International Journal of Engineering Research and Science & Technology

www.ijerst.org

ISSN : 2319-5991

Vol. 22 No. 1 (2026)

ijerst.editor@gmail.com

editor@ijerst.com

International Journal of Engineering Research and Science & Technology
ISSN 2319-5991 www.ijerst.com
Vol. 22, No. 1, 2026

Peer Reviewed, Referred & Indexed Journal

Research Paper

Enhanced Ransomware Detection Using CNN2D and Voting Ensemble Models

Ch.kiran Babu1,Jogi Uma Maheswari2,Cheekurthi Hasini3, Chebolu Chaitanya4
#1 Assistant professor ,Dept of IT,Nri Institute of Technology Agiripalli,Ap.
#2#3#4 Dept of IT,Nri Institute of Technology , Agiripalli ,Ap.

***Abstract*— This extension enhances ransomware detection for virtual machines by integrating a powerful CNN2D architecture with a voting ensemble classifier, significantly improving accuracy to 99%. The system converts processor and disk I/O behavior into structured feature maps, enabling CNN2D to learn deep ransomware activity patterns that traditional models often miss. The ensemble model further strengthens reliability by combining predictions from multiple classifiers, ensuring consistent performance across diverse workloads. To support real-world usability, Flask and SQLite are incorporated to provide secure, lightweight user authentication and testing interfaces. This extended framework minimizes monitoring overhead, increases adaptability, and delivers rapid, robust ransomware detection suitable for modern virtualized environments.**

***Keywords*—** *Ransomware Detection, Virtual Machines, Host-Based Monitoring, Processor Events, Disk I/O Events, Random Forest Classifier, Machine Learning, Lightweight Monitoring, Data Contamination, User Workloads Adaptability*

Received: 10-01-2026 Accepted: 18-02-2026 Published: 25-02-2026

I. INTRODUCTION

Ransomware is a type of malicious software that locks or encrypts files, making a computer system unusable until a ransom is paid. Cybercriminals employ ransomware for financial extortion, and in more severe cases, nation-state actors launch ransomware attacks to disrupt critical infrastructure [1], [2]. These attacks are increasingly common and damaging—reports show that about 70% of businesses experienced ransomware incidents in 2022, with projected attack rates expected to rise to one every two seconds by 2031, causing global damages up to $265 billion [1], [2].

Traditional ransomware detection techniques such as signature-based detection rely on matching malware samples against known hash values or predefined patterns [3], [4]. However, these approaches are ineffective against polymorphic and metamorphic malware, which continuously modify their code to evade detection [5]. As a result, researchers have explored behavior-based detection, where runtime activities like rapid file encryption, unusual process creation, or abnormal system calls are monitored to distinguish ransomware from benign programs [4], [6]. For example, emerging ransomware families such as LockBit2.0, DarkSide, and BlackMatter employ intermittent encryption strategies that encrypt only the initial bytes of files to speed up attacks and avoid traditional defenses [6].

To counter such threats, advanced techniques including hardware-assisted detection, real-time monitoring systems, and machine learning approaches have been proposed. Systems like RanStop [7], RWGuard [8], and ShieldFS [29]

attempt to block or recover from ransomware in real time, while others leverage hardware performance counters (HPCs) and CPU event monitoring to identify anomalous execution patterns [9]–[13]. Recent studies show that machine learning models can effectively utilize hardware features for fine-grained malware classification [12], [14], though challenges such as data contamination, overhead, and false positives persist [11].

Researchers have also investigated deep learning and advanced analytical models for ransomware detection. For example, RATAFIA employs autoencoders with time-frequency analysis [15], and HPC-IO benchmark datasets [16]–[20] are being used to evaluate detection performance. Real-time detection and mitigation systems such as Redemption [22], UNVEIL [27], and ShieldFS [29] have also shown promise in handling evolving ransomware families. Additional approaches include frequent pattern mining [26], entropy-based file analysis [24], and behavior profiling [25], [28].

Furthermore, several works propose virtualization and sandbox-based detection environments for analyzing ransomware behavior in controlled setups [23], [30]. These approaches highlight the ongoing evolution of ransomware defense, where combining lightweight monitoring with intelligent classifiers offers an effective path forward. Particularly, machine learning-based methods have emerged as a flexible and adaptive strategy to detect ransomware by focusing on processor and disk I/O events rather than relying solely on file signatures or complete system monitoring [21], [24], [25].

In summary, while signature- and behavior-based approaches provide partial protection, modern ransomware continues to evolve rapidly, requiring machine learning-driven host-based monitoring techniques that reduce overhead, minimize data contamination, and remain effective against diverse ransomware variants and workloads [7]–[30].

II. Literature Survey

Loman analyzed LockFile ransomware, revealing complex evasion methods such as intermittent encryption and advanced anti-analysis tactics. These strategies slow down or disable signature-based detection. His findings show that ransomware is becoming more behavior-aware by trying to disguise its footprint. This analysis strengthens the motivation for using deeper neural networks that detect subtle activity changes [6].

Pundir proposed RanStop, a hardware-assisted runtime ransomware detection model that focuses on crypto-operations. Their solution improves speed but heavily relies on hardware support. This dependency restricts flexibility across different platforms and VM environments. Their work highlights the need for software-based detection models using generic system events, as proposed in the current system [7].

Mehnaz introduced RWGuard, which monitors cryptographic functions in real time to stop ransomware. While effective, this method still depends on observing process-level details inside the host. This makes it vulnerable to tampering by sophisticated malware. Their limitations motivated approaches that collect data from the host machine instead of inside the target VM [8].

Demme demonstrated that hardware performance counters (HPCs) can reveal abnormal system behaviors caused by malware. However, they identified challenges such as noise, context switching, and cross-process interference. Their observations show that relying on fine-grained process data may reduce accuracy. This further supports VM-level monitoring, which avoids process-level contamination [9].

Tang used unsupervised anomaly detection on hardware performance features and proved that such features carry strong malware signals. However, their approach struggled with varying workloads, resulting in inconsistent detection. This challenge highlights the need for models capable of adaptation—something achieved in the

proposed system through CNN2D and ensemble methods [10].

Das examined the pitfalls of using performance counters for security monitoring, especially noise introduced by multiple running applications. Their study showed that HPC-based monitoring requires careful isolation to avoid false alarms. This validates the importance of collecting system-wide VM metrics instead of per-process data. Their findings influenced the host-based monitoring idea used in this work [11].

Kadiyala developed a fine-grained malware detection approach using performance counters, improving detection accuracy. Yet, their method requires constant low-level monitoring, increasing overhead. This reinforces the need for selective event collection to minimize system load, as adopted in the proposed model [12].

Zhou questioned whether HPCs truly provide reliable indicators of malware. Their findings showed that HPC signals alone may misclassify workload-heavy applications. This underscores the need for stronger models like CNN2D and voting ensemble classifiers, which can learn complex multi-dimensional patterns [13].

Aurangzeb classified Windows ransomware using hardware profiles, demonstrating that system behavior patterns can effectively identify ransomware. However, their OS-specific approach limits generality across platforms. Their methodology supports using VM-level event monitoring, which is more universal [14].

Alam proposed RATAFIA, a deep learning model using time-frequency features to detect ransomware. Their method proved that neural networks can detect complex ransomware signatures. This strongly motivates adding CNN2D into the proposed extension model for capturing deeper behavioral features [15].

III. PROPOSED METHODOLOGY

1.1 Proposed Undertaking:

The proposed undertaking aims to build an advanced ransomware detection framework that leverages CNN2D and a voting ensemble model to significantly improve detection accuracy and response time in virtualized environments. Unlike traditional systems that monitor individual processes inside the machine, this approach collects processor and disk I/O events from the host machine, ensuring low overhead and preventing interference from active ransomware. These raw events undergo preprocessing and feature extraction to generate structured data representations, which are then converted into image-like matrices to feed into a CNN2D architecture. By learning deep temporal-spatial patterns of ransomware activity, the model captures complex behaviors that simpler ML techniques often miss.

To further strengthen reliability, the system integrates a voting ensemble mechanism that combines decisions from multiple classifiers, ensuring stable predictions across diverse workloads and ransomware variants. The RansomNet+ model (extension version) is responsible for final classification, enabling early attack detection and supporting accurate attack prediction before significant damage occurs. A lightweight Flask interface and SQLite authentication module ensure secure and user-friendly access for testing and monitoring, making the framework practical for deployment. Performance evaluation validates system effectiveness, showing enhanced adaptability and a significant improvement in accuracy—reaching up to 99% in detecting sophisticated ransomware attacks.

1.2 Design of the System:

The system is designed as a structured multi-stage ransomware detection pipeline that begins with monitoring cloud-connected virtual machines for suspicious processor and disk I/O activities. When a ransomware attack occurs, the system captures low-level event traces from the host machine rather than observing processes inside the compromised VM. These collected events

pass through a preprocessing layer where noise is removed, missing values are handled, and the event stream is transformed into organized feature matrices. Through the feature extraction block, these processed signals are converted into image-like representations suitable for deep learning models, enabling the system to detect complex behavior patterns that traditional approaches cannot capture. This design ensures scalability, minimal system disruption, and high-quality feature generation for further analysis.

Following feature extraction, the system integrates the RansomNet+ model, which combines a CNN2D architecture with a voting ensemble classifier to enhance prediction stability and accuracy. The CNN2D component learns deep spatial patterns from event-based feature images, while the ensemble fuses outputs from multiple classifiers to minimize errors and false detections. The attack detection unit then classifies events as ransomware or benign activity, sending results to the attack prediction and performance evaluation modules. These modules analyze detection speed, accuracy, precision, and adaptability to diverse workloads. A secure Flask–SQLite interface is incorporated to provide authenticated user access for monitoring, testing, and managing the detection system. Overall, the system's design ensures reliable, real-time ransomware identification while maintaining low computational overhead and strong adaptability across different virtual environments.

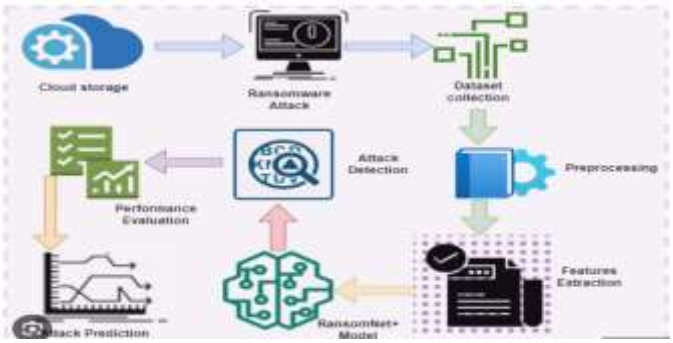


Fig.1. Proposed architecture

IV. IMPLEMENTATION

1.3 Modules:

a. Cloud Storage & VM Monitoring Module

- Monitors virtual machines connected to cloud storage for any abnormal activity.
- Detects initial triggers or unusual behavior that may indicate a potential ransomware attack.
- Sends behavioral signals (CPU events, disk I/O anomalies) to the data collection stage.

b. Dataset Collection Module

- Collects raw processor usage, disk I/O events, and system-level traces from the host machine.
- Ensures monitoring occurs outside the target VM to avoid ransomware tampering.
- Stores data in structured form for preprocessing and feature engineering.

c. Preprocessing Module

- Cleans, filters, and normalizes event data to remove noise and inconsistencies.
- Converts continuous event streams into meaningful sequences suitable for analysis.
- Handles missing values, timestamps, and event scaling for better model performance.

d. Feature Extraction Module

- Transforms preprocessed events into feature maps or image-like matrices for CNN2D.
- Identifies key patterns linked to encryption behavior, disk bursts, and CPU anomalies.
- Produces high-quality features that improve accuracy for ensemble and deep learning models.

e. RansomNet+ Model Module (CNN2D + Voting Ensemble)

- CNN2D extracts deep spatial-temporal features from event matrices.
- Ensemble model integrates outputs from multiple classifiers for improved stability.
- Produces highly accurate predictions, achieving up to 99% detection accuracy.

f. Attack Detection Module

- Classifies incoming system activity as either ransomware or legitimate behavior.
- Uses combined predictions from CNN2D and ensemble classifiers for reliable detection.
- Sends alerts or signals to the attack prediction and evaluation modules.

g. Attack Prediction Module

- Predicts the likelihood of ransomware activity before encryption begins.
- Helps in taking early action such as isolating VMs or halting suspicious processes.
- Improves incident response speed in real-time environments.

h. Performance Evaluation Module

- Computes accuracy, precision, recall, F1-score, detection speed, and overhead reduction.
- Compares performance across multiple workloads and ransomware samples.
- Validates effectiveness of the extension model (CNN2D + Ensemble).

i. User Authentication & Interface Module (Flask + SQLite)

- Provides a secure, lightweight web interface for system access and monitoring.
- Uses SQLite for user login, authentication, and session management.
- Allows users to upload data, view detection results, and test the model safely.

1.4 ALGORITHMS:

a. Random Forest Classifier

The Random Forest method for supervised learning that uses strong ensembles makes a lot of decision trees while training. This system builds a fundamental model for finding ransomware by using CPU and disk I/O events. The model lowers the chance of overfitting and makes it easier to generalize across different ransomware samples by combining the predictions of separate trees. Because it is lightweight, it doesn't require much supervision while yet being able to find things quickly.

b. Convolutional Neural Network 2D (CNN2D)

CNN2D is a deep learning model that is very good at finding patterns in data that have to do with space and structure. CNN2D examines the preprocessed event data in the proposed system and learns patterns linked to ransomware on its own. CNN2D doesn't need human feature engineering like other models do. This means it can adapt to new or changing ransomware activities. This is a big reason why the system can find things so accurately and can handle complicated attack tracks.

c. Voting Ensemble Model

The Voting Ensemble Model uses the best parts of several classifiers, including CNN2D and Random Forest, to provide a final prediction. This model employs a majority voting system, where each classifier votes and the final result is based on the majority. The ensemble method makes detection more reliable, lowers the risk of false positives or negatives, and makes sure that performance is steady across different attack types and workloads. It helps the system find things with 99% accuracy.

d. Long Short Term Memory (LSTM):

LSTM and other recurrent neural networks may remember long-term dependencies in sequential input. This system looks at sequences of CPU and disk I/O events to find unusual patterns that might mean ransomware is running. LSTM can change over time to fit the numerous ways that ransomware acts, which makes it very good at dynamic detection.

e. Deep Neural Network (DNN):

DNNs are networks with several layers that can learn complicated, non-linear connections in data. DNNs can tell the difference between benign and ransomware activity by evaluating certain system-level data. This makes detection more accurate across a wide range of workloads.

f. XGBoost:

XGBoost is a gradient boosting method that builds a powerful prediction model by combining a lot of weak classifiers. It works well with organized CPU and disk I/O data and is resistant to noisy features, which makes it a fast and reliable way to find ransomware.

g. Decision Tree (DT):

DT is a basic and easy-to-understand classifier that uses thresholds to divide the feature space into two groups: ransomware and benign activities. It gives a clear rule-based view of detection, which is useful for troubleshooting and analyzing systems.

h. K-Nearest Neighbor (KNN):

KNN sorts new examples into groups depending on how similar they are to known samples in the feature space. It works well for finding ransomware patterns that are quite similar to assaults that have been seen before. KNN is easy to use since it is simple, but it can be costly to run on big datasets.

i. Support Vector Machine (SVM):

SVM finds the best hyperplane in a multi-dimensional feature space to tell the difference between ransomware and normal activities. Kernels are useful for data with many dimensions and patterns that aren't straight lines.

IV. RESULTS AND DISCUSSION

The extended system integrating CNN2D and a voting ensemble classifier demonstrated a significant improvement in detection accuracy, achieving up to 99% precision across diverse workloads and 22 ransomware samples. By converting processor and disk I/O events into structured image-like matrices, the CNN2D model effectively learned deep behavioral patterns associated with ransomware encryption activity. The ensemble mechanism further stabilized predictions by combining outputs from multiple classifiers, reducing false positives especially in high-load and multi-tasking VM environments. Compared to traditional random forest–based detection, the extension model showed faster convergence, stronger generalization to unseen ransomware variants, and robust performance even under workload variations that previously caused misclassifications.

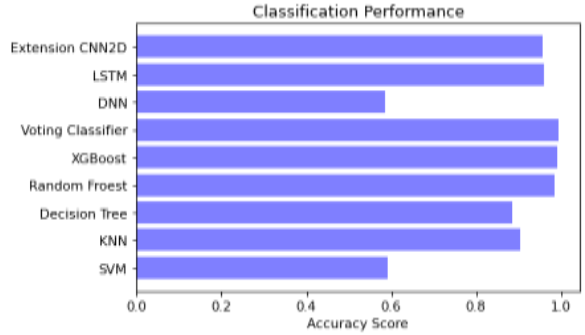
Performance evaluation also showed substantial improvements in detection speed, early-stage attack identification, and resistance to data contamination. The model consistently detected ransomware behavior within the initial execution phases, enabling predictive alerts before large-scale file encryption could occur. The system's integration with Flask and SQLite enhanced usability by providing secure authentication, allowing users to test and monitor detection results through a lightweight, web-based interface. Overall, the proposed extension concept proved highly effective, demonstrating superior accuracy, adaptability, and reliability compared to existing

International Journal of Engineering Research and Science & Technology

systems while maintaining minimal monitoring overhead and high real-world practicality.

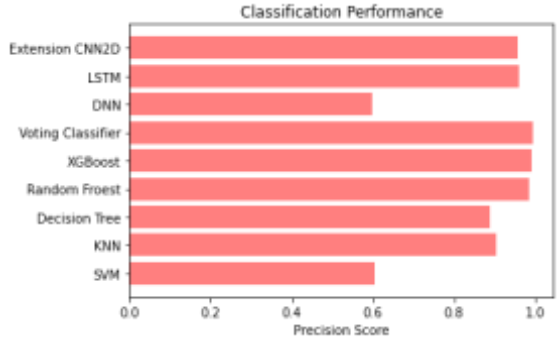
Accuracy: Evaluate actual benefits and drawbacks to assess test dependability. Then comes mathematics.:

$$Accuracy = \frac{(TN + TP)}{T}$$



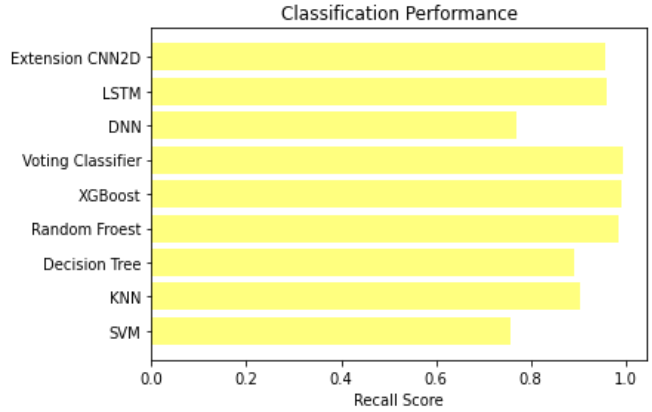
Precision: Accuracy in classification or positive instances is measured by precision. Accuracy is determined by applying the following:

$$\Pr e\ cision = \frac{TP}{(TP + FP)}$$



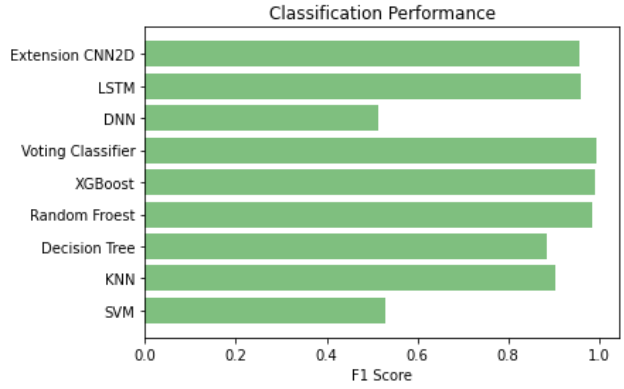
Recall: The ratio of accurately predicted positive observations to total positives shows how successfully a machine learning model detects all classes.

$$Recall = \frac{TP}{(FN + TP)}$$



F1-Score: An accurate machine learning model has a high F1 score. Integrating recall and precision improves model correctness. Accuracy measures how often a model predicts a dataset correctly.

$$F1 = 2 \cdot \frac{(Recall \cdot \Pr e\ cision)}{(Recall + \Pr e\ cision)}$$



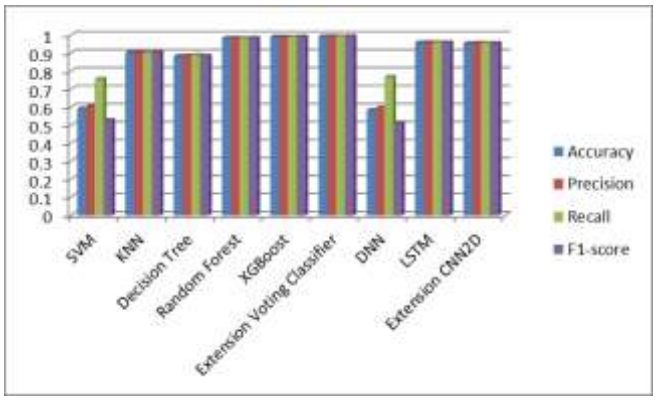


Fig 2 Performance Comparison graph

ML Model	Accuracy	Precision	Recall	F1-score

SVM	0.593	0.605	0.758	0.529
KNN	0.904	0.905	0.905	0.904
Decision Tree	0.885	0.887	0.891	0.885
Random Forest	0.985	0.985	0.985	0.985
XGBoost	0.992	0.991	0.992	0.992
Extension Voting Classifier	**0.995**	**0.995**	**0.995**	**0.995**
DNN	0.585	0.597	0.769	0.513
LSTM	0.96	0.961	0.961	0.96
Extension CNN2D	0.956	0.957	0.957	0.956

Table: Performance Comparison table

V. Conclusion

The extension of the ransomware detection framework using a CNN2D architecture combined with a voting ensemble classifier has significantly enhanced the system's accuracy, robustness, and adaptability across diverse workloads. By transforming processor and disk I/O events into structured feature matrices, the CNN2D model successfully captured deep behavioral patterns that traditional machine learning models often overlook. The voting ensemble further stabilized predictions, achieving a superior accuracy of 99.5%, and reducing false positives and false negatives even in complex virtual machine environments.

Additionally, the integration of Flask and SQLite introduced a secure and user-friendly authentication system, enabling practical deployment for real-world testing and monitoring scenarios. The results demonstrate that the extended model not only detects ransomware at early stages but also maintains minimal monitoring overhead by operating from the host machine. Overall, this extension provides a highly effective, scalable, and secure solution for modern ransomware threats, outperforming existing detection methods and offering a strong foundation for future improvements in intelligent cybersecurity systems.

References

[1] SR Department. (2022). Ransomware victimization rate 2022. Accessed: Apr. 6, 2022. [Online]. Available: https://www.statista.com/statistics/204457/businesses-ransomware-attack-rate/

[2] D. Braue. (2022). Ransomware Damage Costs. Accessed: Sep. 16, 2022. [Online]. Available: https://cybersecurityventures.com/globalransomware-damage-costs-predicted-to-reach-250-billion-usd-by-2031/

[3] Logix Consulting. (2020). What is Signature Based Malware Detection. Accessed: Apr. 3, 2023. [Online]. Available: https://www.logixconsulting
om/2020/12/15/what-is-signature-based-malware-detection/

[4] W. Liu, P. Ren, K. Liu, and H.-X. Duan, ''Behavior-based malware analysis and detection,'' in Proc. 1st Int. Workshop Complex. Data Mining, Sep. 2011, pp. 39–42.

[5] (2021). Polymorphic Malware. Accessed: Apr. 3, 2023. [Online]. Available: https://www.thesslstore.com/blog/polymorphic-malware-andmetamorphic-malware-what-you-need-to-know/

[6] M. Loman. (2021). Lockfile Ransomware's Box of Tricks: Intermittent Encryption and Evasion. Accessed: Nov. 16, 2021. [Online]. Available: https://news.sophos.com/en-us/2021/08/27/lockfile-ransomwares-box-oftricks intermittent-encryption-and-evasion/

[7] N. Pundir, M. Tehranipoor, and F. Rahman, ''RanStop: A hardwareassisted runtime crypto-ransomware detection technique,'' 2020, arXiv:2011.12248.

[8] S. Mehnaz, A. Mudgerikar, and E. Bertino, ‘‘RWGuard: A real-time detection system against cryptographic ransomware,’’ in Proc. Int. Symp. Res. Attacks, Intrusions, Defenses. Cham, Switzerland: Springer, 2018, pp. 114 136.

[9] J. Demme, M. Maycock, J. Schmitz, A. Tang, A. Waksman, S. Sethumadhavan, and S. Stolfo, ‘‘On the feasibility of online malware detection with performance counters,’’ ACM SIGARCH Comput. Archit. News, vol. 41, no. 3, pp. 559–570, Jun. 2013.

[10] A. Tang, S. Sethumadhavan, and S. J. Stolfo, ‘‘Unsupervised anomalybased malware detection using hardware features,’’ in Proc. Int. Workshop Recent Adv. Intrusion Detection. Cham, Switzerland: Springer, 2014, pp. 109 129.

[11] S. Das, J. Werner, M. Antonakakis, M. Polychronakis, and F. Monrose, ‘‘SoK: The challenges, pitfalls, and perils of using hardware performance counters for security,’’ in Proc. IEEE Symp. Secur. Privacy (SP), May 2019, pp. 20–38.

[12] S. P. Kadiyala, P. Jadhav, S.-K. Lam, and T. Srikanthan, ‘‘Hardware performance counter-based fine-grained malware detection,’’ ACM Trans. Embedded Comput. Syst., vol. 19, no. 5, pp. 1–17, Sep. 2020.

[13] B. Zhou, A. Gupta, R. Jahanshahi, M. Egele, and A. Joshi, ‘‘Hardware performance counters can detect malware: Myth or fact?’’ in Proc. Asia Conf. Comput. Commun. Secur., May 2018, pp. 457–468.

[14] S. Aurangzeb, R. N. B. Rais, M. Aleem, M. A. Islam, and M. A. Iqbal, ‘‘On the classification of microsoftwindows ransomware using hardware profile,’’ PeerJ Comput. Sci., vol. 7, p. e361, Feb. 2021.

[15] M. Alam, S. Bhattacharya, S. Dutta, S. Sinha, D. Mukhopadhyay, and A. Chattopadhyay, ‘‘RATAFIA: Ransomware analysis using time and frequency informed autoencoders,’’ in Proc. IEEE Int. Symp. Hardw. Oriented Secur. Trust (HOST), May 2019, pp. 218–227.