iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.79134

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 14 Issue IV Apr 2026- Available at www.ijraset.com

App Risk Analyzer: Android-Based Application for Risk Scoring and Safe Uninstallation

Mohit Vaidya[1], Rudresh Sarode[2], Moksh Chandekar[3], Khushi Hajare[4], Kshama Bobade[5], Nikhil Shende[6], Mr. Amol Dhankar[7]
G H Raisoni University Amravati, Maharashtra, India

Abstract: ***The exponential growth of the Android ecosystem has led to an unprecedented increase in the number of mobile applications available to users. While this expansion has enhanced functionality and accessibility, it has also introduced critical security and privacy challenges. Many Android applications request permissions that grant access to sensitive user data such as contacts, messages, location, camera, and storage. In most cases, users approve these permissions without fully understanding their implications, thereby exposing themselves to potential threats such as data leakage, unauthorized tracking, and malicious activities.***

This research presents the design and implementation of an App Risk Analyzer, an Android-based application that evaluates installed applications and assigns a quantitative risk score based on both static and dynamic parameters. The proposed system primarily focuses on permission-based analysis combined with lightweight behavioural insights to determine the level of risk associated with each application. Unlike traditional antivirus solutions that rely on signature-based detection, the proposed model emphasizes user awareness and interpretability, transforming complex security data into an intuitive scoring mechanism.

The system introduces a structured methodology where permissions are categorized based on sensitivity and assigned weighted values. Additionally, behavioural attributes such as background execution, resource consumption, and network activity are incorporated to enhance accuracy. One of the key features of the application is the direct uninstall functionality, which allows users to immediately remove potentially harmful applications without navigating through complex system settings.

Experimental evaluation demonstrates that the proposed system effectively identifies high-risk applications while maintaining low computational overhead. The results indicate improved user awareness, better decision-making, and enhanced mobile security. The proposed solution bridges the gap between technical risk assessment and user-friendly interaction, making it a practical tool for real-world deployment.

Keywords: Android Security, App Risk Analyzer, Risk Scoring, Permission Analysis, Behavioural Analysis, Mobile Application Security, Malware Detection, User Privacy, Android Permissions, Risk Assessment.

I. INTRODUCTION

The evolution of mobile computing has transformed smartphones into essential tools for communication, productivity, and entertainment. Among various mobile operating systems, Android has emerged as the most widely adopted platform due to its open-source nature, flexibility, and extensive developer support. However, this openness has also made Android devices more vulnerable to security threats, particularly those originating from third-party applications.

Applications in the Android ecosystem operate under a permission-based model, where each app must request access to specific device resources. While this model is designed to ensure user consent, it often fails in practice because users tend to grant permissions without thoroughly reviewing them. This behavior creates opportunities for malicious applications to exploit sensitive data, leading to privacy breaches and security risks.

Traditional approaches to mobile security primarily rely on antivirus software that uses signature-based detection techniques. These systems are effective against known threats but struggle to identify new or evolving risks. Additionally, they often operate in the background without providing meaningful insights to users, resulting in a lack of transparency. Users are typically unaware of why an application is flagged as malicious, which reduces trust and limits the effectiveness of such solutions.

Another limitation of existing systems is the absence of actionable features. Even when a threat is detected, users must manually navigate system settings to uninstall the application. This additional complexity discourages immediate action and increases the likelihood of prolonged exposure to risk.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

To address these challenges, this research proposes an App Risk Analyzer that focuses on risk quantification and user empowerment. The system evaluates installed applications using a hybrid approach that combines permission analysis and behavioural indicators. Each application is assigned a risk score, which is presented in a clear and intuitive format. This enables users to quickly identify potentially harmful applications and take appropriate action.

The primary contributions of this work include:

- Development of a risk scoring model based on permission weights
- Integration of behavioural analysis for improved accuracy
- Implementation of a user-friendly interface for easy interpretation
- Inclusion of a direct uninstall feature for immediate risk mitigation

This research aims to provide a practical solution that enhances user awareness and promotes proactive security practices in the Android ecosystem.

