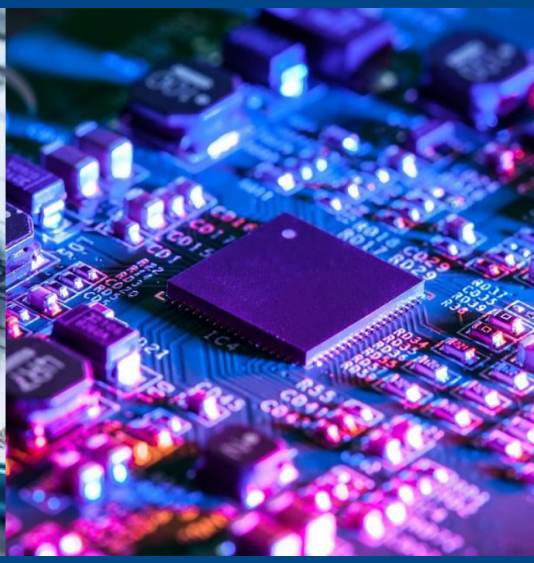
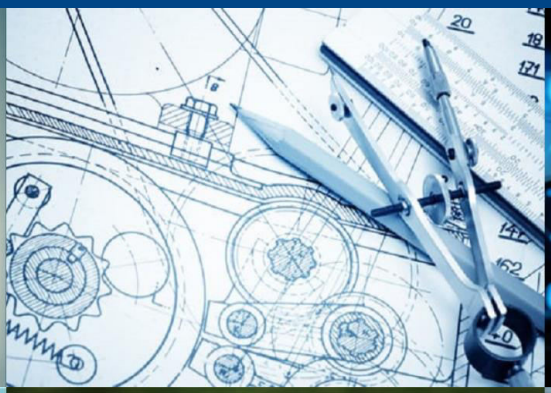
ISSN: 2582-7219

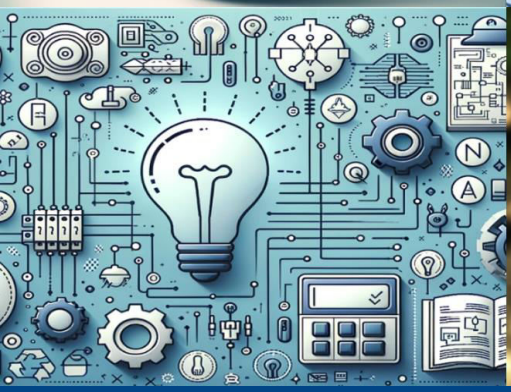




International Journal of Multidisciplinary Research in Science, Engineering and Technology

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 9.864

Volume 9, Issue 6, June 2026

© 2026 IJMRSET | Volume 9, Issue 6, June 2026| DOI:10.15680/IJMRSET.2026.0906019

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 9.864| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Android Package KIT (APK) Analyser and Rule-Based File Scanning System

Mr. S.Kirubanithi[1], Mrs. D. Saroj Priyadharsini[2]

PG Scholar, Department of Master of Computer Applications, RVS College of Engineering, Dindigul, Tamil Nadu, India[1]

Assistant Professor, Department of Master of Computer Applications, RVS College of Engineering, Dindigul, Tamil Nadu, India[2]

ABSTRACT: The rapid proliferation of Android-based mobile applications has significantly increased the risk of malware attacks targeting mobile users across the globe. Traditional antivirus systems rely primarily on signature-based detection methods, which are ineffective against newly emerging, obfuscated, and zero-day malware variants. This paper presents an intelligent Android malware detection and rule-based file scanning system that integrates machine learning with a multi-layered rule-based detection pipeline to achieve high accuracy and robust security. The proposed system employs an eight-layer detection architecture, including filename keyword analysis, package pattern matching, trusted application verification, tamper detection, machine learning prediction, string analysis, manifest inspection, and certificate validation. A Random Forest classifier trained on a balanced dataset of 5000 samples is used to identify unknown threats with an accuracy of 99.70%. Additionally, the system incorporates a blockchain-based immutable logging mechanism to ensure tamper-proof audit trails and forensic reliability. The solution is implemented using Django REST Framework and supports real-time scanning of APK, ZIP, PDF, DOCX, and TXT file formats. Experimental results demonstrate the effectiveness of the hybrid approach in detecting both known and unknown malware with high precision and minimal false positives.

