

# Development of Opaline Attachment Defence System for Proactive Detection and Sanitization of Malicious Email Files

C. Magishashini<sup>\*1</sup>, S. Prabu<sup>\*2</sup>, P. Ravivarma<sup>\*3</sup>, A. Akash<sup>\*4</sup>

<sup>1</sup>Assistant Professor, Department of Information Technology, PSV College of Engineering and Technology, Krishnagiri, Tamil Nadu, India.

<sup>2,3,4</sup>UG Scholars, Department of Information Technology, PSV College of Engineering and Technology, Krishnagiri, Tamil Nadu, India.

**To Cite this Article:** C. Magishashini<sup>\*1</sup>, S. Prabu<sup>\*2</sup>, P. Ravivarma<sup>\*3</sup>, A. Akash<sup>\*4</sup>, "Development of Opaline Attachment Defence System for Proactive Detection and Sanitization of Malicious Email Files", Indian Journal of Computer Science and Technology, Volume 05, Issue 01 (January-April 2026), PP: 350-354.



Copyright: ©2026 This is an open access journal, and articles are distributed under the terms of the [Creative Commons Attribution License](#); Which Permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

**Abstract:** E-mail remains a primary mode of digital communication, and attachments play a crucial role in transmitting documents, reports, and media files. However, these attachments often serve as an entry point for cyberattacks, where malicious PDFs, Office documents, or scripts exploit hidden vulnerabilities the moment a user opens them.

Traditional defenses depend heavily on signature-based malware scanners and user awareness training, which are increasingly inadequate against zero-day exploits, polymorphic malware, and sophisticated spear-phishing attachments. As attackers continue to embed harmful payloads within seemingly legitimate documents, there is a pressing need for a more intelligent and proactive defense mechanism. To address this challenge, this project introduces Opaline Attachment, an advanced attachment-screening system powered by the DistilBERT deep learning model.

DistilBERT analyzes both structural features and embedded textual semantics from incoming email attachments to detect suspicious patterns that may indicate hidden threats. Instead of relying solely on detection, the system incorporates an additional protective layer: potentially harmful attachments are isolated in a secured sandbox and transformed into safe, static PNG renderings. This rendering process removes executable components and embedded scripts, allowing users to preview the content without interacting with the original file.

The system automatically substitutes the user's attachment view with its sanitized PNG version and provides cautionary alerts before granting access to the original file. By combining intelligent threat classification, secure content rendering, and controlled access, Opaline Attachment ensures robust protection against both known and unknown attachment-based attacks. This approach significantly reduces user exposure to harmful files and strengthens overall email security without disrupting normal workflow.

## I. INTRODUCTION

An email attachment is a file sent along with an email message, typically containing documents, images, videos, or software programs. These attachments can be in various formats, such as PDFs, Word documents, JPEG images, or ZIP files. While email attachments are widely used for legitimate purposes, they can also serve as a vector for cyberattacks, making them a significant security concern. Attackers often exploit email attachments to deliver malware, steal sensitive information, or gain unauthorized access to systems.

Email attachments are a critical component of digital communication but have become a major cybersecurity risk due to their potential to carry malicious payloads. Attackers exploit vulnerabilities in commonly used document formats such as PDFs, Word files, and Excel spreadsheets to execute malware, ransomware, or phishing attacks. However, these approaches have notable limitations:

DistilBERT is a state-of-the-art, lightweight natural language processing (NLP) model developed by Hugging Face as a more efficient alternative to the original BERT (Bidirectional Encoder Representations from Transformers). It is designed to be **40% smaller** and **60% faster** than its predecessor while remarkably retaining approximately **97% of BERT's performance** on key language benchmarks. This efficiency is achieved through a process called **knowledge distillation**, where a smaller "student" model is trained to mimic the behavior and output distributions of the larger "teacher" BERT model, learning from its "dark knowledge" rather than just hard labels.

The scope of the project extends to enhancing email security by mitigating threats posed by malicious attachments. The system leverages DistilBERT for intelligent detection of suspicious files based on content analysis and metadata evaluation. Additionally, it employs a sandboxed environment to convert potentially harmful attachments into static PNG images, ensuring secure previews without executing malicious code.

## Literature Review

Malicious email attachments remain one of the primary vectors for malware delivery, phishing attacks, and ransomware infiltration. Traditional email security mechanisms such as spam filters and signature-based antivirus systems are increasingly insufficient against evolving threats.

Existing research highlights that email filtering systems rely on **multi-layered detection techniques**, including keyword analysis, sender reputation scoring, and machine learning-based classification. These systems evaluate email content, metadata, and attachments to identify suspicious behavior. However, attackers continuously bypass these filters using obfuscation, polymorphic malware, and zero-day exploits.

Several studies propose **machine learning-based detection models** for malicious attachments. For instance, models trained on large datasets (e.g., VirusTotal) using deep learning and gradient boosting have achieved detection accuracy above 99% AUC for document and archive-based malware. Despite high accuracy, these systems often face challenges such as high false positives and computational overhead.

Another approach focuses on **behavioral and statistical modeling**, where email transmission patterns and attachment propagation behaviors are analyzed. Techniques like Naïve Bayes classification and anomaly detection compare current email behavior with historical baselines to detect malicious activity.

Additionally, **attachment sanitization techniques** such as Content Disarm and Reconstruction (CDR) convert files into safe formats (e.g., Office to PDF), effectively neutralizing embedded threats. While effective, these methods may degrade file functionality.

## II.METHODOLOGY

### Software Testing

Software testing is a crucial process that ensures the Opaline Attachment E-Mail Server Web App functions effectively and securely. The testing phase begins early in development and continues until deployment, allowing the identification and resolution of potential defects.

### Software Testing Process

The software testing process follows a structured four-step approach to ensure thorough evaluation and reliability. The first step is planning, where the testing scope is defined, key functionalities are identified, and test cases are outlined. The second step, Preparing, involves setting up the test environment, selecting test tools, and designing test cases. The third step, Executing, includes running test cases, recording results, and identifying defects. Finally, the Reporting stage involves documenting test outcomes, analyzing issues, and providing feedback for necessary improvements.

#### • Functional Testing

Functional testing ensures that the system operates according to its intended specifications. Unit testing is conducted on individual components such as email parsing and malware detection modules to verify their correctness. Integration Testing ensures smooth communication between different modules, such as attachment scanning and security alerts.

#### • Non-Functional Testing

Non-functional testing assesses the system's performance, security, and usability. Security Testing is performed to detect vulnerabilities and prevent cyber threats like malware and phishing attacks. Performance Testing evaluates system responsiveness and stability under different loads.

### Test Case

#### Authentication & User Management

##### 1. Test Case ID: TC001

- **Input:** Admin attempts to log in with valid credentials.
- **Expected Result:** Successful login, granting access to administrative functionalities.
- **Actual Result:** Admin successfully logged in, access granted.
- **Status:** Pass

##### 2. Test Case ID: TC002

- **Input:** User attempts to log in with incorrect credentials.
- **Expected Result:** Login fails, displaying an error message.
- **Actual Result:** Error message displayed, login unsuccessful.
- **Status:** Pass

##### 3. Test Case ID: TC003

- **Input:** New user registers with valid details.
- **Expected Result:** Account successfully created, user can log in.
- **Actual Result:** User account created, login successful.
- **Status:** Pass

##### 4. Test Case ID: TC004

- **Input:** User tries to register with an existing email ID.
- **Expected Result:** Registration fails, system shows "Email already exists" error.
- **Actual Result:** Error message displayed, registration blocked.
- **Status:** Pass

### Model Training

#### 5. Test Case ID: TC005

- **Input:** Admin uploads a diverse dataset for training the CANet model.
- **Expected Result:** Dataset uploaded successfully, ready for model training.
- **Actual Result:** Dataset uploaded successfully, system ready for training.
- **Status:** Pass

#### 6. Test Case ID: TC006

- **Input:** Admin starts the CANet model training process.
- **Expected Result:** Model training starts and completes without errors.
- **Actual Result:** Model trained successfully with logs generated.
- **Status:** Pass

#### 7. Test Case ID: TC007

- **Input:** Admin uploads a corrupted dataset file.
- **Expected Result:** System rejects the file and displays an error message.
- **Actual Result:** System detected corruption and rejected the file.
- **Status:** Pass

### Malicious E-Mail Attachment Detection

#### 8. Test Case ID: TC008

- **Input:** User receives an email with a malicious attachment.
- **Expected Result:** System detects and flags the attachment as malicious.
- **Actual Result:** Attachment correctly flagged as malicious.
- **Status:** Pass

#### 9. Test Case ID: TC009

- **Input:** User receives an email with a safe attachment.
- **Expected Result:** System processes the email without flagging the attachment.
- **Actual Result:** Attachment remains unflagged.
- **Status:** Pass

#### 10. Test Case ID: TC010

- **Input:** Email with a malicious attachment is opened by the user.
- **Expected Result:** System blocks direct opening and provides a warning.
- **Actual Result:** Warning displayed, preventing direct access.
- **Status:** Pass

### Attachment Conversion & Security Handling

#### 11. Test Case ID: TC011

- **Input:** Malicious email attachment is processed by the system.
- **Expected Result:** System converts the attachment into a static image format.
- **Actual Result:** Attachment successfully converted to a PNG image.
- **Status:** Pass

## 11.MODELING AND ANALYSIS

The Opaline Attachment E-Mail Server Web App is an advanced email security solution designed to detect and neutralize malicious attachments in emails. This system leverages machine learning and natural language processing (NLP) techniques to analyze email attachments in real-time, ensuring user protection against malware, phishing attempts, and document-based exploits.

The project is developed using Python as the core backend technology, with React JS for an interactive and user-friendly front-end. MySQL is used for storing user data, email metadata, and processed attachments, while WampServer facilitates local development and testing. A key feature of this system is the CANet model, built using DistilBERT, which classifies attachments as either safe or malicious based on extracted features.

### 1. Opaline Attachment E-Mail Server Web App

The Opaline Attachment E-Mail Server Web App is an advanced email security solution designed to detect and neutralize malicious attachments. The system is built using Python as the core backend technology, leveraging its powerful data processing and machine learning capabilities to enhance security measures.

### 2. System User Roles and Operations

**The system is designed to cater to two primary user roles, each with specific functionalities:**

## 2.1. Admin

**The Admin holds full control over the system and is responsible for:**

- **Login Authentication:** Securing system access through user authentication mechanisms.
- **Dataset Management:** Uploading and managing email attachment datasets used for model training.
- **CANet Model Training:** Building, training, and fine-tuning the **CANet** model to detect malicious attachments effectively.
- **User Account Management:** Handling user registration, login authentication, and setting user permissions.

## 2.2. E-Mail Account Holder/User

**The end-users (email account holders) interact with the system to:**

- **Register and log in** to access the email attachment security features.
- **Configure their email accounts** for real-time attachment monitoring.
- **View converted attachments** directly in their email client to ensure safe access.
- **Receive alerts and warnings** about potentially malicious email attachments before opening them.

## 3. Canet Model: Build And Train

The **CANet model** is at the core of the system, using advanced **machine learning techniques** to detect malicious attachments in emails. The training process involves the following steps:

### 3.1. Import Dataset

The system begins by importing a dataset containing email attachments and metadata, ensuring a diverse mix of both safe and malicious email content. This dataset includes legitimate email attachments such as standard documents, spreadsheets, and images that do not pose any security threats.

### 3.2. Preprocessing

To enhance analytical accuracy, the system applies Natural Language Processing (NLP) techniques to refine email content and prepare it for analysis.

### 3.3. Feature Extraction

The system utilizes Term Frequency-Inverse Document Frequency (TF-IDF) to extract significant textual patterns from email attachments.

### 3.4. Classification

To ensure precise identification of malicious attachments, the system employs DistilBERT a state-of-the-art deep learning model specialized in Natural Language Processing.

### 3.5. Build and Train Model

The CANet model undergoes a rigorous training process using the preprocessed datasets and extracted features.

### 3.6. Deploy Model

Once the CANet model is trained and validated, it is integrated into the Opaline Attachment E-Mail Server Web App for real-time detection of malicious email attachments.

## 4. Malicious E-Mail Attachment Detection

Once an email is received, the system analyzes its attachments in real-time to detect potential security threats. The process includes:

### 4.1. Email Reception

The system begins by processing incoming emails, automatically scanning their attachments for any signs of suspicious or potentially harmful content.

### 4.2. Attachment Processing

Once an email is received, its attachments are extracted and subjected to a detailed analysis to detect potentially harmful elements.

### 4.3. Malicious Content Detection

The CANet model then processes the extracted data to detect harmful content. Using advanced machine learning techniques, it examines the behavioral patterns within the attachment, assessing whether the file exhibits characteristics commonly associated with malware or phishing attacks.

### 4.4. Flagging Attachments

If an attachment is classified as malicious, the system takes immediate action to prevent potential damage. The email is flagged as a security threat in the user's inbox, ensuring that the recipient is alerted to the risk. Additionally, access to the original file is restricted to prevent accidental execution of harmful code.

**5. Attachment Conversion**

To ensure the safety of email recipients, the system will convert potentially malicious attachments into static images. The system will automatically convert suspicious email attachments (such as PDFs, Word documents, etc.) into static PNG images, which will neutralize any harmful code or active elements within the file.

**6. Security Handling**

Security is a key aspect of the system, providing users with alerts and warnings about potential threats. When a malicious attachment is detected, the system will issue a warning to the user, notifying them of the risks associated with opening the original file. Users will be informed about the security status of attachments before they decide to interact with them, reducing the risk of compromising their system.

**IV.RESULTS AND DISCUSSION**

**Evaluation Metrics**

- Accuracy
- Precision
- Recall
- False Positive Rate
- Detection Time

**Expected Results (based on similar research)**

- Detection Accuracy: **95–99%**
- False Positive Rate: **< 3%**
- Processing Time: **< 2 seconds per email**
- Improved detection vs traditional filters

Machine learning-based systems have demonstrated high accuracy in identifying malicious attachments and outperform traditional rule-based filters.

**Comparison (What you must include in paper)**

System Type	Detection Rate	False Positives	Response
Traditional Spam Filter	Medium	High	Reactive
Antivirus	Medium	Low	Reactive
Proposed Opaline System	High	Low	Proactive

**V.CONCLUSION**

In conclusion, the project is designed with a security-first approach, leveraging Python with Flask for Back-End processing and React JS for a dynamic Front-End interface. The integration of MySQL ensures structured and efficient data management, handling user accounts, email metadata, and processed attachments. The system's core functionalities, including CANet model-based malicious attachment detection, NLP-powered preprocessing, TF-IDF feature extraction, and DistilBERT-based classification, enhance the security of email communication. Key modules such as Admin Management, User Authentication, Dataset Upload, Model Training, Email Processing, Attachment Analysis, Threat Detection, Alert System, and Safe Attachment Conversion work cohesively to provide a secure and efficient email experience. The attachment conversion mechanism, which transforms suspicious attachments into static PNG images, adds an extra layer of security, ensuring users can access content without the risk of executing malicious code. The real-time security alerts and warnings further empower users to make informed decisions while handling email attachments. With its intelligent threat detection, seamless user experience, and robust security measures, the project presents a comprehensive solution for safeguarding email attachments from cyber threats. It stands as a proactive cyber security tool, revolutionizing traditional email security mechanisms by integrating machine learning, NLP, and real-time alerts into a user-friendly and scalable web application.

**REFERENCES**

1. L. Gallo, D. Gentile, S. Ruggiero, A. Botta and G. Ventre, "The human factor in phishing: Collecting and analyzing user behavior when reading emails", *Comput. Secur.*, vol. 139, Apr. 2024.
2. M. A. Bouke, A. Abdullah, M. T. Abdullah, S. A. Zaid, H. E. Atigh and S. H. Alshatebi, "A lightweight machine learning-based email spam detection model using word frequency pattern", *J. Inf. Technol. Comput.*, vol. 4, no. 1, pp. 15-28, Jun. 2023.
3. M. Salb, L. Jovanovic, M. Živković, E. Tuba, A. Elsadai and N. Bačanin, "Training logistic regression model by enhanced moth flame optimizer for spam email classification", *Proc. 5th Comput. Netw. Inventive Commun. Technol. (ICCNCT)*, pp. 753-768, Oct. 2022.
4. N. H. Marza, M. E. Manaa and H. A. Lafta, "Classification of spam emails using deep learning", *Proc. 1st Babylon Int. Conf. Inf. Technol. Sci. (BICITS)*, pp. 63-68, Apr. 2021.
5. O. E. Taylor and P. S. Ezekiel, "A model to detect spam email using support vector classifier and random forest classifier", *Int. J. Comput. Sci. Math. Theory*, vol. 6, no. 1, pp. 1-11, 2020.