iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** V **Month of publication:** May 2024

DOI: https://doi.org/10.22214/ijraset.2024.62387

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue V May 2024- Available at www.ijraset.com

Adware Detection using LSTM and Comparison with other ML Models

Mr. Himanshu Kishor Khare[1], Mr. Malhar Aniruddha Joshi[2], Dr. Sonali Antad[3]
Computer Science Engineering, Vishwakarma Institute of Technology, Pune, India

Abstract: ***In recent years, the internet has become the predominant communication medium, paralleled by a surge in cyber threats where attackers actively exploit vulnerabilities to obtain sensitive information. Among the various malware tactics employed, Adware poses a significant challenge. Addressing this concern, cyber security specialists leverage Machine Learning (ML) techniques, with this paper proposing a novel Long Short-Term Memory (LSTM) algorithm for adware detection. The research employs a holistic methodology involving diverse data pre-processing techniques, feature selection, and ML algorithms to effectively identify adware samples within the dataset. A comparative analysis of ML classifiers, including Random Forest (RF), k - Nearest Neighbors (k-NN), Decision Tree (DT), and Logistic Regression (LR), reveals optimal detection accuracies of 98.66%, 98.10%, and 98.05% for DT and k-NN, respectively. These findings underscore the efficacy of the proposed approach in fortifying cyber security against adware threats.***

Keywords: Adware, Malware, Machine learning, ML algorithms

I. INTRODUCTION

The ubiquity of the internet, serving as the cornerstone of contemporary technology, has given rise to a surge in cyber threats. In response, malevolent actors are crafting malicious applications to exploit individuals, extracting financial resources, compromising data security, and wreaking havoc on the software that powers smartphones, wearables, IoT devices, and sophisticated transportation services. This escalating threat landscape is exacerbated by the widespread use of apps perpetually connected to the internet, leading to a substantial increase in network traffic. Projections indicate that fifth-generation (5G) connectivity, anticipated to surpass 4G by 2023, will result in nearly triple the volume of data traffic, mirroring a parallel growth in the proliferation of malicious software.

Malware is a form of disruptive software, which is intentionally crafted to interfere, cause harm, or take over unauthorized access to applications, smart devices, systems, or networks [5, 6]. Categorized based on purposes and proliferation systems, malware encompasses a range of types including Ransomware, Adware, Trojan, Bot, Launcher, Rootkit, Downloader, Spyware, Backdoor, Worm, and Virus [7]. Manual analysis of each Android malware is impractical, necessitating the adoption of Machine Learning (ML) techniques for efficient detection and analysis. ML proves invaluable in monitoring network traffic, leveraging a multitude of features for the effective identification and understanding of malware. Adware, a distinct form of malware commonly known as "advertising software," functions by automatically displaying or downloading advertising content, frequently undesirable when a user is online [8]. Its primary objective is to generate revenue for developers through user interactions, such as clicks on displayed advertisements. Some variants of adware go beyond intrusive advertising, collecting user browsing information without consent to tailor ads more effectively. Adware developers receive compensation based on metrics like cost-per-click (CPC), cost-per-action (CPA), impressions, downloads, and installs generated by the application. Cyber threats exploit adware on dual fronts: firstly, targeting advertisers to pilfer budgets and exert unauthorized control, and secondly, compromising users' phones, personal information, or monetary gains [9].

II. REALTED WORK

In 2017, H. Lashkari and their team utilized dynamic analysis to spot malicious applications. Their model showcased an average precision of 91.41% by employing various methodologies such as Decision Trees (DT), Regression (R), k-Nearest Neighbors (k-NN), Random Forest (RF), and Random Trees (RT). The assessment covered 1900 benign applications and scrutinized 12 different strains of general malware and adware, utilizing a feature set consisting of 9 network traffic attributes [10].

In 2018, R. Goswami and their collaborators introduced a prevalent static approach for detecting malevolent entities based on permission characteristics. This technique involved employing machine learning classifiers and centered on scrutinizing the androidManifest.xml file. The researchers carried out their experiments using two separate datasets. A unique feature selection strategy, integrating Levenshtein Distance and Cosine Similarity, was utilized to gauge the likeness between permission vectors. The community-based feature reduction approach achieved a remarkable accuracy detection rate of 98.20% [2].