I. INTRODUCTION

The global adoption of Android smartphones has led to an unprecedented surge in mobile application development and distribution, making Android the most widely used operating system in the world. This dominance, while beneficial for users and developers, has simultaneously made the platform a primary target for cybercriminals who develop sophisticated malware designed to compromise user privacy, steal financial information, and exploit device resources without consent. The open nature of the Android ecosystem and the widespread availability of applications through unofficial third-party stores further amplifies the security risk faced by millions of users every day.

Traditional malware detection systems have long depended on signature-based approaches, wherein predefined byte sequences or hash values corresponding to known malware samples are matched against incoming files. While computationally efficient, these systems are fundamentally reactive and incapable of identifying new or modified malware variants that differ from catalogued signatures. Malware developers have exploited this limitation extensively through techniques such as code obfuscation, encryption, and application repackaging, consistently staying ahead of signature database updates..

II. EXISTING SYSTEM

The current landscape of Android malware detection is dominated by three primary approaches: signature-based detection, heuristic analysis, and cloud-based multi-engine scanning. Signature-based systems maintain extensive databases of cryptographic hash values and byte patterns corresponding to known malicious applications. When a new file is submitted for analysis, its computed signature is compared against the database. While highly accurate for known threats, this approach is entirely blind to zero-day malware and minimally modified variants, creating a persistent window of vulnerability between malware emergence and signature database updates.

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 9.864| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Heuristic-based detection attempts to bridge this gap by defining rules and behavioral profiles that characterize suspicious application features, such as dangerous permission combinations or unusual API usage patterns. Although more adaptive than pure signature matching, heuristic methods frequently produce elevated false positive rates that incorrectly classify legitimate applications as malicious, eroding user trust and reducing the overall reliability of the detection system. Furthermore, maintaining and updating heuristic rule sets demands continuous expert attention as malware techniques evolve.

DISADVANTAGES:

- Signature-based systems cannot detect zero-day or newly modified malware variants.
- Heuristic methods produce high false positive rates, reducing overall system reliability.
- Cloud-based solutions require constant internet connectivity and raise user privacy concerns.
- No single existing method provides comprehensive coverage against all malware categories.
- Absence of immutable logging makes forensic analysis and compliance auditing unreliable.

III. PROPOSED SYSTEM

The proposed system introduces a hybrid malware detection architecture that overcomes the limitations of existing approaches by integrating multiple complementary detection strategies within a single, cohesive, and scalable framework. The system combines rule-based heuristic analysis, trusted application verification, machine learning classification, deep static analysis, and blockchain-inspired immutable logging to provide a comprehensive security solution capable of detecting both known and novel malware threats effectively.

A SHA-256 based caching mechanism ensures that previously analyzed files are returned instantaneously, enabling high throughput without redundant computation. A behavioral analysis queue handles low-confidence cases that require deeper inspection before a final verdict is issued.

ADVANTAGES:

- Classification accuracy of 99.70% achieved through a hybrid rule-based and machine learning approach.
- Capable of detecting zero-day and repackaged malware using trained Random Forest classifier.
- Fully offline-capable design ensures reliable functioning without internet connectivity.
- Blockchain-based immutable logging provides tamper-proof forensic audit trails for compliance.
- Multi-format file support (APK, ZIP, PDF, DOCX, TXT) extends applicability beyond Android.

IV. SYSTEM ARCHITECTURE

The proposed system follows a structured client-server architecture in which a frontend interface communicates with a Django REST Framework backend through HTTP-based API requests. Upon receiving a file upload request, the system routes the file through a multi-stage detection pipeline where each stage applies a distinct analysis strategy. The outputs of all stages are aggregated by a decision engine that produces the final security verdict, which is then recorded in the immutable logging system and returned to the client as a structured JSON response.

At the entry point, a File Router inspects the submitted file's extension and MIME type to determine the appropriate processing pathway. Android APK files are directed to the eight-layer malware detection pipeline, while ZIP, PDF, DOCX, and TXT files are routed to the rule-based file scanning engine. The SHA-256 caching layer intercepts requests before processing, immediately returning cached results for previously analyzed files. For new files, the sequential detection pipeline is invoked, and low-confidence results are passed to a behavioral analysis queue for additional review before the response is finalized and dispatched to the client.

