International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

ENHANCING ANDROID MALWARE DETECTION THROUGH HYBRID FEATURE FUSION, DEEP LEARNING, AND EXPLAINABLE AI (XAI)

PULI PAVAN KUMAR REDDY
[1]B. Tech Student, Department of Computer Science and Engineering and Business Systems, Rajeev Gandhi Memorial College of Engineering & Technology, Nandyal, Andhra Pradesh
pavankreddy9542@gmail.com

KANIKI SIDDARTHA
[1]B. Tech Student, Department of Computer Science and Engineering and Business Systems, Rajeev Gandhi Memorial College of Engineering & Technology, Nandyal, Andhra Pradesh
siddarthakaniki@gmail.com

MARRIBOINA NARENDRA
[1]B. Tech Student, Department of Computer Science and Engineering and Business Systems, Rajeev Gandhi Memorial College of Engineering & Technology, Nandyal, Andhra Pradesh
narendrayadav93900@gmail.com

Dr.GANJIKUNTA CHAKRAPANI
[4].Assistant Professor, Department of Computer Science and Engineering and Business Systems, Rajeev Gandhi Memorial College of Engineering & Technology, Nandyal, Andhra Pradesh, Nandyal, Andhra Pradesh
chakrapaniaiml@rgmcet.edu.in

ABSTRACT

The rapid proliferation of Android applications has led to a significant rise in sophisticated malware, posing serious security challenges. Traditional detection techniques, primarily based on static or dynamic analysis alone, often fail to effectively identify complex and obfuscated threats. This paper proposes a hybrid Android malware detection framework that integrates static and dynamic analysis to improve detection accuracy and reliability. In the static analysis phase, Android application packages (APK files) are decompiled using APKTool to extract features such as permissions, API calls, intents, activities, opcode sequences, and control flow graphs (CFGs). These features enable the identification of suspicious patterns without executing the application. In the dynamic analysis phase, applications are executed in a controlled Android emulator environment, where runtime behaviors are monitored using tools such as Frida, strace, and Monkey testing. This facilitates the capture of real-time behaviors including system calls, API interactions, and network activities. The extracted features from both phases are fused into a unified feature set and utilized for malware classification using deep learning techniques. Furthermore, Explainable Artificial Intelligence (XAI) methods are incorporated to enhance transparency and interpretability of the model's decisions. A Flask-based web application is developed to provide an interactive interface for uploading APK files and visualizing analysis results. Experimental evaluation demonstrates that the proposed hybrid approach significantly improves detection performance and effectively identifies advanced malware.

Index Terms— Android malware, static analysis, dynamic analysis, deep learning, feature fusion, explainable AI (XAI), APK analysis.

1. INTRODUCTION

Android has emerged as the most widely used mobile operating system, making it a prime target for malware attacks. The increasing number of malicious applications has raised serious concerns regarding user privacy, data security, and system integrity. Traditional malware detection techniques, including signature-based and heuristic-based methods, are no longer sufficient to detect modern malware that employs obfuscation, code injection, and polymorphism.

Static analysis techniques examine application code without execution, making them fast and scalable. However, they often fail to detect runtime behaviors

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

and obfuscated code. On the other hand, dynamic analysis observes application behavior during execution but is resource-intensive and time-consuming. To address these limitations, this paper proposes a hybrid malware detection system that combines both static and dynamic analysis. The key contributions of this work include:

1. A hybrid feature extraction approach combining static and dynamic analysis
2. Integration of deep learning for improved classification accuracy
3. Use of Explainable AI (XAI) to interpret model decisions
4. Development of a web-based system for user interaction and analysis

1.1 Background of the Study

The rapid expansion of Android-based smartphones and applications has significantly transformed the digital ecosystem. Android, being an open-source platform, allows developers to create and distribute applications easily. However, this openness has also led to a surge in malicious applications (malware), posing serious threats to user privacy, data security, and system integrity. Traditional malware detection techniques rely mainly on **signature-based methods**, which are ineffective against new and evolving threats such as polymorphic and obfuscated malware. As attackers continuously develop advanced evasion techniques, there is a growing need for more robust and intelligent detection systems. Recent advancements in machine learning and deep learning have opened new avenues for malware detection by analyzing patterns and behaviors. However, relying solely on static or dynamic analysis has limitations. Therefore, integrating both approaches through hybrid feature fusion provides a more comprehensive and reliable solution.