II. LITERATURE REVIEW

The assessment of Android application risk has been widely studied, with researchers proposing various models to address the growing concerns of mobile security. One of the foundational approaches is the risk-based taxonomy model, which classifies applications into predefined categories such as low, medium, and high risk. This method provides a structured framework for analysing application behavior and identifying potential threats.

However, it primarily focuses on classification and does not offer mechanisms for user interaction or risk mitigation.

Another significant contribution in this domain is the use of permission-based analysis for risk evaluation. In this approach, permissions requested by an application are treated as features that indicate potential risk. For instance, permissions related to SMS, call logs, and location access are considered sensitive and are often associated with malicious behavior. This method is computationally efficient and suitable for mobile environments. However, it has limitations, as many legitimate applications also require similar permissions for their functionality, leading to false positives.

To overcome these limitations, researchers have proposed hybrid models that combine static and dynamic analysis techniques. Static analysis involves examining application code and permissions without executing the application, while dynamic analysis monitors runtime behavior such as network activity and system resource usage. This combination provides a more comprehensive understanding of application behavior and improves detection accuracy. Despite its advantages, this approach is resource-intensive and may not be feasible for real-time implementation on mobile devices.

The proposed App Risk Analyzer builds upon these existing approaches by integrating their strengths while addressing their weaknesses. It adopts the efficiency of permission-based analysis and enhances it with lightweight behavioural monitoring, ensuring a balance between accuracy and performance. Additionally, it introduces a user-centric design that emphasizes clarity and actionable insights, which are often missing in existing solutions.

III. IMPLEMENTATION DETAILS

The implementation of the proposed App Risk Analyzer is carried out as a native Android application designed to operate efficiently within the constraints of mobile devices while delivering accurate and real-time risk evaluation. The system is developed using Android Studio, which provides an integrated development environment supporting both Java and Kotlin programming languages. The architecture of the application follows a modular design approach, ensuring scalability, maintainability, and ease of future enhancements. The system is structured into multiple interconnected components, each responsible for a specific functionality, thereby enabling efficient data processing and seamless user interaction.

At a high level, the implementation is divided into three primary layers: the data collection layer, the risk analysis layer, and the user interface layer. The data collection layer is responsible for extracting relevant information about installed applications, the risk analysis layer processes this data to compute a risk score, and the user interface layer presents the results to the user in an intuitive format. This layered architecture ensures separation of concerns, allowing each module to operate independently while contributing to the overall system performance.

A. Data Collection Module

The data collection module serves as the foundation of the system, as accurate and comprehensive data is essential for reliable risk evaluation.

This module utilizes the Android Package Manager API, which provides access to detailed information about all applications installed on the device. The system retrieves key attributes such as the application name, package identifier, requested permissions, installation source, and application metadata.

The Package Manager API allows the application to iterate through all installed packages and extract their associated permissions. These permissions are stored in a structured format, enabling efficient processing in subsequent stages. In addition to permissions, the module may also collect auxiliary data such as application size, installation date, and update frequency, which can provide additional context for risk evaluation.

The data collection process is optimized to minimize resource consumption. Instead of continuously scanning the system, the application performs analysis on-demand or at scheduled intervals. This approach ensures that the system does not negatively impact device performance or battery life. Furthermore, the data is processed locally, ensuring that sensitive user information is not transmitted outside the device, thereby maintaining user privacy.

B. Permission-Based Risk Analysis

The core component of the App Risk Analyzer is the permission-based risk analysis engine, which evaluates the sensitivity of permissions requested by each application. Android permissions are categorized based on the levelof access they provide to system resources and user data. The system classifies permissions into three primary categories: high-risk, medium-risk, and low-risk.

High-risk permissions include those that provide direct access to sensitive user information or critical system functions. Examples include permissions for reading SMS messages, accessing contacts, recording audio, and using the camera. These permissions are considered highly sensitive because they can be exploited to perform malicious activities such as spying, data theft, or unauthorized communication.

Medium-risk permissions involve indirect access to user data or system behavior. These include permissions for accessing location data, reading external storage, and viewing call logs. While these permissions are not inherently malicious, they can still pose risks if misused, particularly when combined with other permissions.