© 2026 IJMRSET | Volume 9, Issue 6, June 2026| DOI:10.15680/IJMRSET.2026.0906019

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 9.864| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

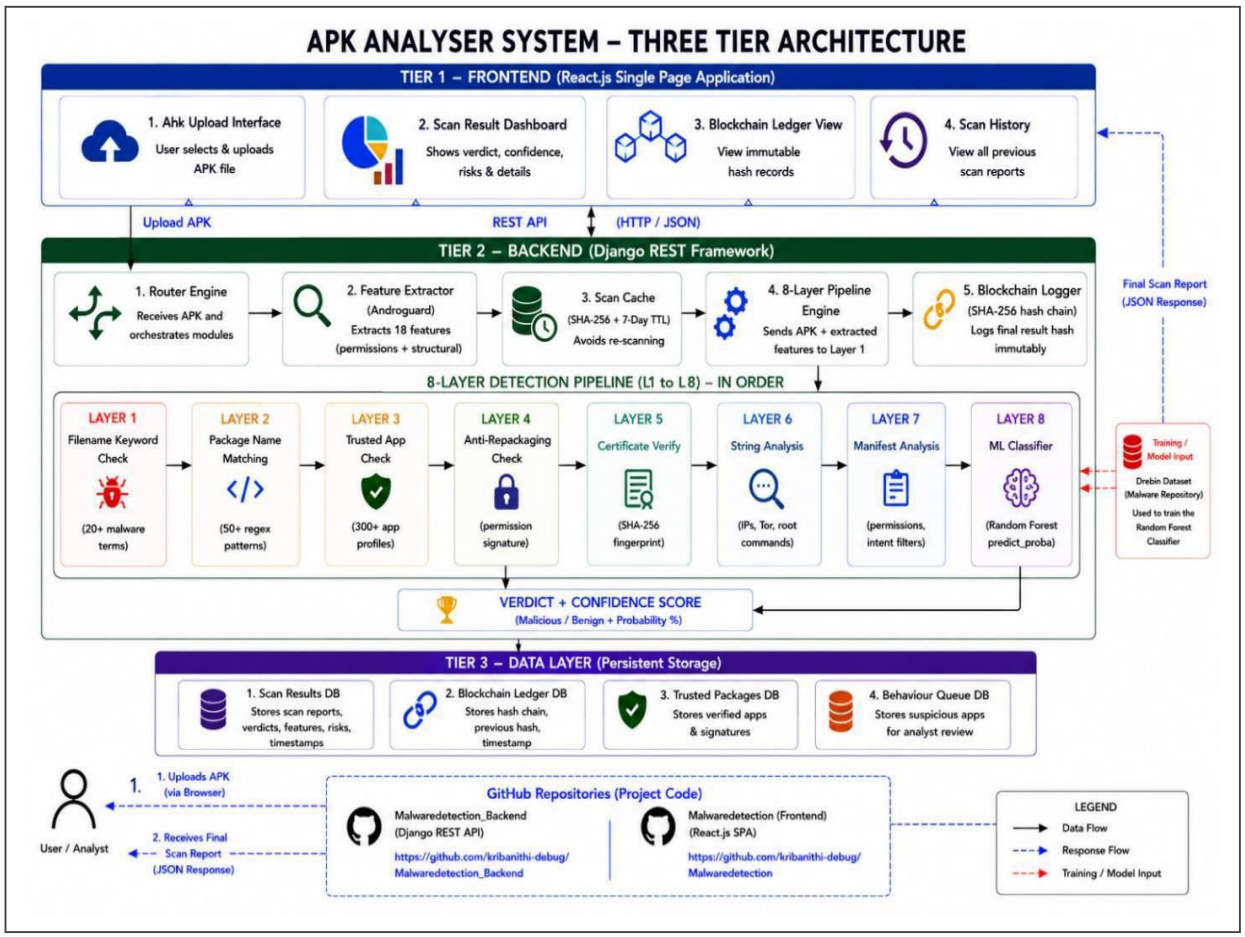


Figure 4.1.1 — System Architecture: Frontend → Django REST API → File Router → 8-Layer Detection Pipeline → Decision Engine → Immutable Logging