In 2019, H. Lashkari and their team leveraged the latter portion of the CICAndMal2017 dataset, which encompassed intents and permissions as static attributes, alongside API calls as dynamic characteristics. Their study comprised a two-layer examination of these attributes. Initially, a binary classification model was formulated to differentiate between malware and benign applications based on static features. Following that, in the second layer, a malware family classification model was developed to categorize malware into their respective families using dynamic attributes. The system exhibited notable success, achieving a precision of 95.3% in static malware analysis during the first layer and 59.7% precision in the dynamic classification of malware families during the second layer [13].

In 2020, A. Sangal and their collaborators employed machine learning algorithms for the detection of a novel Android malware [14]. Utilizing ML classifiers such as Random Forest (RF), k-Nearest Neighbors (k-NN), Support Vector Machine (SVM), and Naive Bayes (NB), the researchers conducted analyses on a dataset comprising Android applications with permissions and intent feature sets. Data pre-processing techniques were implemented to manage null values, and feature selection methodologies were applied to decrease the dimensions of the CICAndMal2019 dataset. The most notable detection outcome was achieved through the Random Forest classifier, resulting in an impressive 96% accuracy in malware detection.

III. ANALYSIS OF DATASET

The dataset utilized in the research titled "Malicious Adware Detection in Android using DL" originates from Kaggle and was shared by Mr.Saeed Seraj. It encompasses diverse samples associated with adware, a form of malicious software notorious for displaying unwanted advertisements. The dataset comprises a range of data types including network traffic, API/SYS calls, logs, memory dumps, and phone statistics, providing a comprehensive view of adware behaviour. With coverage of over 40 malware families, the dataset offers a broad perspective on the landscape of adware variants. In total, the dataset contains 441 columns, with one dedicated to ID and 440 representing integer data. This dataset serves as a valuable resource for researchers aiming to develop and enhance deep learning models for the detection and mitigation of adware on Android devices.

Table I. Number of adware samples for each family

Adware family	AV labelled	Num. of samples
Dowgin	Gdata	42599
Ewind	Koodous	43374
Feiwo	Fortinet	56632
Gooligan	Fortinet	93772
Kemoge	Lookout	38771
Koodous	Koodous	32547
Mobidash	Enet32	31034
Selfmite	AntiVirus	13029
Shuanet	Lookout	39271
Youmi	Gdata	36035

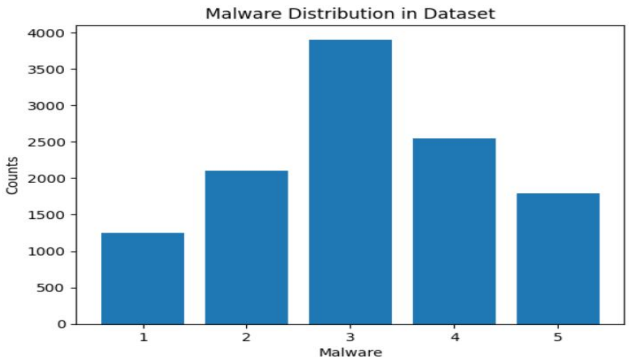


Diagram I. malware distribution in dataset

IV. PROPOSED MODEL

The proposed model consists of five phases, which are:

1) Gathering and analyzing features in Malicious Adware Detection in Android using DL dataset.
2) The second phase involves data preprocessing, which includes removing all columns with zero-value data or columns with low variance. Additionally, data normalization is performed using Quantile scaling.
3) The third phase involved utilizing feature selection techniques to identify and choose the most optimal features.
4) In the fourth phase (training phase), ML classifiers are trained on the 80% training dataset using the best features identified in the previous phase.
5) The final phase (Testing and Evaluation) involves testing and evaluating ML classifiers on the 20% testing dataset using ML performance metrics.

V. EXPERIMENTS

To assess the efficacy of the proposed Long Short-Term Memory (LSTM) algorithm for adware detection, several experiments were conducted. These experiments encompassed data preprocessing, feature selection, utilization of machine learning classifiers, and performance evaluation using various metrics.

A. Data preprocessing

Upon analysis of the dataset, which initially comprised 85 features, it was observed that certain features contained zero values, while others exhibited significant variations in their value ranges. Thus, data preprocessing operations were imperative to optimize the dataset for effective utilization with machine learning algorithms.

