International Journal of Informatics, Information System and Computer Engineering

# Testing Deep Learning Methods to Predict Ransomware Activity from Hybrid Analysis

*Alexander Veach*, Munther Abualkibash **

*Eastern Michigan University, USA
*Corresponding Email: aveach1@emich.edu

**ABSTRACTS**

This article focuses on using deep learning methods to predict ransomware from hybrid analysis samples. Other similar research is analysed to understand the common methods of detection used to predict ransomware using various methods of analysis. By using this knowledge an experiment is created which tests the performance of a model created from hybrid analysis of ransomware samples. The training dataset used is made up of more than five hundred samples containing 38 different ransomware families and benign Windows program samples. The resultant model was then tested against a dataset include ransomware families not represented in the training dataset, which showed a decrease in performance. These results were then compared to other research's reported results which highlights potential issues in the way artificial intelligence models are tested and reported. The paper then proposes a focus on more complex methods of prediction, and other potential methods to ensure the models created are externally as effective as they report.



**ARTICLE INFO**

***Article History:***
*Received 14 Dec 2024*
*Revised 04 Feb 2025*
*Accepted 10 Mar 2025*
*Available online 14 Mar 2025*
*Publication date 01 June 2026*

***Keywords:***
***Machine Learning,***
***Artificial Intelligence,***
***Ransomware,***
***Model Drift,***
***Hybrid Analysis***

DOI: https://doi.org/10.34010/injiiscom.v7i1.14803
p-ISSN 2810-0670 e-ISSN 2775-5584

## 1. INTRODUCTION

Ransomware is a ubiquitous threat in the modern digital landscape. Stories about ransomware attacks disabling critical infrastructure or important business processes have grown more common with each passing year. To combat this growing threat many have looked to automated detection systems powered by Artificial Intelligence (AI) to anticipate and react before irreparable damage has been caused. To that end, this paper tests a commonly proposed method of detection using an artificial neural network (ANN) trained on ransomware activity.

Other research has shown the potential of artificial intelligence when it comes to predicting ransomware activity if properly configured, however research into this topic often report training metrics with high confidence values. While this can imply that a model is efficient, it can also be a sign that the model may have negatives related to the data gathered and the methods used. Alongside this, the aspect of model decay can be overlooked in research. This paper seeks to replicate the effects of this against the commonly proposed models by creating a testing dataset that contains new ransomware samples not represented in the training dataset.

The objective of this research is to test a deep learning model trained on ransomware data gathered from hybrid analysis and compare to other studies to see if there is any difference when accounting for ransomware not represented in the dataset to test the effects of model drift. By understanding showcasing the effect of drift on a model, and the ways to potentially reduce it future research can better understand and account for it.

## 2. METHODOLOGY

### 2.1. Concepts and Related Work

Ransomware is a specific type of malware, and the most common ransomware and the one most often mentioned in related literature is that of cryptographic ransomware. This type of ransomware operates by infiltrating a system, getting access to higher level commands on the network or device by various means, and then encrypting data in a way that forces the victim to pay a ransom for a decryption key or to initiate a system recovery plan.

When it comes to detecting ransomware there are a couple different methods that have been used. Some methods such as those used by Hossain et al. (2022), Ghazi and Raghava (2022), and Almousa et al. (2021) operated based on network communication, focusing on detecting anomalous traffic on a network to identify ransomware. This method relies on the ransomware operating in non-standard ways once it infiltrates a network, which can lead to issues in classification if the network traffic generated by the ransomware is different from what is represented in the dataset.

Another method of detection focuses on detecting high levels of Shannon entropy which is the amount of uncertainty in the potential states of a piece of data, specifically applied to encrypted files and their predicted decrypted outputs. This requires analysis of encrypted files so the trained model can identify and stop the potential ransomware associated with this feature. Hirano and Kobayashi (2019) and Hsu et

DOI: https://doi.org/10.34010/injiiscom.v7i1.14803
p-ISSN 2810-0670 e-ISSN 2775-5584

al. (2021) used Shannon entropy with other features for their decision-making models. Shannon entropy is useful for detecting files with unreliable encryption, however other studies such as Moussaileb et al. (2021) mention that images and other complex files can be falsely identified with this method.