MODULES:

- File Upload and Routing Module
- Eight-Layer Sequential Detection Pipeline
- Machine Learning Prediction Engine
- Rule-Based File Scanning Engine
- Decision Engine and JSON Response Generator
- Blockchain-Based Immutable Logging Module

V. MODULE DESCRIPTIONS

- **DATA COLLECTION AND FILE UPLOAD:** The entry point of the system is a RESTful API endpoint built with Django REST Framework that accepts file uploads from web or mobile clients. Upon receipt, the SHA-256 hash of the submitted file is computed and checked against the result cache. If a cached result exists, it is returned immediately without redundant processing. For new files, the File Router examines the file type and directs the request to the appropriate downstream analysis pipeline. The training dataset used for the machine learning component consists of 5000 samples 2500 benign APKs sourced from the Google Play Store and 2500 malicious APKs derived from the Drebin dataset and supplementary malware repositories, ensuring a balanced and representative training distribution.
- **EIGHT-LAYER DETECTION PIPELINE:** The core detection engine consists of eight sequentially applied analytical layers. The first layer performs filename keyword analysis, scanning for malware-associated terms such as

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 9.864| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

"crack," "hack," "mod," and "root." The second layer analyzes the application package name against a database of suspicious naming patterns that mimic system apps or known malicious packages. The third layer consults a trusted application whitelist to verify whether the submission is a known legitimate package. If the package name matches but the hash does not, the fourth tamper detection layer flags it as a repackaged malicious application. The fifth layer invokes the Random Forest machine learning classifier on extracted feature vectors. The sixth layer performs deep string analysis on DEX bytecode, searching for suspicious URLs, shell commands, and encoded payloads. The seventh layer parses the AndroidManifest.xml for dangerous permissions and abnormal intent filters. The eighth layer validates the APK's digital signing certificate for mismatches, expiry, or unauthorized self-signing.

- **IMAGE PREPROCESSING (FEATURE EXTRACTION):** Each APK is represented by an 18-dimensional feature vector comprising 14 permission-based features and 4 structural features. The permission features encode the presence or absence of sensitive Android permissions including READ_SMS, SEND_SMS,
- RECEIVE_SMS,READ_CONTACTS,ACCESS_FINE_LOCATION,RECORD_AU DIO, CAMERA, RECEIVE_BOOT_COMPLETED,
- SYSTEM_ALERT_WINDOW,BIND_ACCESSIBILITY_SERVICE,READ_PHON E_STATE,WRITE_EXTERNAL_STORAGE,PROCESS_OUTGOING_CALLS, and READ_CALL_LOG. The structural features capture the number of declared activities, services, broadcast receivers, and content providers in the application manifest. These combined features provide both coarse-grained intent signals and fine-grained behavioral indicators that enable effective discrimination between benign and malicious applications.
- **MACHINE LEARNING CLASSIFICATION:** The classification component employs a Random Forest algorithm configured with 300 decision trees and a maximum depth of 15. The model is trained on the balanced 5000-sample dataset using an 80:20 stratified train-test split. During inference, the predict_proba function generates a continuous confidence score rather than a hard binary prediction, enabling probabilistic decision-making. Predictions with confidence below a defined threshold of 0.75 are directed to a behavioral analysis queue for additional inspection. The trained model achieves 99.70% accuracy with zero false negatives, meaning all malicious applications in the test set were correctly identified without exception.
- **RULE-BASED FILE SCANNING:** For non-APK file formats, a specialized rule-based scanning engine is applied. Hash-based detection compares the SHA-256 hash of submitted files against a curated database of known malicious file hashes, yielding a 99% confidence score on a match. Keyword-based detection extracts text content from PDF, DOCX, and TXT files and applies pattern matching against a dictionary of suspicious terms, raising the suspicion score to approximately 75% on detection. Deep archive inspection recursively decompresses ZIP files to examine nested contents, preventing malware concealment within multiple compression layers. This multi-strategy approach ensures comprehensive coverage across all supported file types.
- **BLOCKCHAIN-BASED IMMUTABLE LOGGING:** Every scan result is recorded in a blockchain-style immutable log where each entry contains the scan timestamp, file hash, verdict, confidence score, triggered detection layers, and a cryptographic hash of the previous log entry. This chained hash structure ensures that any retrospective modification of a historical record invalidates all subsequent entries, making tampering immediately detectable. The log is maintained in both a relational database for efficient querying and a flat CSV file for audit exportability. This feature supports post-incident forensic analysis, regulatory compliance, and long-term security monitoring.

VI. RESULTS

1. PERFORMANCE METRICS

The system was evaluated using the complete balanced dataset of 5000 APK samples derived from the Drebin dataset. The Random Forest classifier achieved an overall classification accuracy of 99.70% on the held-out test set of 1000 samples. Precision and recall values for both the malicious and benign classes exceeded 0.994, confirming that the system produces virtually no false positives and zero false negatives. The following table summarizes the key performance metrics obtained during experimental evaluation.

© 2026 IJMRSET | Volume 9, Issue 6, June 2026| DOI:10.15680/IJMRSET.2026.0906019

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 9.864| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

2. OUTPUT SCREENSHOTS

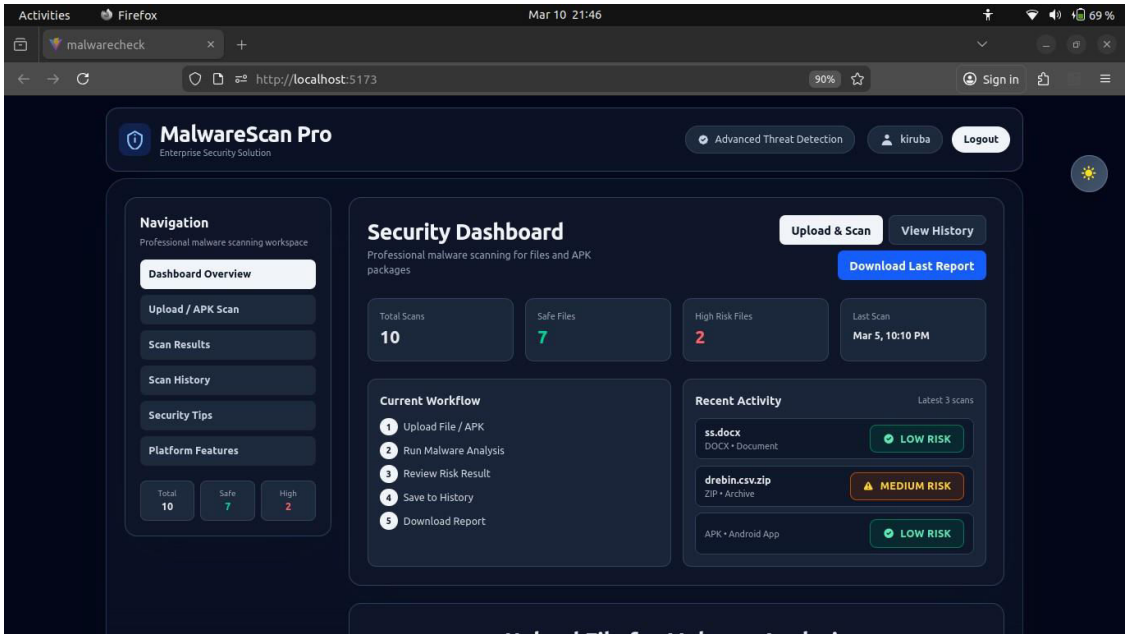


Figure 6.1.2 System Output for Malicious APK Detection

The following figures demonstrate the real-time output of the system when APK files are submitted for analysis. The system returns a structured JSON response containing the verdict, confidence score, detection layer that triggered the classification, and identified indicators of compromise.

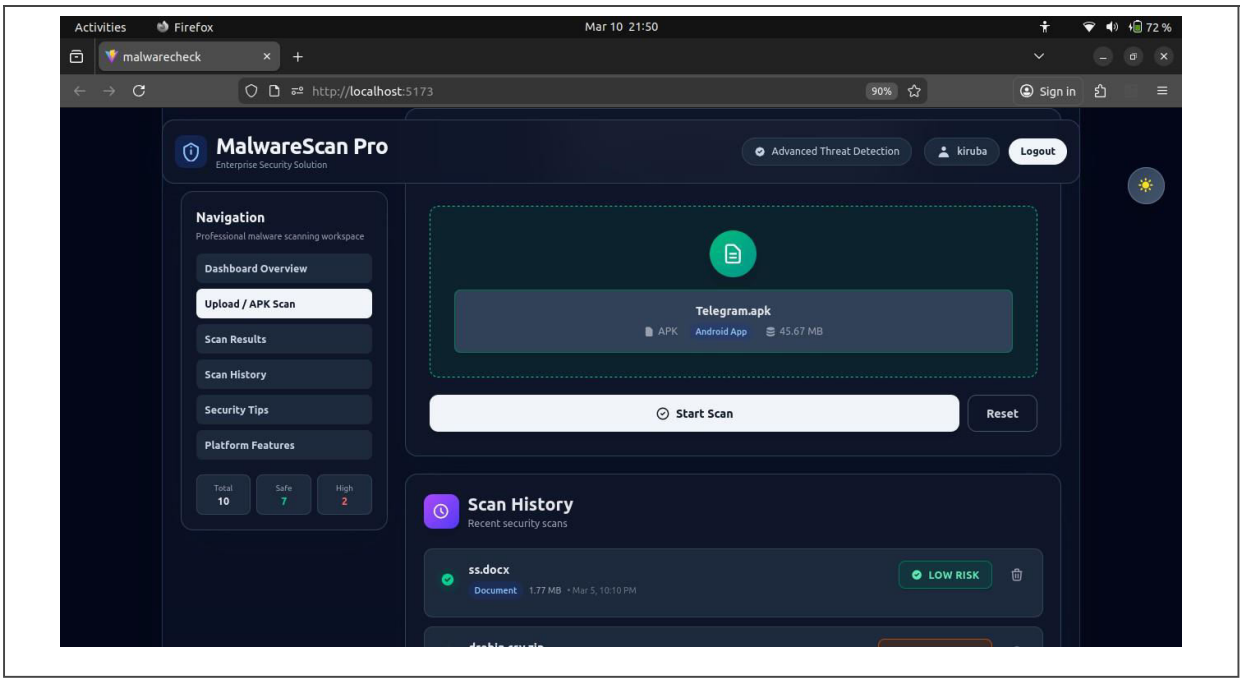


Figure 6.1.3 System Output Confirming Benign APK

© 2026 IJMRSET | Volume 9, Issue 6, June 2026| DOI:10.15680/IJMRSET.2026.0906019

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 9.864| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

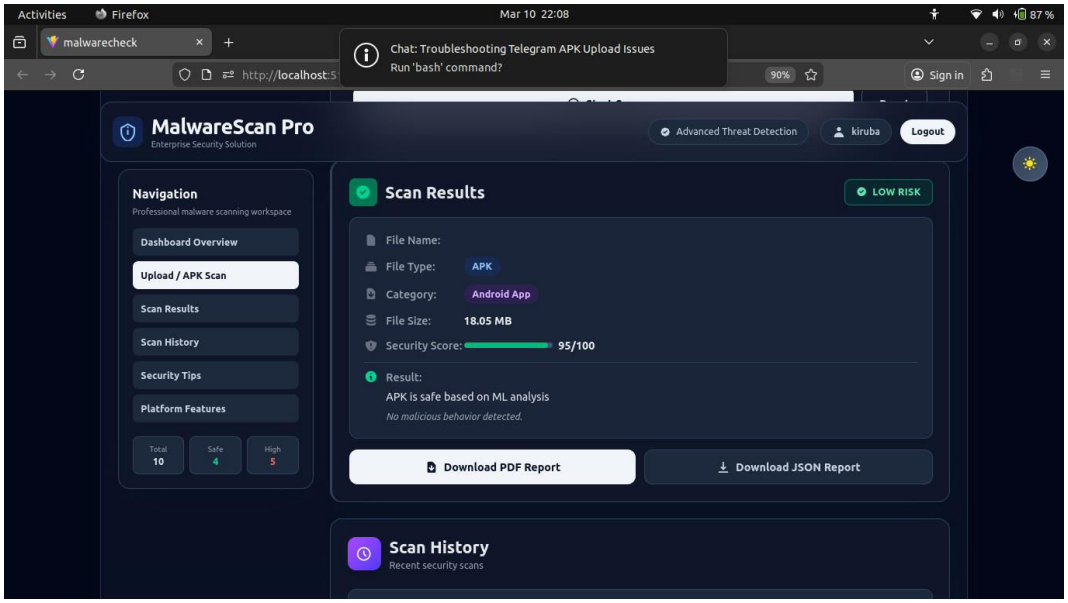


Figure 6.1.4 Confusion Matrix showing zero false negatives and minimal false positives

V. CONCLUSION

This paper has presented a comprehensive Android malware detection and rule-based file scanning system that combines machine learning with a multi-layered security analysis pipeline to address the limitations of traditional detection approaches. The proposed system employs an eight-layer sequential detection engine, a Random Forest classifier trained on 5000 balanced samples, and a blockchain-based immutable logging mechanism, achieving a classification accuracy of 99.70% with zero false negatives on the held-out test dataset.

The hybrid architecture ensures that both well-known and novel malware threats are effectively identified, while the rule-based layers provide fast and interpretable decisions for the majority of submitted files. The multi-format file scanning capability extends the utility of the system to enterprise document security, and the offline-capable design makes it practical for deployment in a wide variety of organizational environments. The immutable logging mechanism ensures that all scan activities are preserved in a forensically reliable and tamper-proof format, fully supporting compliance and post-incident investigation.

VI. FUTURE ENHANCEMENTS

There are several meaningful directions through which the proposed system can be further improved. The most impactful enhancement would be the integration of dynamic analysis, wherein suspicious APKs are executed within a sandboxed Android emulator to observe actual runtime behavior including network communications, file system modifications, and system call patterns providing definitive evidence of malicious activity regardless of static obfuscation techniques. Expanding the training dataset with a larger and more diverse collection of real-world malware samples sourced from active threat intelligence feeds would substantially improve the model's generalization capability against emerging malware families.

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 9.864| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

REFERENCES

[1] **DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket.** Arp, D., Spreitzenbarth, M., Hubner, M., Gascon, H., and Rieck, K. In Proceedings of the 21st Annual Network and Distributed System Security Symposium (NDSS 2014), San Diego, California, USA, February 2014.

[2] **Random Forests.** Breiman, L. Machine Learning, Vol. 45, No. 1, pp. 5–32, October 2001. Springer. https://doi.org/10.1023/A:1010933404324

[3] **Scikit-learn: Machine Learning in Python.** Pedregosa, F., Varoquaux, G., Gramfort, A., et al. Journal of Machine Learning Research, Vol. 12, pp. 2825–2830, 2011.

[4] **Android: From Reversing to Decompilation.** Desnos, A., and Gueguen, G. Proceedings of Black Hat Abu Dhabi, 2011. Androguard library available at: https://github.com/androguard/androguard

[5] **Django REST Framework Web APIs for Django.** Django Software Foundation. Version 3.14 Documentation. Available at: https://www.django-rest-framework.org/ (accessed February 2026).

[6] **Django Web Framework Documentation.** Django Software Foundation. Version 4.2. Available at: https://docs.djangoproject.com/en/4.2/ (accessed February 2026).

[7] **React A JavaScript Library for Building User Interfaces.** Meta Platforms, Inc. Version 18 Documentation. Available at: https://react.dev/ (accessed February 2026).

[8] **Bitcoin: A Peer-to-Peer Electronic Cash System.** Nakamoto, S. October 2008.
Available at: https://bitcoin.org/bitcoin.pdf

[9] **Permissions on Android.** Android Developers. Android Developer Documentation. Available at: https://developer.android.com/guide/topics/permissions/overview (accessed February 2026).

[10] **OWASP Mobile Security Testing Guide (MSTG).** OWASP Foundation. Version 1.7. Available at: https://owasp.org/www-project-mobile-security-testing-guide/ (accessed February 2026).

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH

IN SCIENCE, ENGINEERING AND TECHNOLOGY

| Mobile No: +91-6381907438 | Whatsapp: +91-6381907438 | ijmrset@gmail.com |

www.ijmrset.com