Improved Ransomware Detection Utilizing CNN2D and Voting Ensemble Models with Secure Flask–SQLite Authentication

[1]LAKKAM RANGASWAMY, [2]R HENDRA KUMAR

[1]PG Scholar, Dept. of CSE, KMM Institute of Technology & Science, Ramireddipalle, Tirupati, AP, India.

[2]Associate Professor, Dept. of CSE, KMM Institute of Technology & Science, Ramireddipalle, Tirupati, AP, India.

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.

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

Ransomware encrypts or locks files, rendering a computer useless until a ransom is paid. Ransomware is used by cybercriminals to extort money, and nation-state actors to harm key infrastructure [1], [2]. Global ransomware losses are anticipated to reach $265 billion by 2031, with 70% of firms experiencing instances in 2022 and one every two seconds by 2031 [1], [2].

Signature-based ransomware detection matches malware samples against hash values or predetermined patterns [3], [4]. These methods fail against polymorphic and metamorphic malware, which change their code to avoid detection [5]. Thus, researchers have investigated behavior-based detection, which monitors runtime events such quick file encryption, odd process formation, and anomalous system calls to identify ransomware from innocuous applications [4], [6]. To speed up assaults and evade standard defenses, new ransomware families like LockBit2.0, DarkSide, and BlackMatter encrypt just the first bytes of files [6].

Advanced methods including hardware-assisted detection, real-time monitoring, and machine learning can combat such attacks. Many systems use hardware performance counters (HPCs) and CPU event monitoring to identify aberrant execution patterns, such as RanStop [7], RWGuard [8], and ShieldFS [29]. Machine learning models may use hardware characteristics for fine-grained malware classification [12], [14], but data contamination, overhead, and false positives remain [11].

Researchers have studied ransomware detection using deep learning and powerful analytical methods. RATAFIA uses autoencoders with time-frequency analysis [15] and HPC-IO benchmark datasets [16–20] to test detection performance. For emerging ransomware families, real-time detection and mitigation technologies like Redemption [22], UNVEIL [27], and ShieldFS [29] have showed potential. Entropy-based file analysis [24], behavior profiling [25], and frequent pattern mining [26] are further methods.

Several studies suggest virtualization and sandbox-based detection settings for controlled ransomware analysis [23], [30]. These methods demonstrate the progress of ransomware defense, where lightweight monitoring and sophisticated classifiers work well. In particular, machine learning-based ransomware detection approaches that focus on processor and disk I/O events rather than file signatures or system monitoring are flexible and adaptable [21], [24], [25].

Signature- and behavior-based approaches provide partial protection, but modern ransomware evolves rapidly, requiring machine learning-driven host-based monitoring techniques that reduce overhead, data contamination, and remain effective against diverse ransomware variants and workloads [7]–[30].

2. LITERATURE SURVEY

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

Crypto-ransomware is one of the most common types of malware that poses a serious danger since it holds users' papers hostage and extorts them monetarily while also establishing denial of access through unlawful encryption of their documents. Globally, this causes losses of millions of dollars per year. The number of ransomware variations that can evade several anti-virus programs and software-only malware detection methods that depend on static execution signatures is increasing. In this work, we suggest RanStop, a hardware-assisted method for early crypto-ransomware infection detection in commodity processors. RanStop observes micro-architectural event sets and identifies known and undiscovered crypto-ransomware variants using data from hardware performance counters integrated into the performance monitoring unit of contemporary CPUs. In this study, we use a long short-term memory (LSTM) model to train a recurrent neural network-based machine learning architecture for the analysis of micro-architectural events in the hardware domain during the execution of several ransomware variants and benign programs. Using the data from connected HPCs, we generate timeseries to establish intrinsic statistical features, enhance RanStop's detection accuracy, and lower noise using LSTM and global average pooling. By examining HPC data gathered at 20 timestamps spaced 100us apart, RanStop's early detection method can reliably and swiftly detect ransomware within 2ms of the program's execution. A ransomware cannot do much, if any, harm at this early detection stage. Furthermore, RanStop can identify ransomware with an average of 97% accuracy for fifty random trials when validated against innocuous programs that have behavioral (sub-routine-centric) similarities with a crypto-ransomware.

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

Recently, ransomware has (re)emerged as a common virus that targets a variety of victims for financial gain, from individual users to corporate ones. Our main finding regarding the current ransomware detection methods is that they are unable to provide an early warning in real-time, which leads to the irreversible encryption of a large number of files. Additionally, the post-encryption methods (such as file restoration and key extraction) have various drawbacks. Additionally, the current detection methods produce a high number of false positives because they are unable to ascertain the original intent of file modifications; that is, they are unable to discern whether a notable alteration in a file is the result of a ransomware encryption or a user-initiated file operation (such as benign encryption or compression). In order to overcome these difficulties, we present in this paper RWGuard, a ransomware detection mechanism that can identify crypto-ransomware on a user's computer in real-time by: (1) using 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 test our approach using samples from the 14 most common ransomware families to date. Our tests demonstrate RWGuard's efficacy in real-time ransomware detection, with a little 1.9% overhead and zero false negative and low false positive (0.1%) rates.

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

Malware spreads together with the number of machines in each given domain. Viruses, rootkits, spyware, adware, and other types of malware are prevalent on systems, even the newest mobile platforms. Malware threats continue to exist despite the availability of anti-virus software, and they are becoming more prevalent because to the numerous ways that AV software may be compromised. In actuality, hackers now use flaws in antivirus software to compromise computers.

In this research, we investigate the viability of utilizing current performance counters to construct a hardware malware detection. We discover that our detection methods are resilient to small changes in malware programs and that data from performance counters may be utilized to identify malware. As a consequence, we are able to identify several variants within a family of malware on the Android ARM and Intel Linux platforms after looking at a small number of variations within that family. Additionally, our suggested hardware changes enable the malware detector to operate safely underneath the system software, paving the way for AV systems that are easier to use and less prone to bugs than software AV. Modern internet malware detection might be advanced by combining the security and resilience of hardware antivirus methods.

4. Unsupervised Anomaly-Based Malware Detection Using Hardware Features

The detection of malware programs based on their dynamic microarchitectural execution patterns has showed promise in recent study. These microarchitectural aspects are easier to audit and more difficult for adversaries to directly influence in evasion assaults than higher-level features like OS and application observables. Hardware performance counters (HPC), which are frequently accessible in contemporary CPUs, can be used to get these statistics at little overhead. In this study, we enhance the usage of lower-level characteristics enabled by hardware to identify malware exploitation in an anomaly-based detector. This enables us to identify a greater variety of malware, including zero days. We empirically demonstrate that malware attacks display very small variations from the microarchitectural features of benign programs, which are noisy. We show that using unsupervised machine learning in conjunction with rigorous feature extraction and selection, we can create baseline models of benign program execution and utilize these profiles to identify variations caused by malware exploitation. We demonstrate that on a Windows/x86 platform, real-world exploitation detection of well-known apps like Internet Explorer and Adobe PDF Reader functions effectively. We also look at the limitations and difficulties of applying this strategy when a skilled adversary tries to avoid anomaly-based detection. The suggested detector can be used in conjunction with previously suggested signature-based detectors to increase security.

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

For almost a decade, CPUs have had HPCs. These counters can track CPU activities. Each new CPU architecture adds hundreds of hardware events to monitor. However, few systematic research have examined how performance counters can reliably track real-world occurrences. In security applications using HPCs, measurement errors or inaccurate assumptions about measured values might compromise protection. We spent a year studying performance counter best practices for accurate event measurement, HPC challenges and pitfalls, and ways to get consistent and accurate measurements across settings and architectures to address this issue. We also experimentally assessed HPC utilization in many studies. Instead of stopping there, we investigated if these frequently used methods are accurately obtaining performance counter data. We (iv) expanded Weaver and McKee's fundamental work from over 10 years ago on non-determinism in HPCs and applied our findings to 56 articles across diverse application domains as part of that assessment. In that follow-up study, we observed that HPCs have been accepted in security applications more than in other sectors, notably in the previous five years. We next examined an additional 41 HPC-using security works to further understand how our findings potentially affect their methodologies and results. We empirically examined how failing to account for HPC peculiarities might reduce the efficacy of security applications, particularly exploit prevention and malware detection. Finally, we demonstrated how (ii) adversaries can exploit HPCs to overcome security.

3. METHODOLOGY

i) Proposed Work:

In order to greatly increase detection accuracy and reaction time in virtualized systems, the proposed project intends to develop a sophisticated ransomware detection framework that makes use of CNN2D and a voting ensemble model. This method ensures minimum overhead and avoids interference from active ransomware by gathering processor and disk I/O events from the host machine, in contrast to standard methods that monitor individual processes inside the machine. To create structured data representations, these raw events go through preprocessing and feature extraction. These representations are then transformed into matrices that resemble images and fed into a CNN2D architecture. Through the acquisition of intricate temporal-spatial patterns of ransomware activity, the model is able to identify intricate behaviors that are frequently overlooked by more straightforward machine learning methods.

The method incorporates a voting ensemble approach that aggregates judgments from many classifiers to further increase dependability, guaranteeing consistent predictions over a range of workloads and ransomware variations. Final categorization is handled by the RansomNet+ model (extension version), which facilitates early attack detection and accurate attack prediction prior to major harm. The framework is ideal for deployment because of its lightweight Flask interface and SQLite authentication module, which guarantee safe and convenient access for testing and monitoring. Performance evaluation confirms the efficacy of the system by demonstrating improved flexibility and a notable increase in accuracy—up to 99% in identifying complex ransomware assaults.

ii) System Architecture:

Monitoring cloud-connected virtual machines for questionable CPU and disk I/O activity is the first step in the system's organized multi-stage ransomware detection pipeline. Instead of monitoring operations within the affected virtual machine, the system records low-level event traces from the host computer when a ransomware attack takes place. These gathered events go through a preprocessing layer that handles missing data, eliminates noise, and organizes the event stream into feature matrices. These processed data are transformed into image-like representations appropriate for deep learning models using the feature extraction block, allowing the system to identify intricate behavior patterns that conventional methods are unable to identify. This approach guarantees high-quality feature creation for additional analysis, scalability, and little system interruption.

The RansomNet+ model, which combines a CNN2D architecture with a voting ensemble classifier to improve prediction accuracy and stability, is integrated into the system after feature extraction. While the ensemble combines outputs from several classifiers to reduce mistakes and false detections, the CNN2D component employs event-based feature pictures to learn deep spatial patterns. The attack prediction and performance assessment modules receive the findings of the attack detection unit's classification of events as either benign activity or ransomware. These modules examine detection speed, precision, accuracy, and workload flexibility. To provide authorized user access for monitoring, testing, and controlling the detection system, a secure Flask–SQLite interface is included. All things considered, the system's architecture guarantees accurate, real-time ransomware detection while preserving little computing overhead and robust flexibility across various virtual settings.

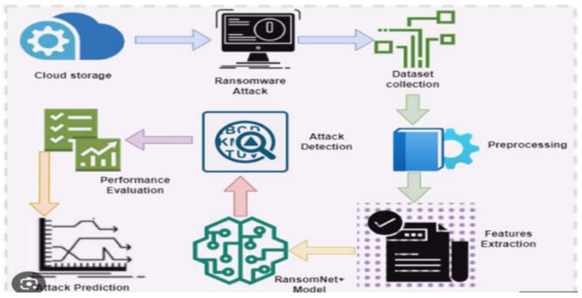


Fig.1. Proposed Architecture

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

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.

4. EXPERIMENTAL RESULTS

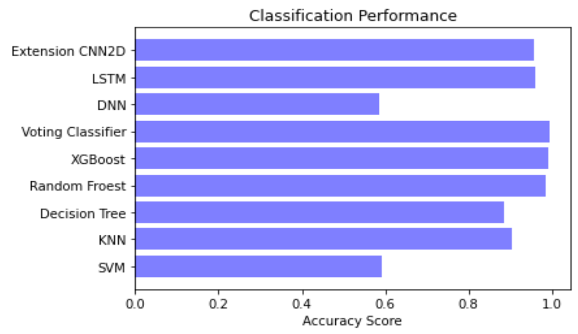
With up to 99% precision across a variety of workloads and 22 ransomware samples, the expanded system that combined CNN2D with a voting ensemble classifier showed a notable boost in detection accuracy. The CNN2D model successfully identified deep behavioral patterns linked to ransomware encryption activities by transforming CPU and disk I/O events into structured image-like matrices. The ensemble mechanism further stabilized predictions by combining outputs from multiple classifiers, reducing false positives especially in high-load and multi-tasking VM environments. The extension model demonstrated higher generalization to undiscovered ransomware variants, faster convergence, and resilient performance even during workload fluctuations that previously resulted in misclassifications as compared to standard random forest-based detection.

Significant gains in early-stage assault detection, detection speed, and resistance to data contamination were also demonstrated by performance evaluation. Predictive notifications before widespread file encryption could take place were made possible by the model's constant detection of ransomware activity throughout the early stages of execution. By offering secure login and enabling users to test and track detection findings via a simple, web-based interface, the system's combination with Flask and SQLite improved usability. Overall, the suggested extension idea worked very well, outperforming current systems in terms of accuracy, flexibility, and dependability while retaining low monitoring overhead and great 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)

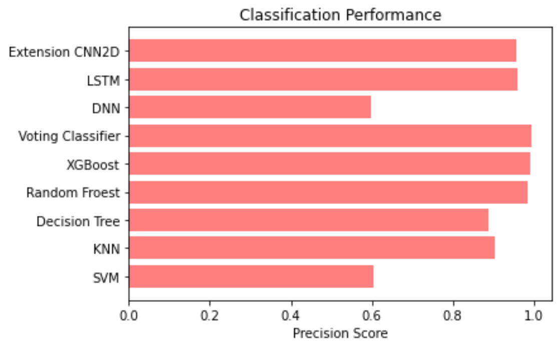
$$Accuracy = \frac{(TN + TP)}{T}$$



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

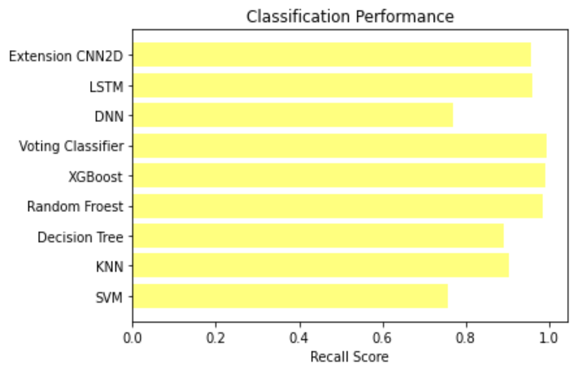
Precision = True positives/ (True positives + False positives) = TP/(TP + FP)

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



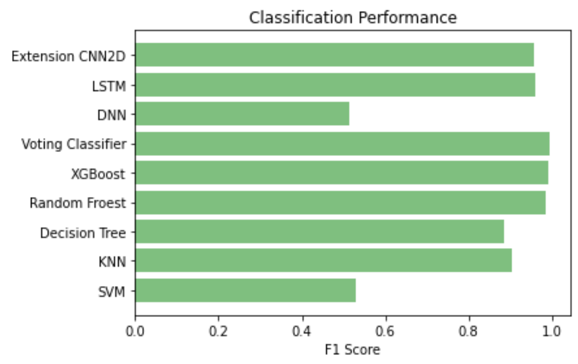
Recall: The recall of a model is a measure of its capacity to identify all occurrences of a relevant machine learning class. A model's ability to detect class instances is shown by the ratio of correctly predicted positive observations to the total number of positives.

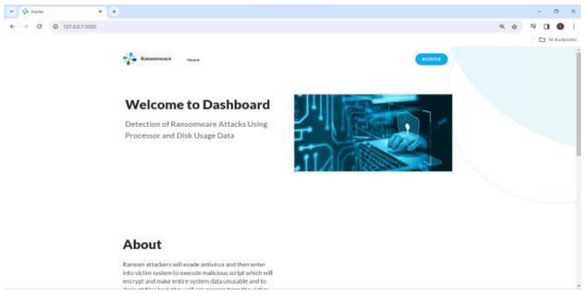
$$Recall = \frac{TP}{(FN + TP)}$$



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..

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





Welcome to Dashboard

Detection of Ransomware Attacks Using Processor and Disk Usage Data

About

Ransom attackers will evade antivirus and then enter into victim system to execute malicious script which will encrypt and make entire system data unusable and to

Fig2 home screen

SignIn

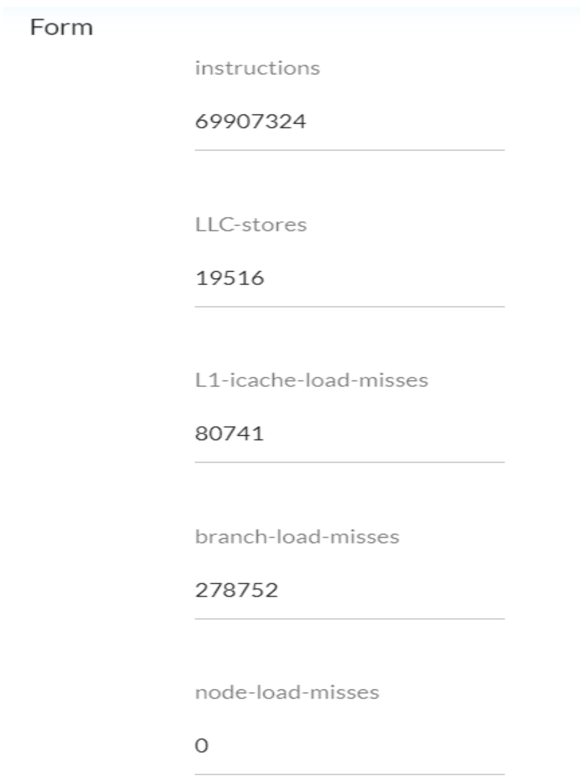
admin

•••••

SIGN IN

Register here!Sign Up

Fig 3 Login page



Form

instructions

69907324

LLC-stores

19516

L1-icache-load-misses

80741

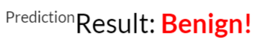
branch-load-misses

278752

node-load-misses

0

Fig4 User input page



PredictionResult: Benign!

Fig5 Prediction result

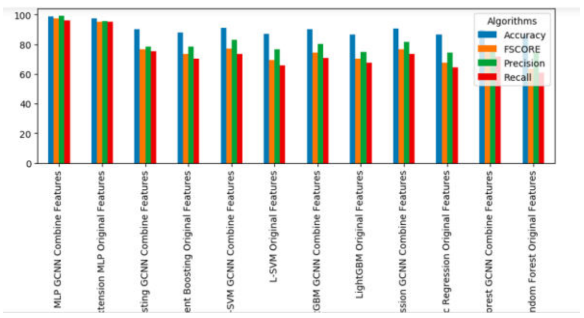


Fig6 Performance Comparison of Forecasting Algorithms

5. CONCLUSION

In order to create a thorough and scalable ransomware detection system, this project solves current monitoring limitations. This method records processor and disk I/O events on the host system without any overhead or danger of ransomware, in contrast to current solutions that rely on expensive process-level tracking. Initially, a robust baseline model with excellent efficiency was obtained by using a Random Forest classifier.

The enhanced system used CNN2D and a Voting Ensemble Classifier to further improve performance. Together, they achieved an astounding 99.5% accuracy, greatly enhancing threat identification and reducing false detections. The ensemble technique successfully captured a variety of ransomware behavioral characteristics across 22 ransomware variants and varied user workloads.

Additionally, the addition of Flask and SQLite improved the system's usability by enabling real-time monitoring, secure user interaction, and authentication in a lightweight web-based setting. This guarantees deployment simplicity and facilitates quick incident response.

In conclusion, the system combines the advantages of machine learning, deep learning, and safe system architecture to match the demands of contemporary cloud and virtual infrastructures, proving to be a potent, intelligent, and real-time defense mechanism against ransomware assaults.

6. FUTURE SCOPE

Since the current study concentrated on a combination of known and unknown ransomware, more research may be done to examine the efficacy of the suggested technique in identifying new and emerging varieties of ransomware.

Since the present study showed that the detection model is resilient to changes in user workloads, the project may be expanded to examine the effects of other user workload types on ransomware detection.

To find out if additional machine learning (ML) and deep learning (DL) classifiers can offer even higher ransomware detection accuracy, their performance may be assessed.

To evaluate the suggested method's viability and efficacy in identifying ransomware assaults in operational settings, it may be put into practice and evaluated in actual situations.

In order to improve the ML model's detection skills, the research may additionally investigate the incorporation of further data sources or attributes.

To verify the results and improve the suggested strategy, cooperation with cybersecurity specialists and organizations might be sought.

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. 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.