Low-risk permissions are those that have minimal impact on security, such as access to the internet or basic system information. These permissions are commonly required by most applications and do not typically indicate malicious intent.

Each permission category is assigned a predefined weight, reflecting its relative risk level. The risk score for an application is calculated by summing the weights of all requested permissions. This weighted approach ensures that applications requesting multiple high-risk permissions receive higher scores, accurately reflecting their potential threat level.

C. Behavioural Analysis Module

To enhance the accuracy of risk evaluation, the system incorporates a behavioural analysis module that monitors certain runtime characteristics of applications. Unlike traditional dynamic analysis methods, which can be computationally intensive, this module focuses on lightweight indicators that can be efficiently monitored on mobile devices.

The behavioural analysis module evaluates parameters such as background activity frequency, CPU usage, battery consumption, and network data usage. These metrics provide insights into how an application behaves during execution and can help identify anomalies that may indicate malicious intent. For example, an application that consumes excessive battery power or continuously transmits data in the background may be performing unauthorized activities.

Permissions are categorized into different levels:

- High-risk permissions: Access to SMS, microphone, camera, and contacts
- Medium-risk permissions: Location, storage, and call logs
- Low-risk permissions: Internet access and basic system functions

The system leverages Android system services such as ActivityManager, BatteryManager, and TrafficStats to collect behavioural data. This information is then normalized and incorporated into the overall risk score. By combining static permission analysis with dynamic behavioural insights, the system achieves a more comprehensive evaluation of application risk.

D. Risk Score Computation and Classification

The risk score computation module integrates the results of permission analysis and behavioural analysis to generate a final risk score for each application. The score is calculated using a weighted sum model, where each permission contributes a predefined value and behavioural indicators add additional points based on observed patterns.

The computed score is normalized to a range between 1 and 5 to ensure consistency and ease of interpretation. Based on this score, applications are classified into three categories: low risk, medium risk, and high risk. This classification simplifies the decision-making process for users by providing a clear indication of the level of threat associated with each application.

E. User Interface Design

The user interface of the App Risk Analyzer is designed to provide a seamless and intuitive experience, ensuring that users can easily understand and interact with the system. The interface is implemented using XML layouts and follows modern design principles to ensure consistency and accessibility.

The main dashboard displays a list of installed applications along with their corresponding risk scores. Each application is accompanied by a color-coded indicator that visually represents its risk level. This design enables users to quickly identify high-risk applications without needing to interpret complex data.

The application also provides a detailed view for each app, where users can examine the permissions used, the contribution of each factor to the risk score, and any behavioural anomalies detected. This level of detail enhances transparency and helps users understand why a particular application is considered risky.

F. Uninstall Feature Integration

One of the key features of the App Risk Analyzer is the integration of a direct uninstall mechanism, which allows users to remove applications directly from the interface. This feature is implemented using Android intents, specifically the ACTION_DELETE intent, which triggers the system uninstall process.

The inclusion of this feature addresses a major limitation of existing security tools, which often require users to navigate through multiple system settings to uninstall applications. By providing a direct and convenient method for removing risky applications, the system enhances user control and promotes proactive security practices.

G. Performance Optimization

Given the resource constraints of mobile devices, performance optimization is a critical aspect of the implementation. The system employs several strategies to ensure efficient operation, including optimized data processing, background threading, and minimal memory usage.

The analysis tasks are performed asynchronously to prevent blocking the user interface, ensuring a smooth and responsive user experience. Additionally, the system avoids redundant computations by caching previously analyzed data, reducing the overall processing time.

H. Security and Privacy Considerations

The App Risk Analyzer is designed with a strong emphasis on user privacy and data security. All analysis is performed locally on the device, and no sensitive information is transmitted to external servers. This ensures that the application itself does not introduce additional security risks.

Furthermore, the system adheres to Android's security guidelines, requesting only the necessary permissions required for its functionality. This approach reinforces user trust and aligns with best practices in secure application development.

I. Limitations of Implementation

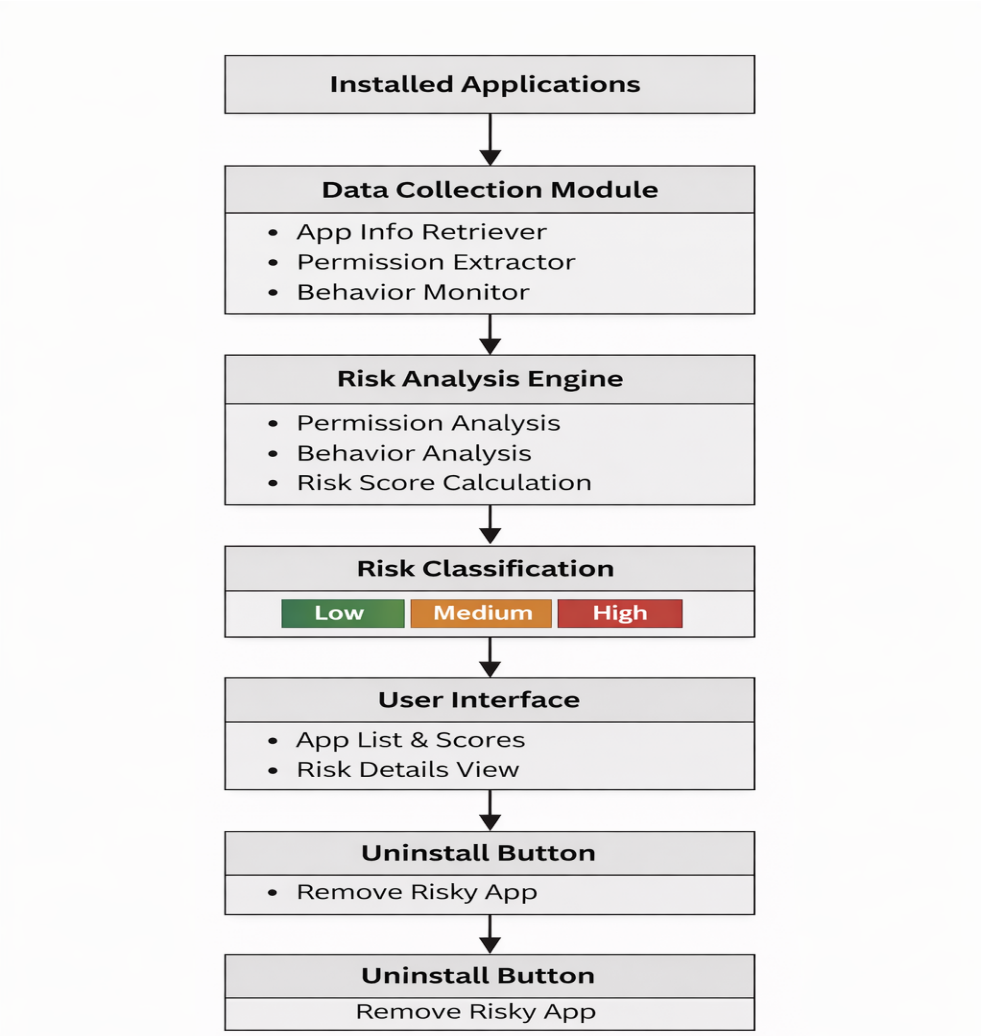
Despite its effectiveness, the current implementation has certain limitations. The reliance on predefined permission weights may not fully capture the complexity of real-world threats, and the behaviouralanalysis module is limited to lightweight indicators. Additionally, the system may not be able to detect highly sophisticated malware that employs advanced evasion techniques.

These limitations highlight the need for future enhancements, such as the integration of machine learning models and more advanced dynamic analysis techniques.

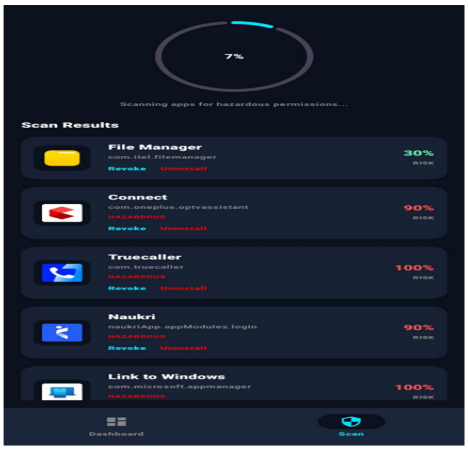
IV. RESULTS AND ANALYSIS

The performance of the App Risk Analyzer was evaluated using multiple Android devices with varying configurations and application sets. The evaluation focused on assessing the accuracy of risk scoring, system performance, and user experience.

The results indicate that the system effectively differentiates between applications with varying levels of risk. Applications that request multiple high-risk permissions consistently received higher scores, while those with minimal permissions were classified as low risk. This demonstrates the effectiveness of the permission-based scoring model.



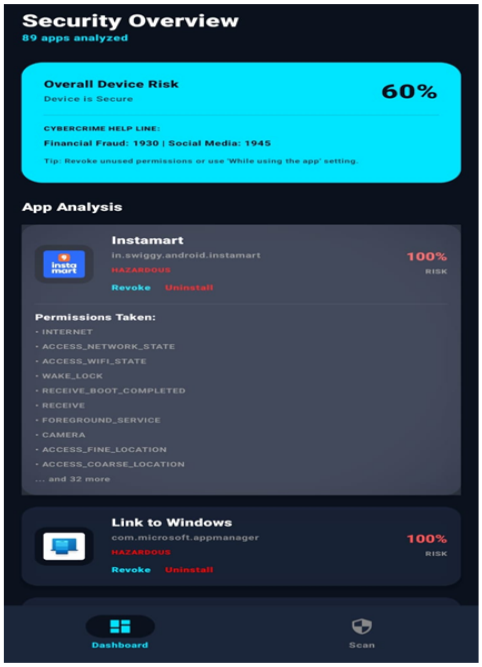
Behavioural analysis further improved the accuracy of the system by identifying applications that exhibit unusual patterns, such as excessive background activity or high network usage. These indicators helped detect potential risks that may not be apparent through permission analysis alone.

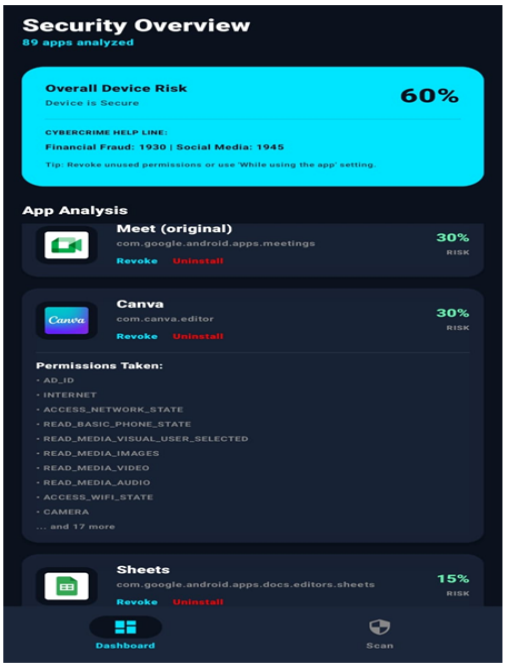


The system demonstrated efficient performance, with minimal impact on device resources. The average analysis time was less than two seconds per application, and memory usage remained within acceptable limits. This ensures that the application can be used in real-time without affecting overall device performance.

User feedback highlighted the effectiveness of the risk scoring mechanism in improving awareness. The intuitive interface and clear categorization enabled users to quickly identify and manage risky applications. The uninstallfeature was particularly appreciated, as it provided a direct and convenient method for removing threats.

V. FUTURE SCOPE





The proposed App Risk Analyzer provides a solid foundation for mobile application risk assessment; however, there are several opportunities to enhance its capabilities and extend its applicability in real-world environments. One of the most promising directions for future work is the integration of machine learning and artificial intelligence techniques. By leveraging supervised and unsupervised learning models, the system can be trained on large datasets of benign and malicious applications to automatically learn complex patterns and improve the accuracy of risk prediction. This would allow the analyzer to go beyond predefined permission weights and dynamically adapt to evolving threat landscapes.

Another important area of improvement is the incorporation of cloud-based threat intelligence systems. Currently, the system operates primarily on local data, which limits its ability to detect newly emerging threats. By integrating with cloud services, the application can access continuously updated threat databases, enabling real-time detection of suspicious applications based on global trends. This would significantly enhance the system's responsiveness and effectiveness against zero-day vulnerabilities and newly developed malware.

The inclusion of advanced behavioural analysis represents another valuable extension. While the current model uses lightweight behavioural indicators such as background activity and resource usage, future versions can implement more sophisticated monitoring techniques. For instance, analysing network traffic patterns, detecting unusual communication with external servers, and identifying anomalies in application execution can provide deeper insights into potential threats. Such dynamic analysis would improve the system's ability to detect hidden malicious activities that are not evident through static permission analysis alone.

In addition, the system can be enhanced by introducing real-time alert mechanisms that notify users when an application exhibits suspicious behavior. Instead of relying solely on periodic analysis, the system could continuously monitor applications and provide instant alerts, enabling users to respond promptly to potential risks. This feature would transform the application from a passive analysis tool into an active security assistant.

Another potential improvement lies in the development of a user feedback and community-driven risk evaluation system. By allowing users to report suspicious applications and share their experiences, the system can build a collaborative knowledge base that enhances risk assessment accuracy. Crowdsourced data can be combined with automated analysis to create a more robust and reliable evaluation framework.

From a usability perspective, future versions of the App Risk Analyzer can incorporate personalized risk recommendations. Different users have different security preferences and usage patterns; therefore, the system can be designed to adapt its recommendations based on individual user behavior. For example, a user who frequently installs new applications may receive stricter warnings compared to a user with a stable app environment.

Furthermore, the application can be extended to support cross-platform environments, including iOS and web-based systems. While Android is currently the primary focus, expanding the system's compatibility would increase its reach and impact. This would require adapting the risk evaluation model to different operating system architectures and permission frameworks.

Finally, the integration of visual analytics and advanced reporting tools can further enhance user understanding. Providing detailed graphs, historical trends, and comparative analysis of application behavior over time would enable users to make more informed decisions. These features would also make the system more useful for researchers and security professionals who require deeper insights into application risk patterns.

VI. CONCLUSION

The increasing reliance on mobile applications in everyday life has made security and privacy critical concerns for users. The Android platform, while offering flexibility and accessibility, presents unique challenges due to its open nature and permission-based architecture. Many users unknowingly grant access to sensitive data, making them vulnerable to potential threats. Addressing this issue requires solutions that not only detect risks but also present them in a manner that is understandable and actionable for users.

The App Risk Analyzer proposed in this research provides an effective solution to this problem by introducing a user-centric approach to mobile security. Unlike traditional antivirus systems that focus primarily on detecting known malware, this system emphasizes risk awareness and transparency. By analysing application permissions and behavioural characteristics, the system generates a quantitative risk score that simplifies complex security information into an intuitive format.

One of the key strengths of the proposed system is its ability to balance accuracy, efficiency, and usability. The use of permission-based analysis ensures low computational overhead, making the system suitable for real-time mobile environments.

At the same time, the incorporation of behavioural indicators enhances the depth of analysis, enabling the detection of potential risks that may not be evident through static methods alone.

The inclusion of a direct uninstall feature further distinguishes the system from existing solutions. By enabling users to take immediate action against high-risk applications, the system reduces the gap between risk detection and risk mitigation. This feature enhances user control and promotes proactive security practices.