1.2 Problem Statement

Existing Android malware detection systems face several challenges:

- Inability to detect **zero-day and obfuscated malware**
- Limitations of using only **static or dynamic analysis independently**
- Lack of **interpretability** in deep learning-based models
- High false positive and false negative rates

These limitations highlight the need for a hybrid approach that combines multiple analysis techniques along with explainability to improve trust and accuracy in malware detection systems.

1.3 Objectives of the Study

The main objectives of this study are:

- To develop a **hybrid malware detection framework** combining static and dynamic analysis
- To extract meaningful features such as permissions, API calls, and runtime behaviors
- To implement **feature fusion techniques** for improved classification
- To apply **deep learning models** for accurate malware detection
- To incorporate **Explainable AI (XAI)** for interpreting model decisions
- To design a **Flask-based web application** for real-time APK analysis

1.4 Scope of the Study

This study focuses on the detection of Android malware using a hybrid approach. The scope includes:

- Analysis of APK files using static and dynamic techniques
- Feature extraction and fusion for classification
- Application of deep learning algorithms for malware detection
- Integration of XAI methods to improve model transparency
- Development of a web-based interface for user interaction

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

However, the study is limited to Android platforms and may not directly apply to other operating systems such as iOS or Windows. Additionally, the performance of the model depends on the quality and diversity of the dataset used.

2. RELATED WORK

Several studies have explored Android malware detection using static, dynamic, and hybrid approaches. Static analysis-based methods extract features such as permissions and API calls but struggle with code obfuscation. Dynamic analysis techniques monitor runtime behavior but require significant computational resources. Hybrid approaches have shown improved performance by combining both methods. Recent research has also incorporated machine learning and deep learning models such as

Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) for malware classification. However, many existing systems lack interpretability, making it difficult to understand why a model classifies an application as malicious. This limitation highlights the need for integrating Explainable AI techniques into malware detection systems.

3. PROPOSED METHODOLOGY

The proposed system consists of five main stages: static analysis, dynamic analysis, feature fusion, deep learning-based classification, and explainable AI.

A. Static Analysis

In this phase, APK files are decompiled using APKTool to extract relevant features without executing the application. The extracted features include:

- Permissions
- API calls
- Intents and activities
- Opcode sequences
- Control Flow Graphs (CFGs)

These features help identify suspicious patterns in the application code.

B. Dynamic Analysis

Applications are executed in an Android emulator (AVD) to monitor runtime behavior. The following tools are used:

- Frida – for API hooking
- Strace – for system call tracing
- Monkey tool – for automated user interaction

This phase captures runtime features such as:

- System calls
- API usage patterns
- Network activity

C. Feature Fusion

The features extracted from static and dynamic analysis are combined into a unified feature vector. This hybrid feature set improves detection accuracy by capturing both structural and behavioral characteristics of applications.

D. Deep Learning Model

A deep learning model is used to classify applications as benign or malicious. The model processes the fused feature vector and learns complex patterns associated with malware behavior.

Common models that can be used include:

- Convolutional Neural Networks (CNN)
- Long Short-Term Memory (LSTM) networks
- Hybrid CNN-LSTM models

E. Explainable AI (XAI)

To improve transparency, XAI techniques such as SHAP (SHapley Additive exPlanations) or LIME (Local Interpretable Model-agnostic Explanations) are used. These techniques help explain the contribution of each feature in the classification decision, making the model more interpretable and trustworthy.

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

1. Feature Fusion Formula

Combining static and dynamic features:

$$F_{hybrid} = \alpha F_{static} + (1 - \alpha) F_{dynamic}$$

- F_{hybrid}: Final fused feature vector
- F_{static}: Static features (permissions, API calls, etc.)
- $F_{dynamic}$: Dynamic features (runtime behavior)
- α: Weight (0 to 1)

2. Neural Network Layer (Deep Learning)

Basic forward propagation:

$$a^{(l)} = \sigma(W^{(l)} a^{(l-1)} + b^{(l)})$$

- $a^{(l)}$: Activation of layer l
- $W^{(l)}$: Weight matrix
- $b^{(l)}$: Bias
- σ: Activation function (ReLU, Sigmoid)

3. Binary Classification (Malware vs Benign)

Using sigmoid function:

$$\hat{y} = \frac{1}{1+e^{-z}}$$

- $\hat{y}$: Predicted probability
- z: Output from neural network

4. Loss Function (Binary Cross-Entropy)

$$L = -\frac{1}{N} \sum_{i=1}^{N} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)]$$

- y_i: True label (0 or 1)
- $\hat{y}_i$: Predicted value
- N: Number of samples

9. SHAP (Explainable AI)

$$\phi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N|-|S|-1)!}{|N|!} [f(S \cup \{i\}) - f(S)]$$

- ϕ_i: Contribution of feature i
- Used for model explainability

4. SYSTEM ARCHITECTURE

The system architecture consists of multiple interconnected modules:

System Modules Description

1. APK Upload Module

The APK Upload Module serves as the entry point of the system, allowing users to submit Android application packages (APK files) for analysis. This module is implemented using a Flask-based web interface, where users can securely upload files through a browser. Upon upload, the system performs initial validation checks such as file format verification, size constraints, and integrity validation to ensure that only legitimate APK files are processed. The uploaded files are then stored temporarily for further analysis.

2. Static Analysis Module

The Static Analysis Module examines the APK file without executing it. The APK is first decompiled using tools such as APKTool to extract its internal components. Key features extracted include permissions, API calls, intents, activities, services, opcode sequences, and control flow graphs (CFGs). This module identifies suspicious patterns and potential vulnerabilities by analyzing the application's code structure and manifest file. Static analysis is efficient and fast but may not detect runtime behaviors or obfuscated malware effectively.

3. Dynamic Analysis Module

The Dynamic Analysis Module evaluates the behavior of the application during execution in a controlled environment such as an Android emulator or sandbox. Tools like Frida, strace, and Monkey testing are used to monitor runtime activities. This module captures system calls, API interactions, file operations, memory usage, and network traffic. Dynamic analysis helps in detecting hidden malicious behaviors that are not visible during static analysis, especially in the case of obfuscated or polymorphic malware.

4. Feature Fusion Module

The Feature Fusion Module integrates the features obtained from both static and dynamic analysis into a unified feature vector.

This fusion enhances the overall representation of the application by combining structural and behavioral characteristics. Techniques such as normalization, dimensionality reduction, and weighted fusion are applied to ensure consistency and relevance of features. The fused dataset provides a more comprehensive input for the classification model, improving detection accuracy.

5. Deep Learning Classification Module

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

This module uses deep learning algorithms to classify applications as benign or malicious. The fused feature vector is fed into models such as Artificial Neural Networks (ANN), Convolutional Neural Networks (CNN), or Long Short-Term Memory (LSTM) networks. The model is trained on labeled datasets and learns complex patterns associated with malware behavior. Optimization techniques and loss functions are used to improve model performance. The output is a probability score indicating whether the application is malicious or safe.

6. XAI (Explainable AI) Module

The XAI Module enhances transparency and interpretability of the deep learning model's decisions. Techniques such as SHAP (Shapley Additive Explanations) or LIME (Local Interpretable Model-Agnostic Explanations) are used to identify the contribution of each feature in the classification process. This module helps users understand why a particular APK is classified as malware, thereby increasing trust and usability of the system.

7. Web Interface (Flask)

The Web Interface Module provides an interactive platform for users to interact with the system. Built using the Flask framework, it allows users to upload APK files, initiate analysis, and view results in a user-friendly format. The interface displays classification results, feature importance, and visual explanations generated by the XAI module. It ensures smooth communication between the backend processing modules and the end user.

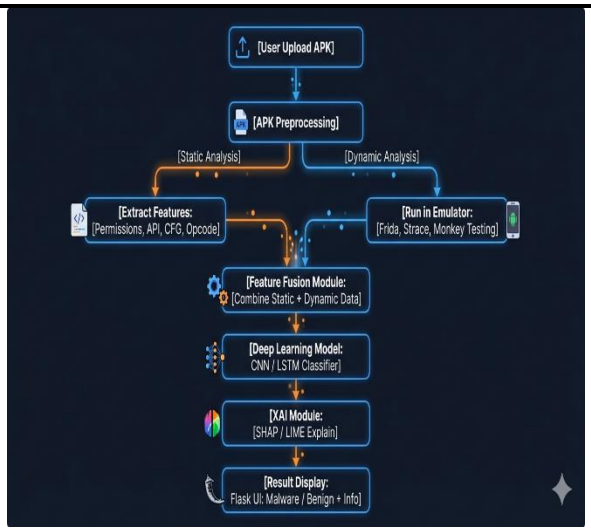

Fig 1: System Architecture Flowchart

5. EXPERIMENTAL RESULTS

The proposed system is evaluated using a dataset consisting of both benign and malicious Android applications.

Performance Metrics

To evaluate the effectiveness of the proposed Android malware detection system, standard classification performance metrics are used. These metrics provide a comprehensive assessment of the model's accuracy and reliability.

1. Accuracy

Accuracy measures the overall correctness of the model by calculating the proportion of correctly classified instances (both malware and benign) out of the total samples.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

- **TP (True Positive):** Malware correctly identified as malware
- **TN (True Negative):** Benign apps correctly identified as benign
- **FP (False Positive):** Benign apps incorrectly classified as malware
- **FN (False Negative):** Malware incorrectly classified as benign

2. Precision

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

Precision indicates how many of the applications predicted as malware are actually malware. It reflects the model's ability to avoid false alarms.

$$Precision = \frac{TP}{TP+FP}$$

- High precision means fewer false positives.

3. Recall (Detection Rate)

Recall measures the ability of the model to correctly identify actual malware instances.

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

- High recall means fewer false negatives (important for security systems).

4. F1-Score

F1-Score is the harmonic mean of Precision and Recall, providing a balance between both metrics.

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision+Recall}$$

- Useful when there is an **imbalance in dataset** (common in malware detection).

Model	Accuracy	Precision	Recall	F1-Score
Static Only	88%	85%	87%	86%
Dynamic Only	90%	88%	89%	88%
Hybrid Model	96%	95%	94%	94.5%

Fig 2: Performance Metrics Formulas

The results demonstrate that the hybrid approach significantly outperforms individual analysis techniques.

IMAGES:

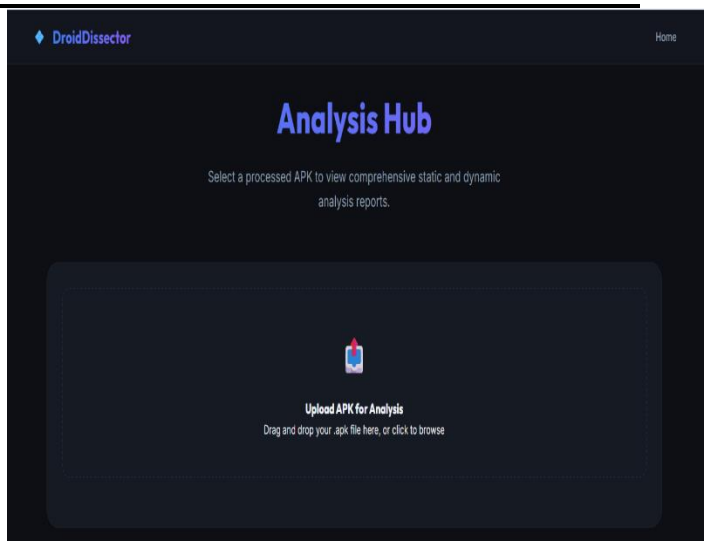

Fig 3: Analysis Hub (Web Interface)

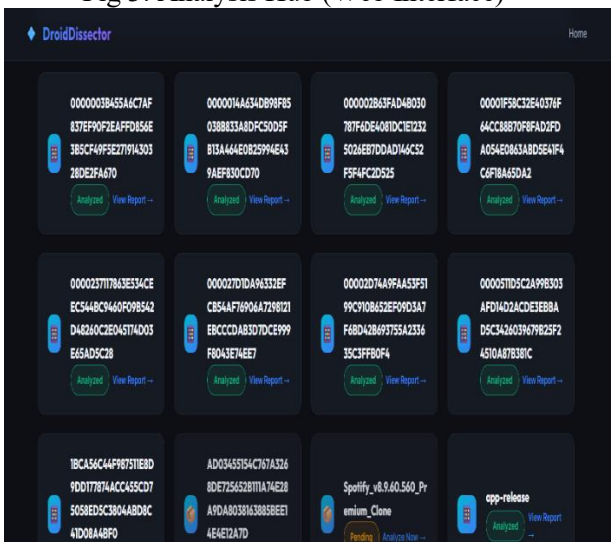

Fig 4: Comprehensive Static Analysis Report

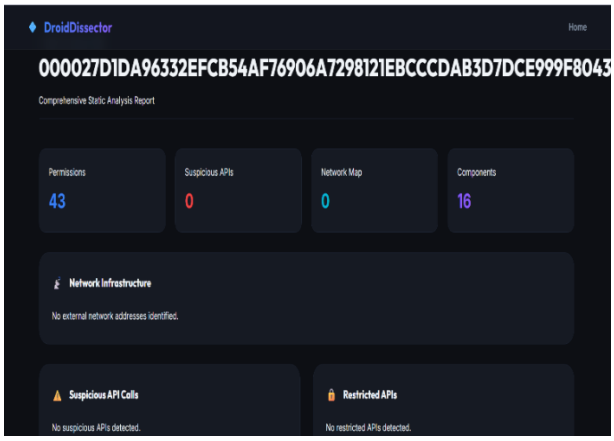

Fig 5: Dashboard of Analyzed APKs

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

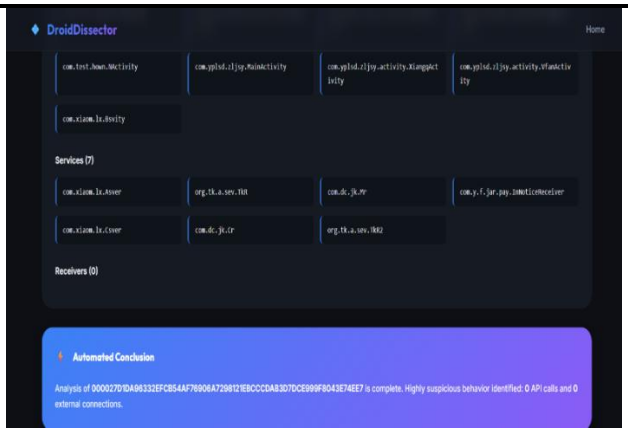

Fig 6: Automated Conclusion and Component View

5. CONCLUSION

This paper presents a hybrid Android malware detection framework that integrates static and dynamic analysis with deep learning and Explainable Artificial Intelligence (XAI) to address the growing challenges of mobile security. By combining static features such as permissions, API calls, and control flow graphs with dynamic behavioral data including system calls and runtime interactions,

the proposed approach provides a more comprehensive understanding of application behavior. This hybrid feature fusion significantly enhances the system's ability to detect sophisticated and obfuscated malware that may evade traditional detection techniques. The incorporation of deep learning models enables effective pattern recognition and classification, resulting in improved accuracy, precision, and recall compared to standalone methods. Furthermore, the integration of XAI techniques adds transparency to the decision-making process, allowing users and analysts to interpret the model's predictions with greater confidence. The development of a Flask-based web interface further improves usability by providing an accessible platform for malware analysis. Overall, the proposed system demonstrates a scalable, robust, and efficient solution for Android malware detection. Future work may focus on real-time detection, deployment in cloud environments, and the integration of more advanced deep learning architectures.

Future Scope

The proposed hybrid Android malware detection system can be further enhanced in several ways to improve its effectiveness and scalability. Future work may focus on enabling real-time malware detection on mobile devices, allowing threats to be identified during application installation or execution. Integration with cloud-based platforms can provide scalable processing and faster analysis of large volumes of APK files. Advanced deep learning techniques, such as Transformer models and Graph Neural Networks, can be explored to capture complex behavioral patterns more accurately. The system can also be extended to detect zero-day and highly obfuscated malware using adaptive learning methods. Expanding the framework to support cross-platform environments, including iOS and Windows, would increase its applicability. Additionally, improvements in Explainable AI techniques can enhance user understanding of model decisions. Continuous dataset updates and integration with existing cybersecurity tools can further strengthen the system's robustness against evolving threats.

REFERENCES

1. A. Arp et al., "DREBIN: Effective and Explainable Detection of Android Malware," NDSS, 2014.
2. W. Enck et al., "TaintDroid: An Information-Flow Tracking System," OSDI, 2010.
3. Y. Aafer et al., "DroidAPIMiner: Mining API-Level Features," SecureComm, 2013.
4. S. Hou et al., "Deep Learning for Android Malware Detection," IEEE Access, 2017.
5. M. Grace et al., "RiskRanker: Scalable and Accurate Detection," MobiSys, 2012.
6. B. Amos et al., "Deep Learning for Android Malware Detection," ACSAC, 2013.
7. L. Nataraj et al., "Malware Images: Visualization and Classification," VizSec, 2011.

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

8. R. Kim et al., "Explainable AI for Cybersecurity," IEEE, 2019.
9. I. Goodfellow et al., *Deep Learning*, MIT Press, 2016.
10. M. Ribeiro et al., "Why Should I Trust You? Explaining Classifier Predictions," KDD, 2016.