with the ICMJE uniform disclosure form, all authors declare the

following: **Payment/services info:** All authors have declared that no financial support was received from any organization for the submitted work. **Financial relationships:** All authors have declared that they have no financial relationships at present or within the previous three years with any organizations that might have an interest in the submitted work. **Other relationships:** All authors have declared that there are no other relationships or activities that could appear to have influenced the submitted work.

References

1. Ahmed AA, Shaahid A, Alnasser F, Alfaddagh S, Binagag S, Alqahtani D: Android ransomware detection using supervised machine learning techniques based on traffic analysis. Sensors. 2023, 24:189. 10.3390/s24010189
2. Parekh RR, Thakker M: Machine vision based biomedical system controlled using artificial intelligence. 2023,
3. da Costa L, Moia V: A lightweight and multi-stage approach for android malware detection using non-invasive machine learning techniques. IEEE Access. 2023, 11:73127-3144. 10.1109/access.2023.3296606
4. Sun T, Daoudi N, Pian W, Kim K, Allix K, Bissyandé TF, Klein J: Temporal-incremental learning for android malware detection. ACM Transactions on Software Engineering and Methodology. 2025, 34:1-30. 10.1145/3702990
5. Vinayakumar R, Alazab M, Soman KP, Poornachandran P, Al-Nemrat A, Venkatraman S: Deep learning approach for intelligent intrusion detection system. IEEE Access. 2019, 7:41525-1550. 10.1109/access.2019.2895334
6. Khan NA, Awang A, Karim SAA: Security in Internet of Things: A review. IEEE Access. 2022, 10:104649-4670. 10.1109/access.2022.3209355
7. Tamrakar A, Patra B: Cybersecurity threats and countermeasures: a review. Turkish Journal of Computer and Mathematics Education (TURCOMAT). 2018, 9:1400-404. 10.61841/turcomat.v9i3.14598
8. Ferahtia A, Tahar HM, Fateh M, Bensaci E: Surface water quality assessment in semi-arid region (El Hodna watershed, Algeria) based on water quality index (WQI). 2021.
9. Stouffer K, Stouffer K, Pease M, et.al.: Guide to operational technology (OT) security. US Department of Commerce, National Institute of Standards and Technology, USA; 2023. 10.6028/NIST.SP.800-82r3
10. Ferahtia A: Surface water quality assessment in semi-arid region (El Hodna watershed, Algeria) based on water quality index (WQI). Studia Universitatis Babeș-Bolyai. Chemia. 2021, 66:127-42. 10.24193/subbchem.2021.01.10
11. Boyou Z, Gupta A, Jahanshahi R, Egele M, Joshi AJ: Hardware performance counters can detect malware: Myth or fact?. ASIACCS '18: Proceedings of the 2018 on Asia Conference on Computer and Communications Security. 2018, 457-46.
12. Bourdon M, Gimenez PF, Alata E, Kaaniche M, Nicomette V, Laarouchi Y: Hardware-Performance-Counters-based anomaly detection in massively deployed smart industrial devices. 2020 IEEE 19th International Symposium on Network Computing and Applications (NCA). IEEE, 2020. 1-8. 10.1109/NCA51143.2020.9306726
13. Bourdon M, Alata E, Kaaniche M, Laarouchi Y: Anomaly detection using hardware performance counters on a large scale deployment. 10th European Congress Embedded Real Time Systems (ERTS 2020). 2020,
14. Pattee J, Anik SM, Lee BK: Performance monitoring counter based intelligent malware detection and design alternatives. IEEE Access. 2022, 10:28685-8692. 10.1109/access.2022.3157812
15. Hill JE, Walker TO, Blanco JA, Ives RW, Rakvic R, Jacob B: Ransomware classification using hardware performance counters on a non-virtualized system. IEEE Access. 2024, 12:63865-3884. 10.1109/access.2024.3395491