Other features commonly used in detection are API calls considered related to ransomware activity as used by Anand et al. (2022), a combination of behaviours as outlined by Alzahrani et al. (2019), or via detecting changes in disk and memory activity as tested by Melaragno and Casey (2022).

There are also different methods of ransomware analysis which is used to generate the datasets used in training and testing for AI. Static analysis focuses on parsing the code contained in the program and returns derived inferences. This method is easy to execute and reduces the complexity of analysis needed for real world testing. Sharma and Sangal (2023) focused on static feature analysis for their work in detecting Android malware.

Dynamic analysis returns information gathered not by parsing the program's code, but rather by storing and analysing the actions taken in a sandboxed operating system. This returns information generated from runtime activity which offers a more in-depth data of ransomware activity. In exchange, this method requires a sandbox for testing and more time to generate the data used in prediction. Sukul et al. (2022) used dynamic analysis to gather runtime data of common ransomware samples for dataset creation and prediction.

Finally, there is hybrid analysis which focuses on gathering information on ransomware using a combination of static and dynamic analysis getting the information from both, at the cost of more time to generate the necessary data for prediction. Iqbal et al. (2022) used hybrid analysis techniques to gather static program data alongside active capture of Android devices screens to detect threatening messages to predict ransomware.

General reviews into the topic have noticed some major aspects of research that are problematic if unaddressed. Davies et al. (2021) specifically mentioned that some of the current features used in detection like Shannon entropy being questioned in other literature for effectiveness. Oz et al. (2022) focuses on the how much of the variance is covered in the theoretical defensive solutions, and how models overly focused on one aspect can be overcome by designing ransomware that avoids using the methods being detected. Razaulla et al. (2023) specifically mentions the issue of concept drift in relation to prediction accuracy against topics that undergo change over time.

Based on the information gathered, the design of the experiment will use hybrid analysis of multiple factors to limit overfitting based on a single dimension. The training and testing set are designed to represent an external environment with ransomware samples not represented in training being used in the testing dataset. The results will be compared against other reported values for similar models to see if there is any clear differences.

### 2.2. Gathering samples for dataset creation

DOI: https://doi.org/10.34010/injiiscom.v7i1.14803
p-ISSN 2810-0670 e-ISSN 2775-5584

To create the dataset used for training and testing, hybrid analysis of ransomware and benign samples were necessary. Samples were gathered from malware repositories such as VX-Underground and Malware Bazaar, specifically targeting recent ransomware families such as REvil, Akira, alongside historic ransomware samples such as Locky. From this process, five hundred and eighty-three samples were gathered from thirty-nine different families. Sixty benign samples made up of common Windows productivity software.

### 2.3. Analysing the collected samples

To analyse these samples CAPEv2 (O'Reilly, 2024), a Cuckoo Sandbox fork, was used for hybrid analysis. This was done within an Ubuntu 22.04 LTS virtual machine hosted on a Windows 11 desktop. The desktop's specifications are as follows: an AMD Ryzen 9 5900x 12-core processor, sixty-four gigabytes of Random Access Memory (RAM), an Nvidia GeForce 3080 graphics processing unit, and a terabyte of solid-state storage. The virtual machine instance had thirty-two gigabytes of RAM, twelve virtualized processor cores, and six hundred gigabytes of storage.

Using CAPEv2's default analysis toolset, each ransomware produced a dynamic analysis log which contained the actions taken by the software sample executed in the sandbox. CAPEv2 also did static analysis on the samples tested, which it derived the signatures detected in the sample. Further information produced by this analysis includes a list of APIs used, the user accounts used, registry keys accessed, and YARA detections from the systems memory. A JavaScript Object Notation (JSON) report was then output for each sample processed.

### 2.4. Deriving features from the JSON reports

Taking the JSON reports, Python 3 was used to parse and select features related to prediction. These features include the Application Programming Interface (API) calls executed during the runtime, the registry keys accessed, the signatures recognized during static analysis, the number of YARA detections, and operations executed. API calls were divided by the API called and the number of calls executed during analysis. The signatures and registry keys were transformed into nominal values from strings using WEKA (Frank, 2016) 3.9.6's StringToWordVector method.

This created a dataset with one thousand and seventeen features, including the value of "safe or unsafe" which represented safe as a value of zero and unsafe as a value of one.

While the prior methods created a dataset for training, to test the performance of the model against drift a separate dataset for testing containing a mixture of ransomware families represented during training, and newer ransomware not included in training such as Blacksuit. This testing dataset was created using the same methods prior. Due to the use of StringToWordVector from WEKA, some signatures represented in the testing set were not included in the training set. This was accounted for in testing via manipulating the features available for decision making to represent real-world scenarios where the entirety of the data may not be available.

### 2.5. Metrics used for analysis

DOI: https://doi.org/10.34010/injiiscom.v7i1.14803
p-ISSN 2810-0670 e-ISSN 2775-5584

To analyze the performance of the models during testing the following metrics were used. These values are derived from a confusion matrix which is made up of four values: true positive (TP) which is where the sample is ransomware and correctly identified as such, true negative (TN) which is where the sample is benign and correctly identified as such, false positive (FP) where the sample is incorrectly identified as ransomware, and false negative (FN) where a sample of ransomware is incorrectly identified as benign.

- Accuracy (ACC) is a measure of how often the model correctly identifies TP and TN compared to the number of total classifications. The closer to 1, the better the model performed.
- Recall (REC) is a measure of how often ransomware is correctly identified compared to the amount of ransomware in the dataset. The closer to one, the better the model detects ransomware.
- Precision (PREC) is a measure of how often a positive label is true, compared to false positives. The closer to one, the better the program is at avoiding false positives.
- True Negative Rate is a measure of how often a negative label is true, compared to the number of false negatives. The closer to one, the better the model is at correctly identifying benign samples.
- F-Score (F1) is a measurement of harmony between recall and precision, with a higher score meaning a model is less likely to misclassify a sample.
- Matthew's Correlation Coefficient (MCC) is a measurement of the total model and shows how reliable the predictions are. MCC works better with imbalanced data. The lower the number, the higher the chances are that guesses are random.

$$ACC = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

$$REC = \frac{TP}{TP+FN} \quad (2)$$

$$PREC = \frac{TP}{TP+FP} \quad (3)$$

$$TNR = \frac{TN}{TN+FN} \quad (4)$$

$$F1 = \frac{(2*\left(\frac{TP}{TP+FP}\right)*\left(\frac{TP}{TP+FN}\right)}{\left(\frac{TP}{TP+FP}\right)+\left(\frac{TP}{TP+FN}\right)} \quad (6)$$

$$MCC = \frac{TN*TP-FN*FP}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \quad (7)$$

## 2.6. Creating the Model

To create the models used for testing, Conda and PyTorch (Ansel, 2024) were used to generate a feedforward artificial neural network (ANN) with three layers. The model's predictive features were optimized using Adam, a popular stochastic optimizer for deep learning. This model was trained using five-fold cross-validation optimizing around reducing loss and high MCC performance.

DOI: https://doi.org/10.34010/injiiscom.v7i1.14803
p-ISSN 2810-0670 e-ISSN 2775-5584

## 3. RESULTS AND DISCUSSION

### 3.1. Procedure

After creating the model as outlined in the prior section, the model's training metrics were generated using a combination of "NumPy", "scikit-learn", "matplotlib", and "pandas" packages. The performance during training was high, with PREC and TNR returning values of 1.0, and other metrics returning values in the high .9 range. When graphed to show the Receiver Operating Characteristics (see figure 1), the model covers at least ninety percent of results with proper classification. While these metrics potentially show a powerful model, these high values suggest that the model may have overfit during training. This means that the model believes that it can accurately predict all values in the training dataset, however it could have external validity issues and could misidentify samples not included during training.