1) *Removing Features with Zero or Low Variance:* The VarianceThreshold technique [21] was employed to eliminate low-variance features that do not significantly contribute to model performance. After this process, 12 low-variance features and four string-based columns were removed, resulting in a refined dataset containing 69 features.
2) *Feature Scalling:* Feature selection is a crucial step aimed at identifying a subset of input variables that significantly impact the target variable, leading to improved model performance and generalization. Various feature selection techniques were applied to different feature sets, each method selecting features based on specific criteria.

B. Data Preprocessing

Feature selection is a pivotal step directed towards identifying and isolating a subset of input variables that wield substantial influence on the target variable. Its advantages encompass trimming classification time, curbing over fitting tendencies, and bolstering accuracy rates. This becomes especially imperative when dealing with expansive datasets characterized by high dimensionality, wherein the exclusion of extraneous features can significantly refine model performance and enhance generalization capabilities. In this study, a spectrum of feature selection methodologies was deployed across diverse feature sets, each technique tailored to select features based on distinct criteria pertinent to the research objectives.

1) Univariate feature selection

Univariate feature selection employs statistical tests to identify features that exhibit the strongest relationship with the output variable. This method operates by evaluating each feature individually, comparing its relevance to the desired target variable, and disregarding irrelevant features. Each feature is assigned a weight based on its significance, and these weights are subsequently compared. In this study, the f-test or f-statistic method [24] was utilized to identify features with the highest scores. The f-test is commonly employed for selecting features in numerical input data when the target variable is categorical. The SelectKBest() function [25] was then employed to select a specific number of top features. Specifically, this function was applied to select the best subset of features, ranging from 10 to 25, from the initial set of 69 features.

2) Select from Model

The researchers employed the SelectFromModel technique, which served as a robust transformer method compatible with estimators featuring the 'feature_importance' attribute [26]. This method operated by identifying crucial features based on their weights. Unlike univariate feature selection, SelectFromModel evaluated all features simultaneously, enabling it to capture interactions effectively. Initially, three models—Random Forest (RF), Support Vector Machine (SVM), and k-Nearest Neighbors (k-NN)—were utilized to train the data and select features. Subsequently, the SelectFromModel technique was applied to these models, resulting in the selection of 20, 17, and 25 features by RF, SVM, and k-NN, respectively, from the original set of features.

C. Machine Learning Classifiers

Four common classifiers frequently employed in the cybersecurity domain were utilized for this study, as referenced in [27, 28, 29]. These classifiers play a crucial role in analyzing and identifying potential security threats within various systems and networks

1) Logistic Regression (LR)

Logistic Regression is a statistical technique utilized for binary classification tasks, estimating the probability of a sample falling into one of two categories. By modeling this relationship with a logistic function, it constrains output between 0 and 1. Widely applied in cybersecurity, it's effective for tasks like malware identification and fraud detection, leveraging its simplicity, interpretability, and efficiency. Through optimization algorithms, it learns optimal weights for input features, facilitating predictions. Despite its name, it's employed for classification rather than regression problems, making it a cornerstone algorithm in various fields due to its robustness and ease of implementation.

2) SVM

Support Vector Machine (SVM) is a powerful supervised machine learning algorithm used for classification and regression tasks. It works by finding the optimal hyperplane that best separates data points belonging to different classes in a high-dimensional space. SVM is effective in handling both linearly separable and non-linearly separable data through the use of kernel functions, making it versatile for various applications. In cybersecurity, SVM is employed for tasks such as intrusion detection and malware classification due to its ability to handle complex datasets and achieve high classification accuracy.

3) Random Forest (RF)

Random Forest is an ensemble learning algorithm used for both classification and regression tasks. It operates by constructing a multitude of decision trees during training and outputs the mode or mean prediction of the individual trees for classification and regression, respectively. Random Forest mitigates over fitting and improves accuracy by combining predictions from multiple trees. Widely employed in cyber security, it's effective for tasks like malware detection and network intrusion detection due to its robustness and ability to handle high-dimensional data with complex relationships.

4) k-Nearest Neighbor (k-NN)

k-Nearest Neighbor (k-NN) is a simple yet effective supervised machine learning algorithm used for classification and regression tasks. It operates by finding the k-nearest data points to a given query point and determining the majority class or average value among them for classification or regression, respectively. k-NN is non-parametric and instance-based, meaning it does not make strong assumptions about the underlying data distribution. In cyber security, k-NN is utilized for tasks such as anomaly detection and malware classification, thanks to its simplicity and adaptability to various types of data.

D. Deep Learning Model: Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) networks were first proposed by Hochreiter & Schmidhuber in 1997 as a solution to the vanishing gradient problem encountered with traditional recurrent neural networks (RNNs). LSTMs are designed to address the challenge of capturing long-range dependencies within data sequences of varying lengths. This capability is facilitated by a unique structure known as a memory cell, which features self-connected units that help maintain information over extended periods.

The design of an LSTM cell is characterized by three key components: the input gate, the forget gate, and the output gate. These gates are crucial for regulating the information that enters and exits the memory cell, thereby allowing the LSTM to dynamically adjust its internal state according to the incoming data and its contextual relevance.

- *Input Gate:* Decides the amount of new information to be added to the memory cell.
- *Forget Gate:* Determines what portion of the previous information remains or is discarded from the cell state.
- *Output Gate:* Controls the extent of the memory cell's current state that should be passed on to the next layer of the network.

By regulating the flow of information through these gates, LSTMs can effectively remember and learn long-term dependencies in sequential data, making them particularly well-suited for tasks such as natural language processing, time series prediction, and, relevant to cyber security, anomaly detection and malware classification.

In cyber security applications, LSTMs are used to analyze sequences of network traffic, system logs, or other temporal data to detect patterns indicative of malicious activity. They can learn complex temporal relationships and detect deviations from normal behaviour, helping security professionals identify and respond to threats more effectively.

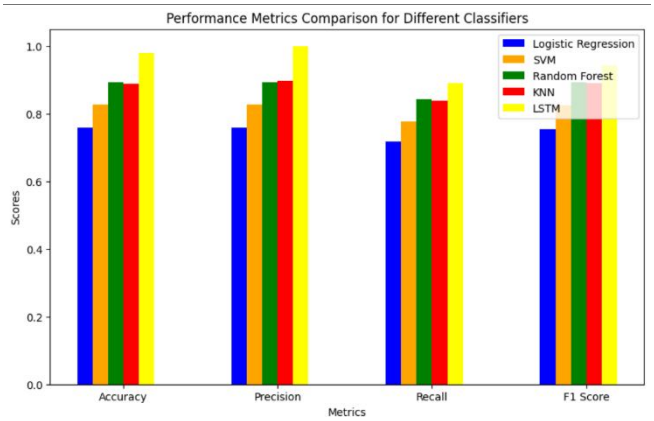


Diagram II. performance metrics comparison for different classifiers

E. Performance Matrix

In assessing machine learning classifiers, different performance metrics derived from the confusion matrix (CM) are used. Each row of the CM corresponds to instances in a true class, and each column indicates instances in a predicted class. A detailed description of the CM is provided in Table II.

TABLE II. Confusiom Matrix

Parameter		Prediction	
		Adware	*Benign*
Actual	*Adware*	True Positive (TP)	False Negative (FN)
	Benign	False Positive (FP)	True Negative (TN)

Where:

- TP: The count of malware samples that are correctly classified.
- TN: The count of benign samples that are correctly classified.
- FP: The count of benign samples that are incorrectly classified.
- FN: The count of malware samples that are incorrectly classified.

The research uses seven ways to check how well the Machine Learning works:

1) *Accuracy (Acc):* The percentage of correctly classified instances among all classifications, calculated by dividing the number of correctly classified instances by the total number of instances.

$$Acc = (TP+TN) / (TP+TN+FP+FN) \quad (1)$$

2) *Precision (P):* the proportion of true positive results among all positive predictions.

$$P = TP / (TP+FP) \quad (2)$$

3) *True Positive Rate (TPR)/Recall (Re):* the proportion of malware samples correctly detected by the model.

$$TPR = TP/(TP+FN) \quad (3)$$

4) *True Negative Rate (TNR):* it is the proportion of correctly predicted negative samples out of all negative samples.

$$TNR = TN / (TN+FP) \quad (4)$$

5) *False Positive Rate (FPR):* the proportion of benign samples incorrectly classified as malicious.

$$FPR = FP / (FP+TN) \quad (5)$$

6) *Classification Time (T):* The duration required to train the algorithm using the dataset.

7) *F -score (F):* a metric that combines precision and recall into a single value to evaluate the performance of the model.

$$F = 2\times (P\times Re) / (P+Re) \quad (6)$$

VI.CONCLUSION AND FUTURE WORK

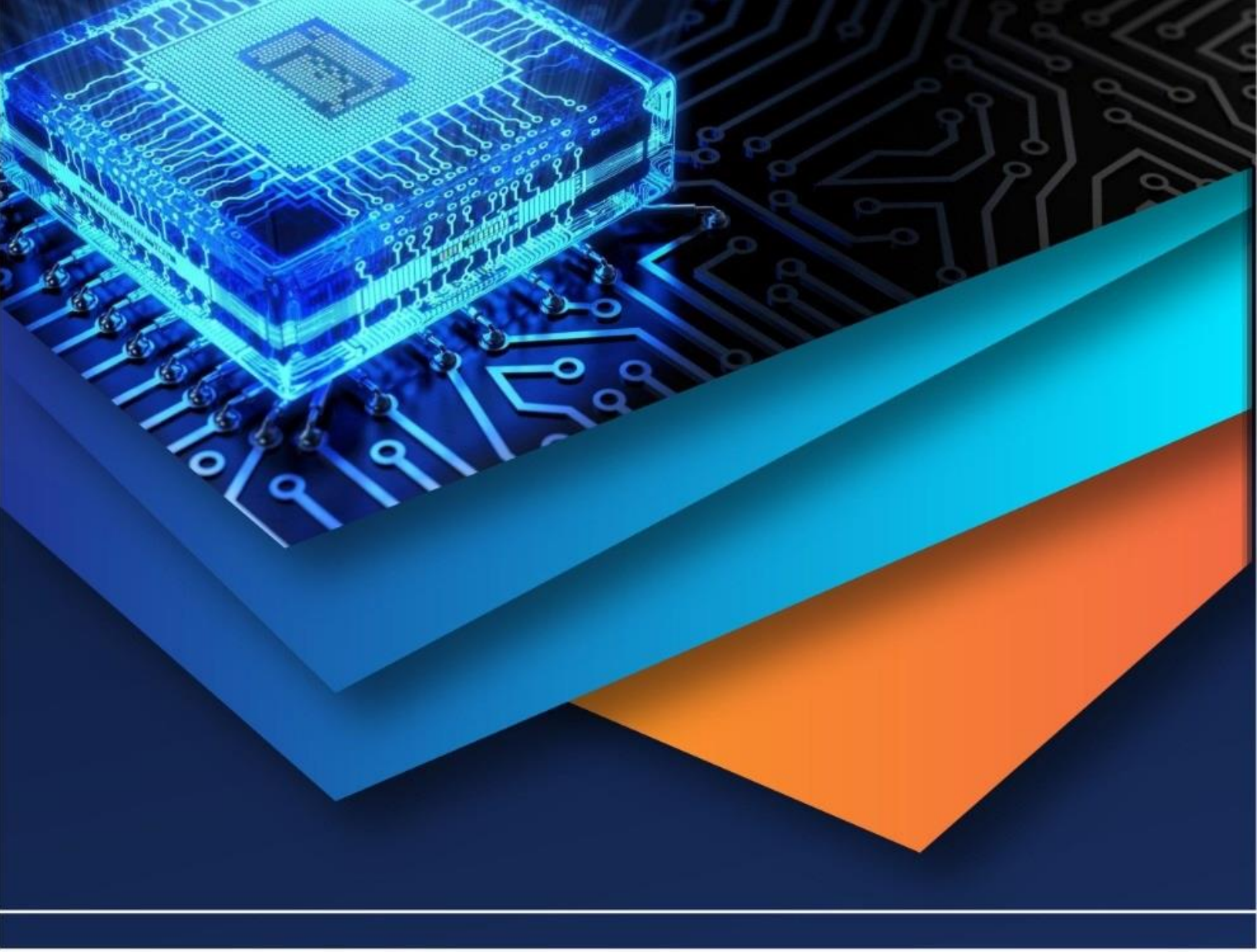
In the realm of cybersecurity, certain attackers possess the intricate ability to discern patterns within both malicious and benign applications, often employing sophisticated behavior-based tactics like mimicking network flow durations. Despite their efforts, their capacity to manipulate the variance of numerous features remains limited, posing challenges in evading detection models, especially those integrated into the proposed research framework. This study distinguishes itself from existing works by focusing on the comprehensive selection of the most effective features from the entire feature set, rather than concentrating solely on specific network traffic categories. Experimental findings underscore the effectiveness of network traffic features for adware detection. Through a rigorous approach involving dataset preprocessing and feature selection techniques, the proposed model optimizes the feature set for enhanced detection accuracy. Subsequent evaluation utilizing machine learning classifiers yields promising results, typically achieving a detection accuracy of 98% and a low average false positive rate of 0.006%. Future endeavors aim to extend this approach to encompass other types of Android malware and explore the integration of additional feature types for a more robust detection framework.

VII. ACKNOWLEDGMENT

This study was conducted at Vishwakarma Institute of Technology. The authors extend their gratitude to all individuals who contributed to the completion of this research.

REFERENCES

[1] I. Almomani, A. AlKhayer, and M. Ahmed, “An Efficient Machine Learning-based Approach for Android v. 11 Ransomware Detection,” 2021 1st International Conference on Artificial Intelligence and Data Analytics (CAIDA). IEEE, 2021.
[2] A. Bhattacharya, and R. T. Goswami, “Community based feature selection method for detection of android malware,” Journal of Global Information Management (JGIM) 26.3 (2018): 54-77.
[3] P. Thorat, N. K. Dubey, K. Khetan, and R. Challa, “SDN-based Predictive Alarm Manager for Security Attacks Detection at the IoT Gateways,” 2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC). IEEE, 2021.
[4] The Cisco Annual Internet Report (2018–2023) White Paper. https://www.cisco.com/c/en/us/solutions/collateral/executive- perspectives/annual-internet-report/white-paper-c11-741490.html
[5] M. W. Azeem, M. S. Sarfraz, and U. Shoaib, “Comparison of Different Techniques for Detecting Malware in Smartphones,” International Journal of Computer Science and Information Security 14.5 (2016): 642.
[6] I. Martín, J. A. Hernández, and S. L. Santos, “Machine-Learning based analysis and classification of Android malware signatures,” Future Generation Computer Systems 97 (2019): 295-305.
[7] F. Noorbehbahani, F. Rasouli, and M. Saberi, “Analysis of machine learning techniques for ransomware detection,” 2019 16th International ISC (Iranian Society of Cryptology) Conference on Information Security and Cryptology (ISCISC). IEEE, 2019.
[8] J. Gao, L. Li, P. Kong, T. F. Bissyande, and J. Klein, “Should you consider adware as malware in your study?,” 2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER). IEEE, 2019.
[9] A. Javaid, I. Rashed, H. Abbas, and M. Fugini, “Ease or privacy? a comprehensive analysis of Android embedded adware,” 2018 IEEE 27th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE). IEEE, 2018.
[10] A. H. Lashkari, A. F. Kadir, H. Gonzalez, K. F. Mbah, and A. A. Ghorbani, “Towards a network-based framework for android malware detection and characterization,” 2017 15th Annual conference on privacy, security and trust (PST). IEEE, 2017.
[11] A. H. Lashkari, A. F. Kadir, L. Taheri, and A. A. Ghorbani, “Toward developing a systematic approach to generate benchmark android malware datasets and classification,” 2018 ICCST. IEEE, 2018.
[12] S. Fallah, and A. J. Bidgoly, “Benchmarking machine learning algorithms for android malware detection,” Jordanian Journal of Computers and Information Technology (JJCIT) 5.03 (2019).
[13] L. Taheri, A. F. A. Kadir, and A. H. Lashkari, “Extensible android malware detection and family classification using network-flows and API-calls,” 2019 ICCST. IEEE, 2019.
[14] A. Sangal, and H. K. Verma, “A static feature selection-based android malware detection using machine learning techniques,” 2020 (ICOSEC). IEEE, 2020.
[15] A. Mahindru, A. L. Sangal, “Dldroid: Feature selection based malware detection framework for android apps developed during covid-19,” Int J Emerg Technol 11.3: 516-525.
[16] S. J. Lee, et al, “Study on Systematic Ransomware Detection Techniques,” 2021 23rd International Conference on Advanced Communication Technology (ICACT), IEEE, 2021.
[17] The CICAndMal2017 dataset, produced by Canadian Institute for Cybersecurity in 2017.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY