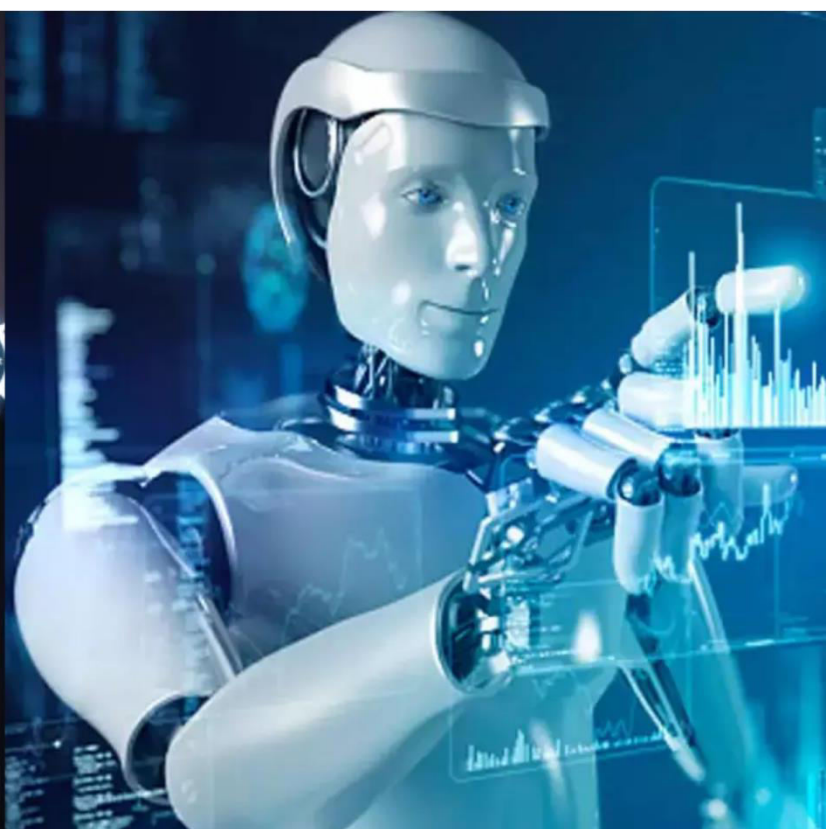




International Journal of Innovative Research in Science Engineering and Technology (IJIRSET)

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



Impact Factor: 9.935

Volume 15, Issue 10, October 2026

Intelligent Malware Report Summarizer

Akileshwari A¹, Chelladevi T², Deepika S³

Assistant Professor, Department of Computer Science and Engineering, K.L.N. College of Engineering, Pottapalayam,
Tamil Nadu, India¹

U.G. Student, Department of Computer Science and Engineering, K.L.N. College of Engineering, Pottapalayam,
Tamil Nadu, India²

U.G. Student, Department of Computer Science and Engineering, K.L.N. College of Engineering, Pottapalayam,
Tamil Nadu, India³

ABSTRACT: Malware analysis reports contain large amounts of technical information such as malware behaviours, Indicators of Compromise (IoCs), file hashes, IP addresses, URLs, domains and attack techniques. Reading these reports manually is time-consuming and needs cybersecurity expertise. This paper presents the “Intelligent Malware Report Summarizer”, a web-based application for automated malware report analysis and summarization. The user uploads a report in PDF or text format; the text is extracted and processed with Natural Language Processing (NLP) techniques. IoCs such as IP addresses, URLs, domains, file names and cryptographic hashes are extracted using regular expressions, while a TF-IDF and Random Forest classifier together with behavioural rules assigns the malware to a category such as ransomware, trojan, spyware, worm, botnet, backdoor, rootkit, keylogger, infostealer or downloader. A risk score and severity level are then computed from the identified characteristics and indicators. A Large Language Model accessed through the Groq API generates an evidence-based summary and recommended security actions, and all results are compiled into a professional incident report that can be downloaded as a PDF. The system integrates Python, Django, NLP, machine learning, SQLite, IoC extraction and AI-based summarization on a single platform and reduces the manual effort needed to review lengthy malware reports.

KEYWORDS: Malware analysis, Indicators of Compromise, Natural Language Processing, Random Forest, Large Language Model, Django, Incident report.

I. INTRODUCTION

Malware analysis reports contain large amounts of technical information such as malware behaviours, Indicators of Compromise (IoCs), network activities, file hashes, vulnerabilities, attack techniques and security observations. Manually reading and analysing these reports is time-consuming, especially when they run to many pages of technical detail.

The Intelligent Malware Report Summarizer is a web-based cybersecurity application developed to automate the analysis and summarization of malware analysis reports. An authorised user uploads a report in PDF format through a Django-based web interface. The textual content is extracted using PyMuPDF/PyPDF and then cleaned and processed with Natural Language Processing (NLP) techniques.

The system automatically extracts IoCs such as IP addresses, URLs, domains, file names, MD5, SHA-1 and SHA-256 hashes, CVE identifiers and MITRE ATT&CK technique identifiers using regular expressions. Malware classification is performed using machine learning and behavioural analysis, and a risk analysis module calculates a risk score and assigns a severity level of LOW, MEDIUM, HIGH or CRITICAL. A Large Language Model (LLM) accessed through the Groq API generates a concise summary and security recommendations, and the combined results are produced as a downloadable PDF incident report.

The paper is organised as follows. Section II reviews related work. Section III describes the existing system and its limitations. Section IV presents the proposed system. Section V explains the system architecture and requirements. Section VI describes the implementation modules. Section VII discusses the results, Section VIII covers system testing, and Section IX presents the conclusion and future enhancements.

II. LITERATURE SURVEY

Research on machine-learning-based PDF malware detection focuses mainly on identifying whether a PDF file is malicious or benign. Features are extracted from PDF documents and algorithms such as Random Forest are used for classification [1]. Although such work gives automated detection and effective feature extraction, it does not provide complete malware report summarization, detailed IoC extraction or recommendations.

Explainable Artificial Intelligence (XAI) research improves the transparency of machine-learning models used for malware analysis. Techniques such as feature importance, SHAP and LIME help analysts understand why a model produces a particular prediction [2]. However, many of these works are survey-oriented and do not provide an end-to-end report analysis application with summarization and incident-report generation.

Large Language Models have been investigated for malware analysis, threat intelligence, report summarization and identification of malicious behaviours [3]. They generate concise summaries and reduce manual reading effort, but their output requires validation against source evidence, they may not give reliable structured IoC extraction on their own, and they do not necessarily integrate malware classification, risk scoring and downloadable incident reports.

Machine-learning and deep-learning approaches have also been studied for malware-family classification and explainable classification [4]. In such systems classification is usually the primary objective, and report summarization, IoC extraction and professional report generation are often not integrated. The proposed system addresses these gaps by combining all of these tasks in a single workflow.

III. EXISTING SYSTEM

Existing malware-analysis approaches generally require cybersecurity analysts to manually examine lengthy reports to identify malware behaviour, network indicators, file hashes, vulnerabilities, suspicious files and attack techniques. Some systems only detect whether a PDF file is malicious or safe, while others are designed specifically for malware classification, threat-intelligence analysis or extraction of security information, and these tasks are usually handled with different tools. The main limitations are:

- Manual analysis of lengthy reports requires considerable time and effort.
- Important IoCs such as IP addresses, URLs, domains, hashes and file paths are difficult to identify in large reports.
- Classification, IoC extraction, threat analysis and summarization are handled as separate processes using different tools.
- Security recommendations require additional analysis and expert knowledge.
- Preparing a structured incident report manually requires additional effort.
- Manual processing may be inconsistent and may overlook important security indicators.
- The lack of an integrated system increases the time needed to understand and respond to a malware incident.

IV. PROPOSED SYSTEM

The proposed Intelligent Malware Report Summarizer is a web-based system that provides an integrated workflow combining document processing, NLP, IoC extraction, malware classification, risk assessment, AI-assisted summarization, security recommendations and incident report generation. It is developed using Python and Django [6, 12] for the backend, HTML, CSS, Bootstrap and JavaScript for the interface, and SQLite [13] for data storage. PDF and text reports are processed using PyMuPDF/PyPDF [5], NLP preprocessing is performed using NLTK [7], and TF-IDF and Random Forest from Scikit-learn [8] are used for classification.

A. Report Upload and Text Extraction

An authorised user logs in, opens the analysis dashboard and uploads a report in PDF or TXT format. The file is validated before processing, so users do not need to copy and paste the contents of lengthy reports. Text is then extracted from the document using PyMuPDF/PyPDF; this text is the input for all subsequent NLP, IoC extraction, classification and summarization modules.

B. NLP Preprocessing

The extracted text is cleaned and normalised using text cleaning, lowercase conversion, tokenization, stop-word removal and removal of unnecessary characters and spaces. The processed text is used for feature generation and classification, whereas the original extracted text is preserved for IoC extraction, because aggressive preprocessing can modify structured indicators such as URLs, hashes, IP addresses and file paths.

C. IoC Extraction

Regular-expression pattern matching is applied to the original text to extract IP addresses, URLs, domains, file names, file paths, MD5, SHA-1 and SHA-256 hashes, CVE identifiers and MITRE ATT&CK IDs [9]. The indicators are organised into network, file, hash and threat-intelligence categories, and duplicate or invalid values are filtered, so the analyst can review them without searching the whole report.

D. Malware Classification

The processed text is converted into numerical features using TF-IDF and given to a Random Forest classifier. The supported categories are ransomware, trojan, spyware, worm, botnet, backdoor, rootkit, keylogger, infostealer, downloader and no malware detected. Behavioural rules improve the interpretation of specific activities; for example, file encryption suggests ransomware-like activity, while self-propagation and network scanning suggest worm-like behaviour. The module also returns a confidence value, which represents the classifier's estimate and not absolute proof of the predicted category.

E. Risk and Severity Assessment

A risk score between 0 and 100 is calculated from the malware category, the number and type of IoCs and the observed behaviours such as suspicious network activity, credential-related activity, file modification or encryption, persistence mechanisms and command-and-control activity. The score is mapped to a severity level as shown in Table I, which helps analysts prioritise incidents.

TABLE.I. RISK SCORE AND SEVERITY LEVEL

Risk Score	Severity	Meaning
0–24	LOW	Limited or low-impact suspicious activity
25–49	MEDIUM	Potentially harmful activity requiring investigation and monitoring
50–74	HIGH	Significant malicious activity that may affect systems or data
75–100	CRITICAL	Highly dangerous activity requiring immediate response and containment

F. AI-Based Summarization and Security Recommendations

After extraction, classification and risk assessment, an LLM accessed through the Groq API [11] generates a structured summary containing the malware overview, observed behaviour, important IoCs, threat assessment and recommended actions. The summary is based only on the evidence available in the uploaded report, and unsupported findings are not presented as confirmed facts. Defensive recommendations include isolating affected systems, blocking malicious IP addresses and domains, resetting potentially compromised credentials, removing malicious files, reviewing persistence mechanisms, monitoring network activity and applying relevant security patches.

G. Incident Report Generation

The classification, confidence, risk score, severity, extracted IoCs, observed behaviour, AI-generated summary and recommendations are combined into a structured malware incident report. The report is generated as a downloadable PDF using ReportLab [10] and can be used for documentation, investigation, security review and future reference.

V. SYSTEM ARCHITECTURE AND REQUIREMENTS

A. System Architecture

The system follows a web-based layered architecture. The user interacts through the web interface, while the Django application manages authentication, file uploads, application logic and communication between modules. The uploaded report passes through text extraction and NLP preprocessing; the processed text is then given to the IoC extraction and malware classification modules. Their outputs are used by the risk and severity module and, together with the report

information, by the Groq-based LLM module. The final results are stored in the SQLite database and compiled into a PDF report using ReportLab (Fig. 1).

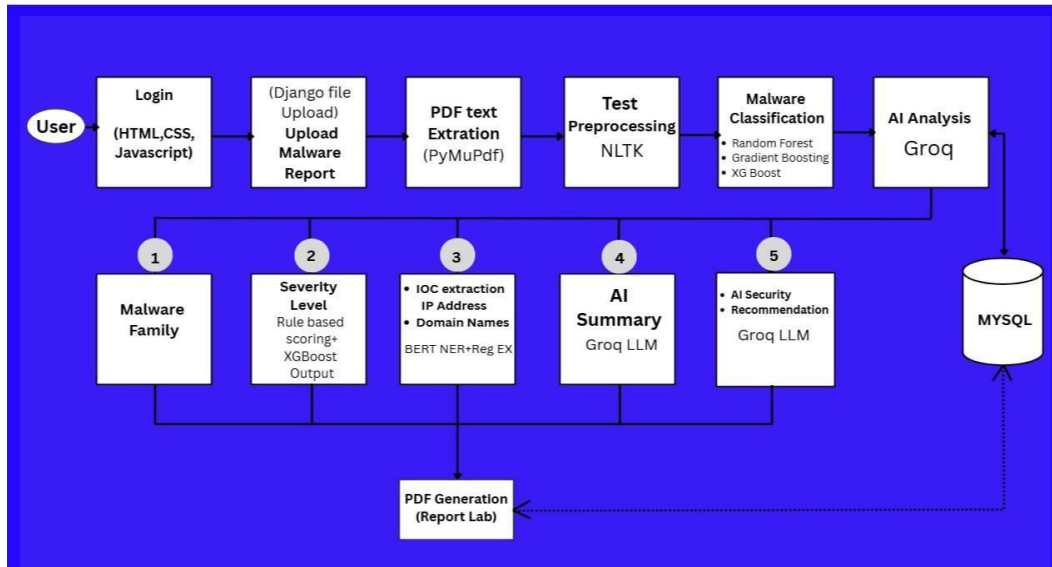


Fig.1. System Architecture

B. Data Flow

The data flow begins when the authorised user logs in and uploads a report. The extracted text is cleaned by the NLP preprocessing module and used by multiple analysis modules: IoC extraction, malware classification with behavioural rules, risk assessment, LLM summarization and recommendation generation. The processed information is combined, presented through the dashboard and generated as a PDF incident report (Fig. 2).

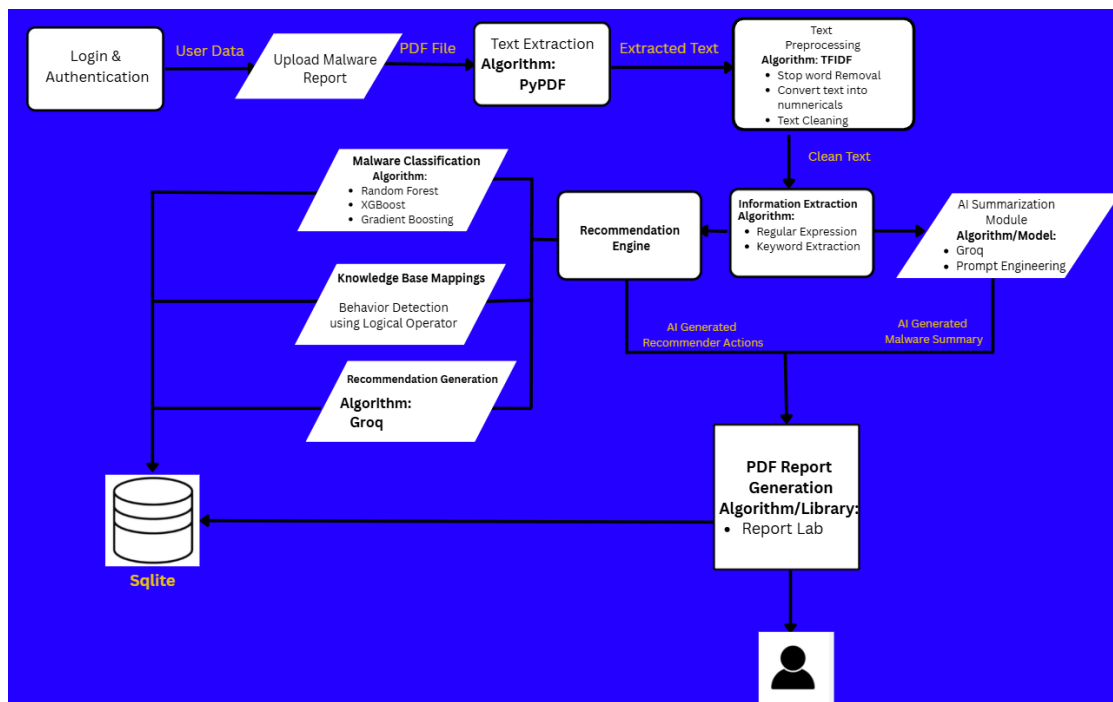


Fig.2. Data Flow Diagram

C. Hardware and Software Requirements

The system requires an Intel Core i5 or equivalent processor, 8 GB or more RAM, 10 GB or more free storage, a laptop or desktop computer, a stable internet connection for Groq API access and a display with a minimum resolution of 1366×768 . The software components are listed in Table II.

TABLE.II. SOFTWARE REQUIREMENTS

Component	Technology
Operating System	Windows / Linux
Programming Language	Python
Frontend	HTML, CSS, JavaScript, Bootstrap
Backend	Django
Database	SQLite
PDF Processing	PyMuPDF / PyPDF
NLP	NLTK
NER	BERT-based NER
Machine Learning	Scikit-learn
Classification	TF-IDF + Random Forest + Behavioural Rules
LLM	Groq API
IoC Extraction	Regular Expressions
PDF Generation	ReportLab
Development Tool	Visual Studio Code

VI. SYSTEM IMPLEMENTATION

The system is implemented as ten interconnected modules that follow the sequence: User Authentication → Report Upload → Text Extraction → NLP Preprocessing → IoC Extraction → Malware Classification → Risk and Severity Assessment → AI Summarization → Security Recommendations → Incident Report Generation → Downloadable PDF.

- **User Authentication Module:** uses the built-in Django authentication for credential validation, session management and logout, and redirects unauthenticated users to the login page.
- **Malware Report Upload Module:** provides a file-upload interface for PDF and TXT files, validates and stores the uploaded report, and displays a message when an invalid file is submitted.
- **Text Extraction and NLP Preprocessing Module:** extracts text using PyMuPDF/PyPDF (or reads TXT directly) and prepares a cleaned copy for classification while preserving the original text for IoC extraction.
- **IoC Extraction Module:** uses regular expressions to identify network, file, hash and threat-intelligence indicators and displays them in separate categories.
- **Malware Classification Module:** applies TF-IDF feature extraction, the Random Forest classifier and behavioural rules to predict the malware type and confidence.
- **Risk and Severity Assessment Module:** calculates the 0–100 risk score and assigns the severity level. For synthetic, simulated or educational reports, it distinguishes simulated evidence from confirmed real-world compromise.
- **AI Summarization and Recommendation Module:** sends the relevant report information to the Groq-based LLM to produce an evidence-based summary and defensive recommendations.
- **Incident Report Generation Module:** combines all results into an “AI Malware Incident Report” PDF that the authenticated user can download.

VII. RESULTS AND DISCUSSION

The developed system provides an integrated web-based interface for analysing malware reports. After login, a report is uploaded and the system produces structured results. A sample output is: malware type – Ransomware, confidence – 95%, risk score – 30/100 and severity – MEDIUM. These are sample values and vary with the contents and evidence of the uploaded report. The classification and confidence are generated by the classification module, whereas the risk score and severity are determined from the available security evidence.

The extracted IoCs (IP addresses, URLs, domains, file names, file paths, MD5, SHA-1 and SHA-256 hashes, CVE identifiers and MITRE ATT&CK identifiers) are displayed in categorised form, which reduces the need to search manually through lengthy reports. The predicted category and confidence are not treated as absolute proof and should be interpreted together with the extracted IoCs and behavioural evidence. The AI-generated summary and recommendations help the analyst understand the threat and the possible next steps, and the final PDF provides a professional record for documentation and further investigation.

The system can also process synthetic, simulated or educational reports created for testing. When a report identifies itself as synthetic, test indicators such as sample IP addresses, domains, hashes or simulated attacks are distinguished from confirmed real-world compromise, which prevents them from being presented as evidence of an actual incident.

VIII. SYSTEM TESTING

The system was tested at five levels. Unit testing verified individual modules, functional testing executed the complete workflow from login to PDF download, integration testing verified data flow between components (Django–SQLite, extraction–NLP, NLP–classification, Django–Groq API, results–ReportLab), performance testing observed processing behaviour for reports of different sizes, and usability testing confirmed that users can navigate the interface easily. Representative unit test cases are given in Table III.

TABLE.III. SAMPLE UNIT TEST CASES

Module	Test Performed	Expected Result
Login	Valid / invalid credentials	User is authenticated and redirected to the dashboard / access is denied with an error message
Upload	Valid PDF and TXT file	File is accepted and passed for analysis
Upload	Unsupported file	File is rejected with an appropriate message
Text Extraction	PDF text extraction	Text is extracted correctly from the report
IoC Extraction	IP, URL/domain, hash, CVE/ATT&CK	Relevant indicators are identified correctly
Classification	Malware report analysis	Appropriate category and confidence are displayed
Risk Analysis	Risk calculation	Score on a 0–100 scale; LOW to CRITICAL severity assigned
AI Summary	Valid report content	Summary is generated through the Groq API
PDF Generation	Completed analysis and download	Incident report PDF is generated and downloaded

IX. CONCLUSION AND FUTURE ENHANCEMENT

The Intelligent Malware Report Summarizer provides an integrated and automated approach for analysing lengthy malware analysis reports. By combining NLP preprocessing, Regex-based IoC extraction, TF-IDF and Random Forest classification, rule-based risk assessment and LLM-based summarization through the Groq API, it presents the important findings in a structured format and generates a downloadable incident report. The system is intended to assist analysts rather than replace expert judgment, and it can significantly reduce repetitive manual work in malware report analysis.

Future enhancements include multilingual report support, real-time threat-intelligence enrichment of IoCs, automated MITRE ATT&CK mapping of detected behaviours, training with larger and better-balanced malware datasets, an interactive dashboard with charts, cloud deployment, real-time alerts for critical indicators, integration of more advanced LLMs, historical analysis and report comparison, and authorised automated incident-response workflows.

REFERENCES

1. G. M. S. Hossain, K. Deb, H. Janicke, and I. H. Sarker, "PDF Malware Detection: Toward Machine Learning Modeling With Explainability Analysis," IEEE Access, 2024.
2. H. Manthana, S. Shajarian, J. C. Kimmell, and M. Abdelsalam, "Explainable Artificial Intelligence (XAI) for Malware Analysis: A Survey of Techniques, Applications, and Open Challenges," 2025.
3. "Large Language Models for Automated Malware Analysis and Threat Intelligence," IEEE Access, 2024.
4. "Explainable AI-Based Malware Classification Using Deep Learning," IEEE Transactions on Information Forensics and Security, 2025.
5. PyMuPDF Documentation, "PyMuPDF – Python Bindings for MuPDF," Python PDF processing and text extraction documentation.
6. Django Documentation, "Django Web Framework Documentation," Python-based web application development framework.
7. NLTK Documentation, "Natural Language Toolkit," documentation for natural language processing and text preprocessing.
8. Scikit-learn Documentation, "Machine Learning in Python," documentation for TF-IDF, classification, Random Forest and machine-learning techniques.
9. MITRE ATT&CK, "MITRE ATT&CK Knowledge Base," cybersecurity framework for documenting adversary tactics and techniques.
10. ReportLab Documentation, "ReportLab PDF Library," documentation for programmatic PDF and incident-report generation.
11. Groq Documentation, "Groq API Documentation," documentation for integrating large language models into applications.
12. Python Software Foundation, "Python Documentation," documentation for the Python programming language and standard libraries.
13. SQLite Documentation, "SQLite Database Engine," documentation for lightweight relational database management.
14. Google Research, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," research work on transformer-based language representation and natural language processing.
15. OWASP Foundation, "OWASP Application Security Verification Standard," security guidelines relevant to web application development and security practices.



INTERNATIONAL
STANDARD
SERIAL
NUMBER
INDIA



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

 9940 572 462  6381 907 438  ijirset@gmail.com



www.ijirset.com

Scan to save the contact details