To test this, the testing set which contained a mixture of included and excluded ransomware family samples was used to test the performance of the model. When tested, the model achieved a value of 0.862 in accuracy, and a true negative rate of 0.913. When the ROC was graphed (see figure 2), it still showed that eighty percent of results were correctly identified. However, the F1 and MCC lowered to 0.666 and 0.58 respectively. This means that while the model could correctly predict the results, there was a moderate chance that the model could not determine the result with confidence and misidentified samples because of it. See figure 3 for a graphical comparison of the training and testing performance of the ANN.

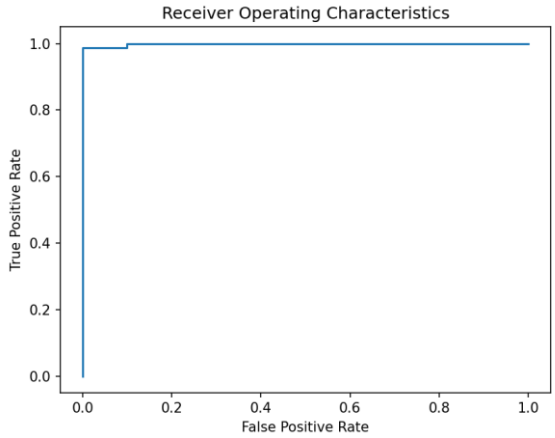


**Fig. 1. The ROC graph generated based on the ANN model's performance in training.**

DOI: https://doi.org/10.34010/injiiscom.v7i1.14803
p-ISSN 2810-0670 e-ISSN 2775-5584

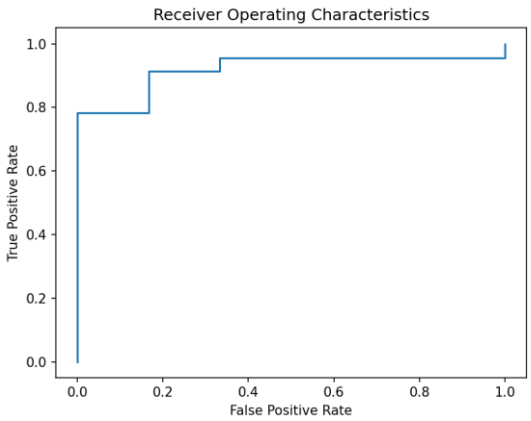


**Fig. 2. The ROC graph generated based on the ANN model's performance in testing.**

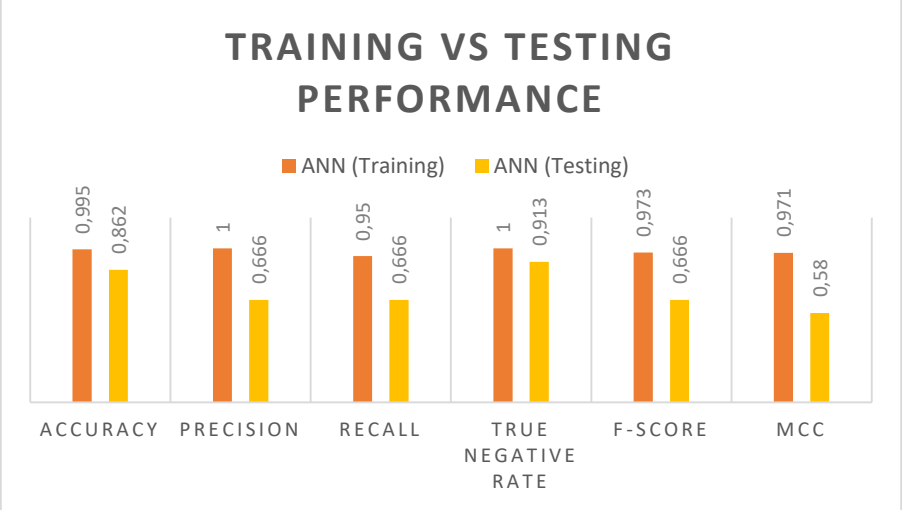


**Fig. 3. Bar graph of the ANN model's metrics from training and testing.**

DOI: https://doi.org/10.34010/injiiscom.v7i1.14803
p-ISSN 2810-0670 e-ISSN 2775-5584

| Model Tested | Accuracy | Precision | Recall | True Negative Rate | F-Score | MCC |
|---|---|---|---|---|---|---|
| ANN (Training) | 0.995 | 1.0 | .95 | 1.0 | 0.973 | 0.971 |
| ANN (Testing) | .862 | .666 | 0.666 | 0.913 | 0.666 | 0.580 |
| DNN by Nurnoby and El-Alfy (2019) | .972 | .980 | .952 | N/A | .966 | N/A |
| ANN by Sharma, Krishna and Kumar (2020) | 0.992 | 0.997 | 0.989 | N/A | 0.993 | N/A |
| RF by Masum et al. (2022) | 0.99 | 0.99 | 0.97 | N/A | 0.97 | N/A |

**Table 1. Testing Results vs Other Reported Results**

### 3.2. Analysis of Results

The results gathered from the testing shows some major limitations that need to be fixed to increase performance. First, the training dataset is less effective when it comes to identifying ransowmare not represented in training, this means that the features used in prediction were inefficient when it comes to external validity. To fix this, the features used in training and testing should be reduced or re-factored to increase the models resiliance when predicitng new ransomware. Alongside this, current predictions are being done using a bag of words method, other methods such as term frequency inverse frequency detection (TF-IDF) could offer benefits to the performance and resiliancy of the model used.

The model's performance in training had similar results to other literature in the field, which often use similar methodology to test their models.

Nurnoby and El-Alfy (2019) created a deep neural network similar to the one constructed in this experiment, however they also used other features for decision-making such as the files deleted. They reported similar to this articles training metrics, however their ransomware was older and the perforamnce of their work was not tested on something new. Sharma, Krishna, and Kumar (2020) did something similar but tested against

DOI: https://doi.org/10.34010/injiiscom.v7i1.14803
p-ISSN 2810-0670 e-ISSN 2775-5584

specifically android operating systems. Their dataset had 1045 features for prediction and preformed comparably to this papers ANN model during training. Masum et al. (2022) tested a less complex ensemble classifier, specifically random forest, and reported almost ones in their metrics.

This could mean that other works could suffer a similar decrease in efficacy when testing against samples not included in training. While the other models differing features may offer some explanation for the difference in performance, this could be caused by insufficient datasets that are designed for internal validity rather than representing real world scenarios as explained by Oz and Aris (2022) in their review.

## 4. Future Work

For future work into this detecting ransomware with AI, a major focus should be on ensuring external validity with the models created. Currently many studies report high performance values during their training and testing phase, however the datasets used are often using samples from similar times and families which can lead to high internal performance while having sensitivity to data not represented in the training set. Furthermore, there should be a focus on ensuring that all important metrics are included in the reported results. For example, TNR is important to understand how often a model can correctly identify a sample as much as precision.

Beyond this model drift is a major topic of concern as cited by Davies et al. (2021) and Razaulla et al (2023), as this limits the effectiveness of these models in protecting against zero-day attacks or novel ransomware with little to no samples. Designing models without accounting for this will lead to rapidly decaying effectiveness as time goes on, while this drift cannot be stopped entirely the models can be designed in a way to reduce the impact on effectiveness from the change's ransomware methods.

Another avenue of future research is in modifying the deep learning classifier to take advantage of TF-IDF or using meta heuristics to optimize the features of the dataset to increase performance. TF-IDF is unlikely to stop model drift as it in of itself is only one aspect of prediction. As stated by Oz and Aris (2022), current models lack comprehensiveness when it comes to prediction and can overly focus on a single predictive aspect. To create a model that is less prone to drift, a comprehensive list of features is important to prevent over specificity.

## 5. CONCLUSION

Ransomware is a major threat that all organizations and scholars wish to defeat. Artificial intelligence is just one of many layers of defence which have been developed as a tool against ransomware. Through the creation of a deep learning model and testing it against samples not represented in the training phase, an estimation of model drift is given. This model drift reduces the ACC of the model by ten percent, and the MCC almost halves. This shows that the neural network commonly outlined in similar research has flaws when predicting samples not represented in training. To fix this issue, the datasets used must comprehensively cover the many different aspects used by ransomware for predictive capabilities to remain resistant to change. While this cannot stop model drift, it can strengthen the performance over time. Alongside this, other deep

DOI: https://doi.org/10.34010/injiiscom.v7i1.14803
p-ISSN 2810-0670 e-ISSN 2775-5584

learning techniques such as recurrent neural networks, or bi-directional models could offer benefit comparatively. More data should also be added to the dataset to help balance the samples as it may be influencing the predictive qualities of the ANN.

## ACKNOWLEDGMENTS

Thanks to GameAbove and the College of Engineering and Technology at Eastern Michigan University for the support given with this study.

## REFERENCES

Almousa, M., Osawere, J., & Anwar, M. (2021). Identification of Ransomware families by Analyzing Network Traffic Using Machine Learning Techniques. In *2021 Third International Conference on Transdisciplinary AI (TransAI)* (pp. 19–24). IEEE. https://doi.org/10.1109/transai51903.2021.00012

Alzahrani, A., Alshahrani, H., Alshehri, A., & Fu, H. (2019). An Intelligent Behavior-Based Ransomware Detection System For Android Platform. In *2019 First IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*. IEEE. https://doi.org/10.1109/tps-isa48467.2019.00013

Anand, P. M., Charan, P. S., & Shukla, S. K. (2022). A Comprehensive API Call Analysis for Detecting Windows-Based Ransomware. In *2022 IEEE International Conference on Cyber Security and Resilience (CSR) Workshops* (pp. 337–344). IEEE. https://doi.org/10.1109/csr54599.2022.9850320

Ansel, J., Yang, E., He, H., Gimelshein, N., Jain, A., Voznesensky, M., Bao, B., Bell, P., Berard, D., Burovski, E., Chauhan, G., Chourdia, A., Constable, W., Desmaison, A., DeVito, Z., Ellison, E., Feng, W., Gong, J., Gschwind, M., . . . Chintala, S. (2024). *PyTorch 2: Faster machine learning through dynamic Python bytecode transformation and graph compilation* (Vol. 5, pp. 929–947). https://doi.org/10.1145/3620665.3640366

Davies, S. R., Macfarlane, R., & Buchanan, W. J. (2021). Review of current ransomware detection techniques. *2021 International Conference on Engineering and Emerging Technologies (ICEET)*, 1–6. https://doi.org/10.1109/iceet53442.2021.9659643

Ghazi, M. R., & Raghava, N. S. (2022). *Detecting Ransomware Attacks in Cloud Environment Using Machine Learning-Based Intelligence System in COVID-19 Chaos*. 2022 IEEE Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI). https://doi.org/10.1109/iatmsi56455.2022.10119441

Hirano, M., & Kobayashi, R. (2019). *Machine Learning Based Ransomware Detection Using Storage Access Patterns Obtained From Live-forensic Hypervisor* (pp. 1–6). 2019

Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS). https://doi.org/10.1109/iotsms48152.2019.8939214

Hossain, M. S., Hasan, N., Samad, M. A., Shakhawat, H. M., Karmoker, J., Ahmed, F., Fuad, K. F. M. N., & Choi, K. (2022). Android ransomware detection from traffic analysis using metaheuristic feature selection. *IEEE Access*, *10*, 128754–128763. https://doi.org/10.1109/access.2022.3227579

Hsu, C., Yang, C., Cheng, H., Setiasabda, P. E., & Leu, J. (2021). Enhancing file entropy analysis to improve machine learning detection rate of ransomware. *IEEE Access*, *9*, 138345–138351. https://doi.org/10.1109/access.2021.3114148

Iqbal, M. J., Aurangzeb, S., Aleem, M., Srivastava, G., & Lin, J. C. (2022). RTHREatDroid: A ransomware detection approach to secure IoT based healthcare systems. *IEEE Transactions on Network Science and Engineering*, *10*(5), 2574–2583. https://doi.org/10.1109/tnse.2022.3188597

Masum, M., Faruk, M. J. H., Shahriar, H., Qian, K., Lo, D., & Adnan, M. I. (2022). Ransomware classification and detection with machine learning algorithms. *2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)*, 0316–0322. https://doi.org/10.1109/ccwc54503.2022.9720869

Melaragno, A., & Casey, W. (2022). Change Point Detection with Machine Learning for Rapid Ransomware Detection. *2021 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, 1–9. https://doi.org/10.1109/dasc/picom/cbdcom/cy55231.2022.9927828

Moussaileb, R., Cuppens, N., Lanet, J., & Bouder, H. L. (2021). A survey on Windows-based ransomware taxonomy and detection Mechanisms. *ACM Computing Surveys*, *54*(6), 1–36. https://doi.org/10.1145/3453153

Nurnoby, M. F., & El-Alfy, E. M. (2019). *Overview and Case Study for Ransomware Classification Using Deep Neural Network*. 2019 2nd IEEE Middle East and North Africa COMMunications Conference (MENACOMM). https://doi.org/10.1109/menacomm46666.2019.8988551

Razaulla, S., Fachkha, C., Markarian, C., Gawanmeh, A., Mansoor, W., Fung, B. C. M., & Assi, C. (2023). The Age of Ransomware: A survey on the evolution, taxonomy, and research directions. *IEEE Access*, *11*, 40698–40723. https://doi.org/10.1109/access.2023.3268535

DOI: https://doi.org/10.34010/injiiscom.v7i1.14803
p-ISSN 2810-0670 e-ISSN 2775-5584

O'Reilly, K., & Brukhovetskyy, A. CAPE: Malware Configuration And Payload Extraction (Version 2) [Computer software]. https://github.com/kevoreilly/CAPEv2

Oz, H., Aris, A., Levi, A., & Uluagac, A. S. (2022). A survey on Ransomware: Evolution, taxonomy, and defense solutions. *ACM Computing Surveys*, *54*(11s), 1–37. https://doi.org/10.1145/3514229

Sharma, S., Krishna, C. R., & Kumar, R. (2020). *Android Ransomware Detection using Machine Learning Techniques: A Comparative Analysis on GPU and CPU* (pp. 1–6). 2020 21st International Arab Conference on Information Technology (ACIT). https://doi.org/10.1109/acit50332.2020.9300108

Sharma, N., & Sangal, A. (2023). Machine Learning Approaches for Analysing Static features in Android Malware Detection. In *2023 Third International Conference on Secure Cyber Computing and Communication (ICSCCC)* (pp. 93–96). IEEE. https://doi.org/10.1109/icsccc58608.2023.10176445

Sukul, M., Lakshmanan, S. A., & Gowtham, R. (2022). Automated Dynamic Detection of Ransomware using Augmented Bootstrapping. *2022 6th International Conference on Trends in Electronics and Informatics (ICOEI)*, 787–794. https://doi.org/10.1109/icoei53556.2022.9777099

DOI: https://doi.org/10.34010/injiiscom.v7i1.14803
p-ISSN 2810-0670 e-ISSN 2775-5584