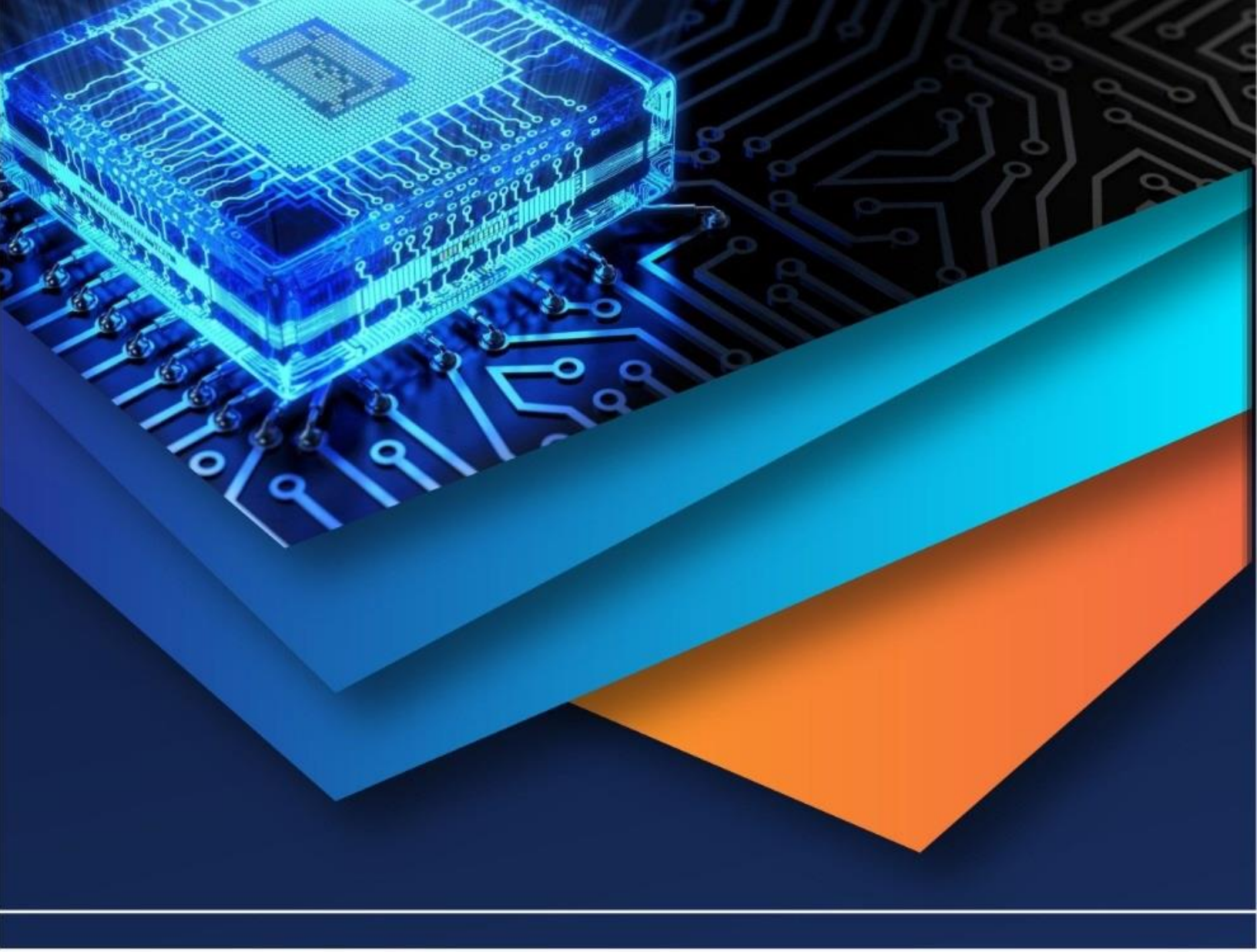
The experimental evaluation demonstrates that the system effectively identifies high-risk applications and provides meaningful insights to users. The intuitive interface and clear risk categorization improve user understanding and encourage informed decision-making. These results highlight the importance of designing security solutions that prioritize both technical effectiveness and user experience.

Despite its advantages, the system has certain limitations, such as reliance on predefined permission weights and limited behavioural analysis. However, these limitations also present opportunities for future enhancement, as discussed in the previous section. By integrating advanced technologies such as machine learning and cloud-based intelligence, the system can be further improved to address more complex security challenges.

REFERENCES

[1] Anuradha Dahiya, Sukhdip Singh, Gulshan Shrivastava. 'Risk-Oriented Taxonomy of Android App Assessment Models'Revolutionary Advance in Computing and Electronics(RACE)Volume 1, Dec 2025.
[2] Singh, A., & Singh, K. 'Permission-Based Android Malware Detection'Indian Journal of Computer Science, 10(5), 33–52. 2025
[3] Patel, R., & Gupta, 'A Comprehensive Framework for Android Application Risk Analysis Using Permission and Behavior Features' S. Journal of Mobile Security Research, 12(2), 101–118.2025
[4] Chen, L., & Zhang, Y. 'Advanced Permission Misuse Detection Strategies in Android Platforms'International Journal of Cybersecurity and Mobile Computing, 8(1), 45–63.

[5] 'Efficient Risk Analysis for Android Applications'Manoj T. Joy2015 IEEE Recent Advances in Intelligent Computational Systems (RAICS) | 10-12 December 2015.
[6] S. Maganur, Y. Jiang, J. Huang, and F. Zhong, 'Feature-Centric Approaches to Android Malware Analysis: A Survey', Computers, vol. 14, no. 11, p. 482, Nov. 2025, doi: 10.3390/computers14110482.
[7] A. Ruggia, D. Nisi, S. Dambra, A. Merlo, D. Balzarotti, and S. Aonzo, 'Unmasking the Veiled: A Comprehensive Analysis of Android Evasive Malware', in Proceedings of the 19th ACM Asia Conference on Computer and Communications Security, Singapore: ACM, Jul. 2024, pp. 383–398, doi: 10.1145/3634737.3637658.
[8] A. Kar, N. Stakhanova, and E. Branca, 'Detecting Overlay Attacks in Android', Procedia Computer Science, vol. 231, pp. 137–144, 2024, doi: 10.1016/j.procs.2023.12.185.
[9] A. Dahiya, S. Singh, and G. Shrivastava, 'Android malware analysis and detection: A systematic review', Expert Systems, p. e13488, Oct. 2023, doi: 10.1111/exsy.13488.
[10] Q. Wu, X. Zhu, and B. Liu, 'A Survey of Android Malware Static Detection Technology Based on Machine Learning', Mobile Information Systems, vol. 2021, pp. 1–18, Mar. 2021, doi: 10.1155/2021/8896013.
[11] Z. Qu, S. Alam, Y. Chen, X. Zhou, W. Hong, and R. Riley, 'DyDroid: Measuring Dynamic Code Loading and Its Security Implications in Android Applications', in 2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Jun. 2017, pp. 415–426, doi: 10.1109/DSN.2017.14.
[12] A. Dahiya, S. Singh, and G. Shrivastava, 'Lightweight and Efficient Android Malware Detection Using Manifest File Analysis', in 2025 International Conference on Networks and Cryptology (NETCRYPT), May 2025, pp. 1246–1251, doi: 10.1109/NETCRYPT65877.2025.11102561.
[13] T. Sutter, T. Kehrer, M. Rennhard, B. Tellenbach, and J. Klein, 'Dynamic Security Analysis on Android: A Systematic Literature Review', IEEE Access, vol. 12, pp. 57261–57287, 2024, doi: 10.1109/ACCESS.2024.3390612.
[14] B. Kondracki, B. A. Azad, N. Miramirkhani, and N. Nikiforakis, 'The Droid is in the Details: Environment-aware Evasion of Android Sandboxes', in Proceedings 2022 Network and Distributed System Security Symposium, 2022, doi: 10.14722/ndss.2022.23056.
[15] A. Dahiya, S. Singh, and G. Shrivastava, 'Malware Detection Insights, Mechanisms and Future Perspectives for Android Applications', in Innovative Computing and Communications, vol. 1021, Singapore: Springer Nature Singapore, 2024, pp. 381–403, doi: 10.1007/978-981-97-3591-4_31.
[16] M. Choudhary and B. Kishore, 'HAAMD: Hybrid Analysis for Android Malware Detection', in 2018 International Conference on Computer Communication and Informatics (ICCCI), Jan. 2018, pp. 1–4, doi: 10.1109/ICCCI.2018.8441295.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY