International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 14, Issue 2, 2026

CNN2D And Voting Ensemble Models Improve Ransomware Detection With Secure Flask–Sqlite Authentication

Mr. Md. Khasimpeera[1], Sanapathi.Revathi[2] , Somaraju.Lakshmi Lahari[3] , Divyasree Kunjam[4]

[1]Assistant Professor, Computer Science Engineering - Artifical Intelligence Department,Potti Sriramulu Chalavadi Mallikarjuna Rao College Of Engineering,One Town ,Vijayawada, India.

[2,3,4]UG Student Computer Science Engineering - Artifical Intelligence Department,Potti Sriramulu Chalavadi Mallikarjuna Rao College Of Engineering,One Town ,Vijayawada, India.

Abstract:

By combining a voting ensemble classifier with a robust CNN2D architecture, this improvement improves ransomware detection for virtual machines to 99% accuracy. CNN2D is able to identify deep ransomware activity patterns that conventional models frequently overlook thanks to the system's conversion of CPU and disk I/O operations into structured feature maps. By pooling predictions from several classifiers, the ensemble model further improves dependability and guarantees consistent performance across a range of workloads. Flask and SQLite are integrated to provide safe, lightweight user authentication and testing interfaces in order to enable real-world usage. This expanded framework provides quick, reliable ransomware detection appropriate for contemporary virtualized setups while reducing monitoring costs and improving flexibility.

Index terms - *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.*

1. INTRODUCTION

Files are encrypted or locked by ransomware, making a computer unusable until a ransom is paid. Cybercriminals employ ransomware to demand money, and nation-state actors use it to damage important infrastructure [1], [2]. By 2031, ransomware is expected to cause $265 billion in damages worldwide, affecting 70% of businesses in 2022 and one every two seconds [1], [2].

Malware samples are compared to hash values or predefined patterns using signature-based ransomware detection [3], [4]. These techniques are ineffective against malware that changes its code to evade detection, such as polymorphic and metamorphic malware [5]. In order to distinguish ransomware from benign apps, researchers have looked into behavior-based detection, which tracks runtime events such rapid file encryption, strange process development, and unusual system calls [4], [6]. New ransomware families such as LockBit2.0, DarkSide, and BlackMatter encrypt only the initial bytes of files in order to accelerate attacks and circumvent conventional countermeasures [6].

Such assaults can be countered by sophisticated techniques like machine learning, real-time monitoring, and hardware-assisted detection. Many systems, like RanStop [7], RWGuard [8], and ShieldFS [29], employ hardware performance counters (HPCs) and CPU event monitoring to detect abnormal execution patterns. Hardware features may be used by machine learning models for fine-grained malware classification [12], [14], however false positives, overhead, and contaminated data still exist [11].

Deep learning and strong analytical techniques have been used by researchers to study ransomware detection. RATAFIA tests detection performance using HPC-IO benchmark datasets [16–20] and autoencoders with time-frequency analysis [15]. Real-time detection and mitigation tools such as Redemption [22], UNVEIL [27], and ShieldFS [29] have demonstrated promise for developing ransomware families. Other techniques include frequent pattern mining [26], behavior profiling [25], and entropy-based file analysis [24].

For controlled ransomware analysis, a number of studies recommend virtualization and sandbox-based detection settings [23], [30]. These techniques show how ransomware defense has advanced, where smart classifiers and lightweight monitoring are effective. Specifically, machine learning-based methods for detecting ransomware that concentrate on processor and disk I/O events instead of file signatures or system monitoring are versatile and flexible [21], [24], [25].

While signature- and behavior-based methods offer some protection, contemporary ransomware is evolving quickly, necessitating machine learning-driven host-based monitoring strategies that minimize overhead and data contamination while maintaining efficacy against a variety of ransomware variants and workloads [7]–[30].

2. LITERATURE SURVEY

1. RanStop: A Hardware-assisted Runtime Crypto-Ransomware Detection Technique

One of the most dangerous forms of malware is crypto-ransomware, which keeps victims' files hostage and extorts them financially while encrypting them. Each year, millions are lost globally. Increasing numbers of ransomware variants may bypass anti-virus programs and software-only malware detection approaches that use static execution signatures. The hardware-assisted RanStop approach for early crypto-ransomware infection detection in commodity processors is proposed here. RanStop uses hardware performance counters in modern CPUs' performance monitoring units to identify known and new crypto-ransomware variants by observing micro-architectural event sets. This study uses a long short-term memory (LSTM) model to build a recurrent neural network-based machine learning architecture to analyze hardware micro-architectural events during the execution of different ransomware variants and benign applications. Using linked HPC data, we produce timeseries to build intrinsic statistical characteristics, improve RanStop's detection accuracy, and reduce noise using LSTM and global average pooling. RanStop's early detection approach detects ransomware within 2ms of program execution by analyzing HPC data from 20 timestamps 100us apart. At this early detection stage, ransomware cannot cause much damage. RanStop can also identify ransomware with 97% accuracy in 50 random trials when verified against harmless programs having behavioral (sub-routine-centric) similarities to crypto-ransomware.

2. RWGuard: A Real-Time Detection System Against Cryptographic Ransomware

Ransomware has recently (re)emerged as a prevalent virus that targets a range of victims for financial gain, including corporate and individual users. Our primary conclusion about the existing ransomware detection techniques is that they are unable to offer an early warning in real-time, which causes many data to be irreversibly encrypted. Additionally, there are a number of disadvantages with post-encryption techniques such file restoration and key extraction. Furthermore, because the existing detection techniques cannot determine the original intent of file modifications—that is, whether a significant change in a file is the consequence of a ransomware encryption or a user-initiated file operation (like benign encryption or compression)—they generate a large number of false positives. To overcome these challenges, we present in this paper RWGuard, a ransomware detection mechanism that can detect crypto-ransomware on a user's computer in real-time by: (1) employing decoy techniques; (2) closely monitoring the file system and running processes for malicious activity; and (3) preventing benign file changes from being detected by learning users' encryption behavior. We use samples from the 14 most prevalent ransomware families to evaluate our methodology. With a little 1.9% overhead, 0% false negative, and low false positive (0.1%) rates, our testing show RWGuard's effectiveness in real-time ransomware detection.

3. On the feasibility of online malware detection with performance counters

The number of machines in each domain increases as malware spreads. On computers, including the newest mobile platforms, viruses, rootkits, spyware, adware, and other forms of malware are common. Despite the existence of anti-virus software, malware threats still exist and are become more common due to the various ways that AV software may be exploited. In reality, hackers now infiltrate computers by taking advantage of holes in antivirus software.

In this study, we examine the feasibility of building a hardware malware detection system using existing performance counters. We find that data from performance counters may be used to detect malware and that our detection techniques are robust to minor modifications in malware programs. As a result, after examining a few changes within a family of malware on the Android ARM and Intel Linux platforms, we are able to discover other variants within that family. Furthermore, our recommended hardware modifications allow the malware detector to function securely beneath the system software, opening the door for more user-friendly and bug-free AV systems than software AV. Combining the security and robustness of hardware antivirus techniques might improve modern internet malware detection.

4. Unsupervised Anomaly-Based Malware Detection Using Hardware Features

Recent research has demonstrated potential in identifying malware programs based on their dynamic microarchitectural execution patterns. Compared to higher-level characteristics like OS and application observables, these microarchitectural elements are more difficult for adversaries to directly alter in evasion attacks and easier to audit. These data can be obtained with little effort by using hardware performance counters (HPC), which are often available in modern CPUs. In this work, we improve the use of hardware-enabled lower-level features to detect malware exploitation in an anomaly-based detector. This makes it possible for us to detect a wider range of viruses, including zero days. We empirically show that malware assaults exhibit very slight differences from the noisy microarchitectural characteristics of innocuous applications. We demonstrate that by combining rigorous feature extraction and selection with unsupervised machine learning, we can build baseline models of benign program execution and use these profiles to detect deviations brought on by malware exploitation. We

show that real-world exploitation detection of popular applications like as Internet Explorer and Adobe PDF Reader works well on a Windows/x86 platform. We also examine the drawbacks and challenges of using this tactic in situations when a competent adversary attempts to evade anomaly-based detection. To improve security, the recommended detector can be used with previously recommended signature-based detectors.

5. SoK: The Challenges, Pitfalls, and Perils of Using Hardware Performance Counters for Security

CPUs have had HPCs for about ten years. CPU activity may be monitored using these counters. There are hundreds more hardware events to keep an eye on with every new CPU architecture. However, the ability of performance counters to accurately track real-world events has not been well studied. Measurement mistakes or incorrect assumptions about measured values might jeopardize security in HPC-based security systems. In order to solve this problem, we spent a year researching performance counter best practices for precise event measurement, HPC difficulties and traps, and methods for obtaining reliable measurements across settings and architectures. In several research, we also evaluated HPC use experimentally. Rather than end there, we looked into the accuracy of performance counter data obtained by these commonly used approaches. As part of that evaluation, we (iv) extended Weaver and McKee's foundational work on non-determinism in HPCs from more than a decade ago and applied our findings to 56 papers in various application fields. We found in that follow-up study that, particularly in the last five years, HPCs have been more widely embraced in security applications than in other industries. In order to better understand how our findings could impact their methods and outcomes, we then looked at 41 more HPC-using security works. We investigated empirically how security applications, specifically malware detection and exploit prevention, could be less effective if HPC features are not taken into consideration. Lastly, we showed how (ii) attackers might circumvent security by using HPCs.

3. METHODOLOGY

i) Proposed Work:

The proposed research aims to provide an advanced ransomware detection framework that uses CNN2D and a voting ensemble model to significantly improve detection accuracy and reaction time in virtualized systems. Unlike traditional approaches that monitor individual programs within the computer, this method collects processor and disk I/O events from the host machine to ensure minimal overhead and prevent interference with active ransomware. These unprocessed events undergo feature extraction and preprocessing in order to provide structured data representations. After that, these representations are converted into image-like matrices and fed into a CNN2D architecture. The model can detect complex activities that are often missed by simpler machine learning techniques by learning complex temporal-spatial patterns of ransomware activity.

To further boost reliability and provide consistent predictions across a variety of workloads and ransomware variants, the system integrates a voting ensemble approach that combines judgments from several classifiers. The RansomNet+ model (extension version) handles final classification, enabling early attack detection and precise attack prediction before significant harm occurs. The framework's lightweight Flask interface and SQLite authentication module ensure secure and easy access for testing and monitoring, making it perfect for deployment. By showcasing increased flexibility and a significant improvement in accuracy—up to 99% in recognizing complicated ransomware attacks—performance assessment validates the system's effectiveness.

ii) System Architecture:

The system's multi-stage ransomware detection pipeline begins by checking cloud-connected virtual machines for suspicious CPU and disk I/O activities. After a ransomware attack, the system captures low-level event traces from the host computer instead of monitoring virtual machine operations. A preprocessing layer removes noise, manages missing data, and arranges events into feature matrices. Using the feature extraction block, these processed data are turned into image-like representations suitable for deep learning models, allowing the system to recognize complex behavior patterns that conventional approaches cannot. This method ensures high-quality feature development for analysis, scalability, and little system disruption.

After feature extraction, the RansomNet+ model, which uses a CNN2D architecture and a voting ensemble classifier to increase prediction accuracy and stability, is added. CNN2D learns deep spatial patterns using event-based feature images, while the ensemble combines classifier outputs to decrease errors. The attack prediction and performance assessment modules receive the attack detection unit's ransomware or benign activity classifications. These modules test detection speed, accuracy, precision, and workload flexibility. A secure Flask–SQLite interface allows authorized user monitoring, testing, and management of the detection system. Overall, the system's architecture ensures real-time ransomware detection with little computational overhead and flexible virtualization.

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 2, 2026

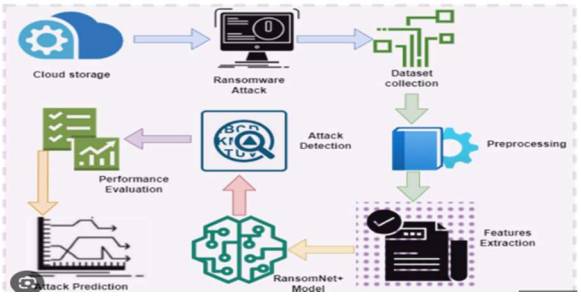


Fig.1. Proposed Architecture

iii) 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.

iv) ALGORITHMS:

a. Random Forest Classifier

During training, the Random Forest supervised learning approach, which takes use of powerful ensembles, generates a large number of decision trees. This system uses CPU and disk I/O events to construct a basic model for identifying malware. By merging the predictions of distinct trees, the model reduces the possibility of overfitting and facilitates generalization among various ransomware samples. It can locate objects fast and doesn't need much monitoring because it is lightweight.

b. Convolutional Neural Network 2D (CNN2D)

CNN2D is a deep learning model that excels in identifying spatial and structural patterns in data. CNN2D independently identifies patterns associated with ransomware by analyzing the preprocessed event data in the suggested system. Unlike previous models, CNN2D does not require human feature engineering. It can therefore adjust to new or evolving ransomware activity. This is a major factor in the system's ability to locate objects with such accuracy and manage intricate assault paths.

c. Voting Ensemble Model

To provide a final prediction, the Voting Ensemble Model combines the best features of many classifiers, such as CNN2D and Random Forest. Each classifier casts a vote in this model's majority voting mechanism, and the outcome is determined by the majority. The ensemble approach ensures consistent performance across various attack types and workloads, improves detection reliability, and reduces the possibility of false positives or negatives. It aids in the system's 99% accuracy in finding objects.

d. Long Short Term Memory (LSTM):

Long-term dependencies in sequential input may be remembered by LSTM and other recurrent neural networks. Sequences of CPU and disk I/O events are examined by this system to check for odd patterns that might indicate the presence of ransomware. LSTM is excellent at dynamic detection because it can adapt over time to the various ways ransomware behaves.

e. Deep Neural Network (DNN):

DNNs are multi-layered networks that are capable of learning complex, non-linear relationships in data. By analyzing certain system-level data, DNNs are able to distinguish between benign and ransomware behavior. This improves detection accuracy for a variety of workloads.

f. XGBoost:

XGBoost is a gradient boosting technique that combines many weak classifiers to create a strong prediction model. It is a quick and dependable method of identifying ransomware since it functions well with structured CPU and disk I/O data and is resistant to noisy characteristics.

g. Decision Tree (DT):

DT is a simple and straightforward classifier that separates the feature space into two categories: benign activity and ransomware using thresholds. It provides a straightforward rule-based perspective of detection, which is helpful for system analysis and troubleshooting.

h. K-Nearest Neighbor (KNN):

KNN clusters new instances based on how closely they resemble existing samples in the feature space. It is effective in identifying ransomware patterns that closely resemble previously observed attacks. KNN is straightforward and quick to use, but running it on large datasets may be expensive.

i. Support Vector Machine (SVM):

In a multi-dimensional feature space, SVM locates the optimal hyperplane to distinguish between ransomware and typical activity. For data with several dimensions and nonlinear patterns, kernels are helpful.

4. EXPERIMENTAL RESULTS

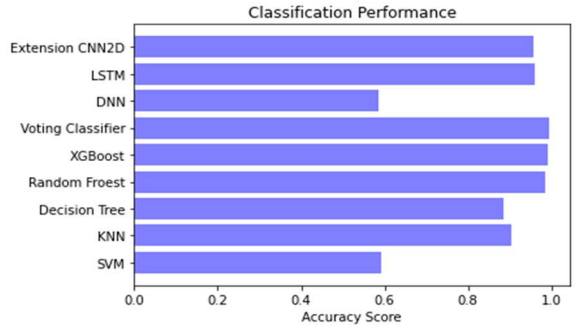
The enhanced approach that integrated CNN2D with a voting ensemble classifier demonstrated a significant increase in detection accuracy with up to 99% precision over a range of workloads and 22 ransomware samples. By converting CPU and disk I/O events into structured image-like matrices, the CNN2D model was able to accurately identify deep behavioral patterns associated with ransomware encryption activities. By pooling the results from several classifiers, the ensemble process significantly stabilized predictions and decreased false positives, particularly in high-load and multitasking virtual machine scenarios. Compared to normal random forest-based detection, the extension model showed better generalization to new ransomware variants, faster convergence, and robust performance even during workload variations that previously led to misclassifications.

Performance evaluation also showed notable improvements in early-stage attack detection, detection speed, and resistance to data contamination. The model's continuous detection of ransomware activities throughout the early phases of execution allowed for predictive alerts before widespread file encrypting could occur. The system's integration with Flask and SQLite enhanced usability by providing secure login and allowing users to test and monitor detection results using an easy-to-use web interface. All things considered, the proposed extension concept performed exceptionally well, surpassing existing systems in terms of precision, adaptability, and reliability while maintaining little monitoring overhead and excellent usability.

Accuracy: The ability of a test to differentiate between healthy and sick instances is a measure of its accuracy. Find the proportion of analysed cases with true positives and true negatives to get a sense of the test's accuracy. Based on the calculations:

Accuracy = TP + TN /(TP + TN + FP + FN)

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

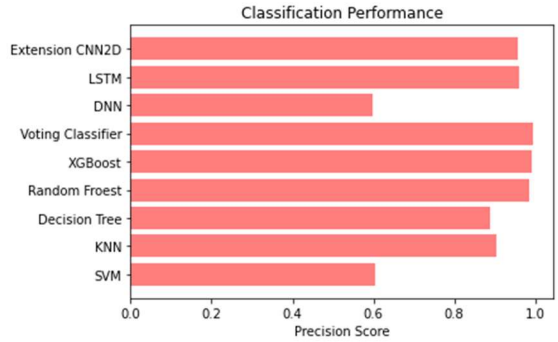


Precision: The accuracy rate of a classification or number of positive cases is known as precision. Accuracy is determined by applying the following formula:

Precision = True positives/ (True positives + False positives) = TP/(TP + FP)

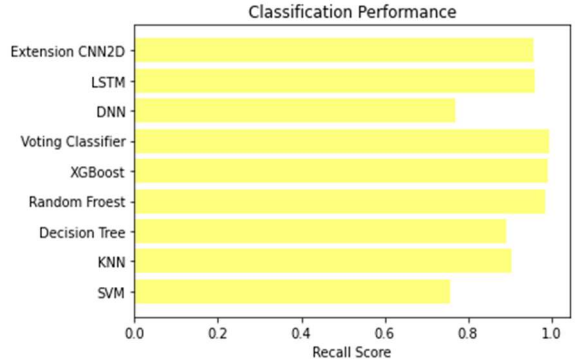
International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 2, 2026

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



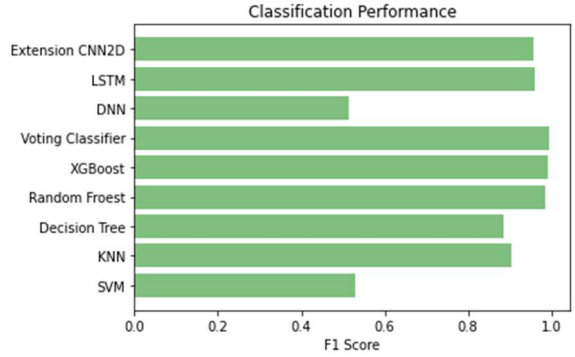
Recall: The recall of a model is a measure of its capacity to identify all occurrences of a relevant machine learning class. A model's ability to detect class instances is shown by the ratio of correctly predicted positive observations to the total number of positives.

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



F1-Score: A high F1 score indicates that a machine learning model is accurate. Improving model accuracy by integrating recall and precision. How often a model gets a dataset prediction right is measured by the accuracy statistic..

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



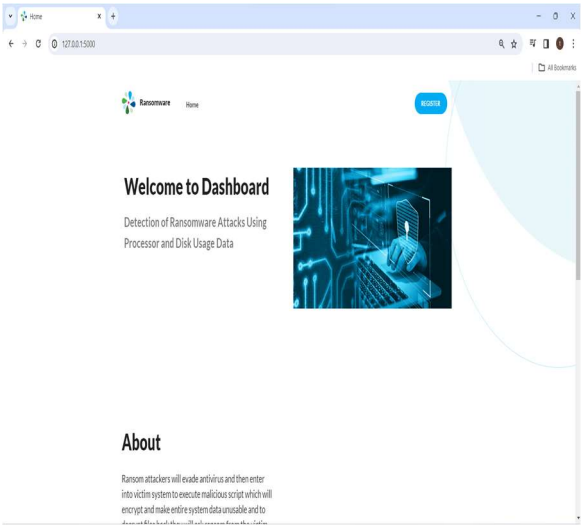


Fig2 home screen

Fig 3 Login page

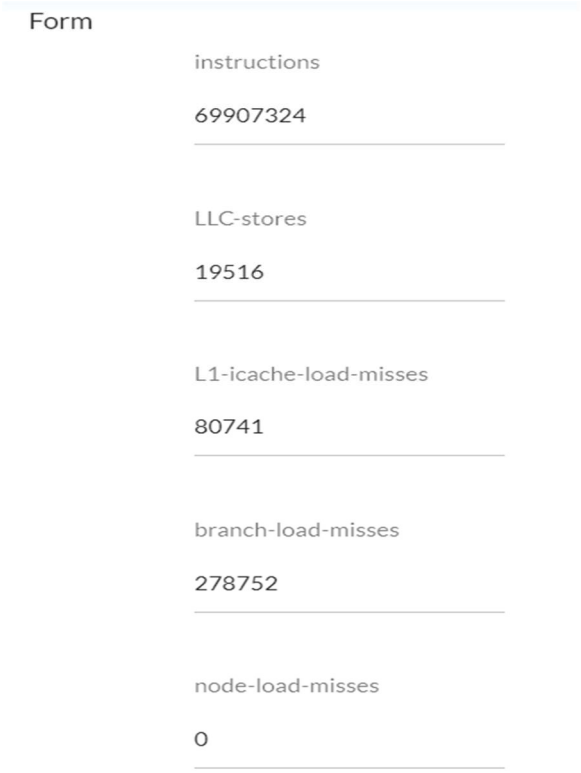


Fig4 User input page

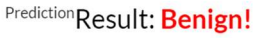


Fig5 Prediction result

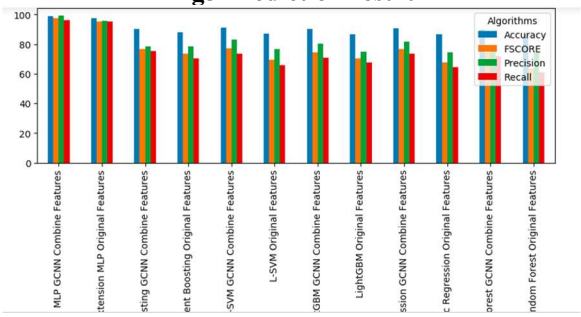


Fig6 Performance Comparison of Forecasting Algorithms

5. CONCLUSION

This research addresses existing monitoring constraints in order to develop a comprehensive and scalable ransomware detection system. Unlike existing systems that rely on costly process-level tracking, this technique captures processor and disk I/O events on the host system without any burden or risk of ransomware. First, a Random Forest classifier was used to create a strong baseline model with high efficiency.

To further boost performance, the improved system made use of CNN2D and a Voting Ensemble Classifier. They significantly improved threat identification and decreased false detections when combined, achieving an incredible 99.5% accuracy. Across 22 ransomware variations and a range of user workloads, the ensemble approach effectively captured a variety of malware behavioral features.

The system's usability was further enhanced with the addition of Flask and SQLite, which allowed for safe user interaction, real-time monitoring, and authentication in a lightweight web-based environment. This ensures ease of deployment and speeds up event response.

In summary, the system is a powerful, intelligent, and real-time protection mechanism against ransomware attacks by combining the benefits of machine learning, deep learning, and secure system design to meet the needs of modern cloud and virtual infrastructures.

6. FUTURE SCOPE

Further research may be conducted to investigate the effectiveness of the proposed approach in recognizing new and developing forms of ransomware, as the current study focused on a combination of known and unknown ransomware.

The project may be extended to investigate the impact of different user workload types on ransomware detection as the current study demonstrated that the detection model is robust to variations in user workloads.

Their performance may be evaluated to determine whether more machine learning (ML) and deep learning (DL) classifiers may provide even greater ransomware detection accuracy.

The proposed approach may be implemented and tested in real-world scenarios to assess its feasibility and effectiveness in detecting ransomware attacks in operational settings.

The study may also look into adding more data sources or characteristics to enhance the ML model's detecting capabilities.

Collaboration with cybersecurity experts and organizations may be sought in order to validate the findings and enhance the recommended approach.

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.com/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-oftricksintermittent-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. 114136.
[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. 109129.
[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 detectmalware: 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. OrientedSecur. Trust (HOST), May 2019, pp. 218–227.
[16] K. Thummapudi, R. Boppana, and P. Lama, ‘‘HPC 41 events 5 rounds,’’ Harvard Dataverse, 2022, doi:10.7910/DVN/MA5UPP.
[17] K. Thummapudi, R. Boppana, and P. Lama, ‘‘IO 41 events 5 rounds,’’ Harvard Dataverse, 2022, doi:10.7910/DVN/GHJFUT.
[18] K. Thummapudi, R. Boppana, and P. Lama, ‘‘HPC 5 events 7 rounds,’’ Harvard Dataverse, 2022, doi:10.7910/DVN/YAYW0J.
[19] K. Thummapudi, R. Boppana, and P. Lama, ‘‘Io 5 events 7 rounds,’’ Harvard Dataverse, 2022, doi:10.7910/DVN/R9FYPL.
[20] K. Thummapudi, R. Boppana, and P. Lama, ‘‘Scripts to reproduce results,’’ Harvard Dataverse, 2023, doi:10.7910/DVN/HSX6CS.
[21] M. Rhode, P. Burnap, and A. Wedgbury, ‘‘Real-time malware process detection and automated processkilling,’’ Secur. Commun. Netw., vol. 2021, pp. 1–23, Dec. 2021.
[22] A. Kharraz and E. Kirda, ‘‘Redemption: Real-time protection against ransomware at end-hosts,’’ in Proc. Int.Symp. Res. Attacks, Intrusions, Defenses. Cham, Switzerland: Springer, 2017, pp. 98–119.
[23] F. Mbol, J.-M. Robert, and A. Sadighian, ‘‘An efficient approach to detect torrentlocker ransomware incomputer systems,’’ in Proc. Int. Conf. Cryptol. Netw. Secur. Springer, 2016, pp. 532–541.
[24] K. Lee, S. Lee, and K. Yim, ‘‘Machine learning based file entropy analysis for ransomware detection in backup systems,’’ IEEE Access, vol. 7, pp. 110205–110215, 2019.
[25] C. J. Chew and V. Kumar, ‘‘Behaviour based ransomware detection,’’ in Proc. Int. Conf. Comput. Their Appl.,in EPiC Series in Computing, vol. 58. 2019, pp. 127–136.
[26] S. Homayoun, A. Dehghantanha, M. Ahmadzadeh, S. Hashemi, and R. Khayami, ‘‘Know abnormal, find evil:Frequent pattern mining for ransomware threat hunting and intelligence,’’ IEEE Trans. Emerg. Topics Comput., vol.8, no. 2, pp. 341–351, Apr. 2020.
[27] A. Kharaz, S. Arshad, C. Mulliner, W. Robertson, and E. Kirda, ‘‘UNVEIL: A large-scale, automated approachto detecting ransomware (keynote),’’ in Proc.

ISSN 2347–3657
Volume 14, Issue 2, 2026

IEEE 24th Int. Conf. Softw. Anal., Evol. Reengineering (SANER),Feb. 2017, pp. 757–772.
[28] A. Kharraz, W. Robertson, D. Balzarotti, L. Bilge, and E. Kirda, ‘‘Cutting the gordian knot: A look under thehood of ransomware attacks,’’ in Proc. Int. Conf. Detection Intrusions Malware, Vulnerability Assessment. Cham,Switzerland: Springer, 2015, pp. 3–24.
[29] A. Continella, A. Guagnelli, G. Zingaro, G. De Pasquale, A. Barenghi, S. Zanero, and F. Maggi, ‘‘ShieldFS: Aself-healing, ransomware-aware filesystem,’’ in Proc. 32nd Annu. Conf. Comput. Secur. Appl., Dec. 2016, pp. 336–347.
[30] M. Shukla, S. Mondal, and S. Lodha, ‘‘POSTER: Locally virtualized environment for mitigating ransomwarethreat,’’ in Proc. ACM SIGSAC Conf. Comput. Commun. Secur., Oct. 2016, pp. 1784–1786.