2024 IEEE 30th International Conference on Telecommunications (ICT) | 979-8-3503-5669-4/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICT62760.2024.10606032

Unveiling Hidden Threats in PDFs with Hybrid Malware Classification

Sh M Zeeshan Javed
National University of Science and Technology
Islamabad, Pakistan
sjaved.phdismcs@student.nust.edu.pk

Faisal Amjad
National University of Science and Technology
Islamabad, Pakistan
faisal@nust.edu.pk

***Abstract*—Antivirus developers encounter significant challenges in handling PDF files due to their susceptibility to various types of malwares. These threats span from exploiting inherent vulnerabilities in PDF readers to employing phishing tactics and injecting JavaScript and shellcodes for malicious purposes. Our methodology encompasses the extraction of a defined set of static features complemented by dynamic behavioral indicators that may manifest during attacks. We integrate established Python-based tools with our framework to facilitate the static extraction process. Sandbox is utilized to identify potential infection attempts occurring during runtime execution. Additionally, for PDF classification purposes, the VirusTotal API is employed for PDF labeling. Subsequently, several machine learning-based classifiers such as Random Forest (RF), k-Nearest Neighbors (KNN), and Support Vector Machine (SVM) are utilized. These classifiers undergo evaluation based on metrics including Accuracy, Precision, Recall, and F1-Score. Experimental analysis demonstrates that the the proposed system offers an efficacious solution for the classification of PDF malware.**

***Index Terms*—PDF Malware, Malware Behavior, Dynamic Analysis, Machine Learning**

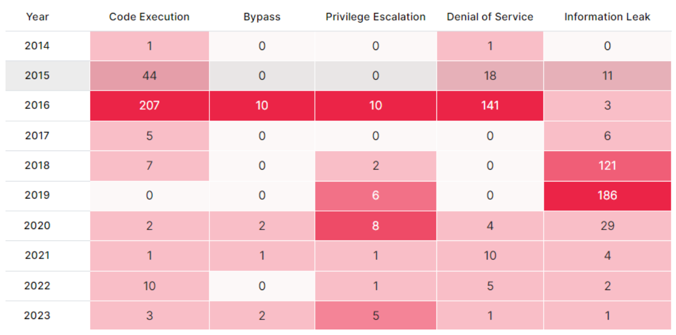
I. Introduction

Documents serve as efficient, convenient, and secure means for transmitting information. However, the widespread sharing of documents has led to a visible increase in sophisticated attacks utilizing documents harboring malicious code. Certain intentional constructs enable the execution of malicious code within a document or exploit vulnerabilities in document parsing software to breach the operating system [1].

Malicious documents often materialize in formats such as PDF, Word, and images. These formats offer the capacity to accommodate various types of content within documents, and script programs can be integrated to augment specific document functionalities. Among documented instances of malware, PDF-based attacks stand out as significant threats due to the versatile nature of PDFs in comparison to other document formats [2].

Malicious PDF documents frequently embed binary or JavaScript codes, leveraging vulnerabilities to execute malicious actions. Although JavaScript plays a crucial role in developing user-friendly interfaces with sophisticated functionalities, its adaptability also renders it a preferred language among cybercriminals for crafting malicious exploits [4]. The integration of scripting capabilities additionally enables PDF documents to operate in a manner closely resembling executables, thereby facilitating functionalities such as internet connectivity, process execution, and interaction with various files and programs. The heightened complexity of content provides attackers with an augmented array of tools to obscure malicious payloads, thus enhancing their ability to evade detection.

The widespread adoption of Adobe Acrobat Reader, coupled with its expansive attack surface, positions it as one of the primary focal points for malicious actors seeking to exploit vulnerabilities. Figure 1 illustrates the categories of impact associated with vulnerabilities identified in Adobe Acrobat Reader DC, as derived from CVEDETAILS statistics [3]. These vulnerabilities pose threats such as code execution, data disclosure, and privilege escalation. Despite concerted efforts to address various vulnerabilities in newer versions, the data underscores the persistent endeavors of attackers to exploit emergent vulnerabilities for malicious objectives.



Year	Code Execution	Bypass	Privilege Escalation	Denial of Service	Information Leak
2014	1	0	0	1	0
2015	44	0	0	18	11
2016	207	10	10	141	3
2017	5	0	0	0	6
2018	7	0	2	0	121
2019	0	0	6	0	186
2020	2	2	8	4	29
2021	1	1	1	10	4
2022	10	0	1	5	2
2023	3	2	5	1	1

Fig. 1. Adobe Acrobat Reader DC's vulnerability by year [3].

Currently, two primary methodologies dominate the landscape of PDF document analysis: static analysis and dynamic analysis. Static analysis primarily revolves around scrutinizing intrinsic, unalterable attributes within PDF documents. This approach involves the utilization of decompilation techniques to extract structural elements, content constituents, JavaScript code segments, and other pertinent features from the PDF document. These extracted features serve as the foundation for developing machine learning models aimed at identifying malicious PDF documents. In contrast, dynamic analysis employs virtualization technologies and hardware simulations. By utilizing mechanisms such as API hooks, it observes the

Authorized licensed use limited to: CMU Libraries - library.cmich.edu. Downloaded on December 05,2025 at 03:39:32 UTC from IEEE Xplore. Restrictions apply.

real-time execution behavior of malicious PDF documents and their embedded code structures. Through dynamic analysis, a behavioral model is established to identify anomalous patterns exhibited by PDF documents during execution. [4].

Previous studies have indicated potential limitations in existing tools utilized for PDF file analysis, specifically concerning incomplete extraction of malicious payloads and challenges in scrutinizing JavaScript components and shellcodes [4,5]. This paper presents a systematic approach aimed at identifying instances of malicious JavaScript and shellcode embedded within PDF documents, accompanied by an examination of their corresponding triggering actions, including open action, additional action, and launch. Structural constituents such as objects, streams, headers, and trailers have been scrutinized to recognize these elements.During the dynamic analysis of malicious PDF behaviors, we identify potential suspicious activities, such as the exploitation of vulnerabilities to compromise systems and the acquisition and execution of malware. This dynamic analysis entails the meticulous tracing of runtime operations performed by Adobe Acrobat Reader during various document processing stages, encompassing loading, parsing, rendering, and script execution. External behaviors observed include filesystem operations, network activities, registry activities, and program launches.To validate the efficacy of our proposed methodology, we conducted a comprehensive series of experiments employing a dataset comprising approximately 11,896 malicious PDF documents, sourced from publicly available repositories, namely EvasivePDF2022 [6] and MalwareBazaar [7]. Furthermore, meticulous labeling of our dataset was undertaken leveraging VirusTotal [8]. In summary, this paper contributes as follows:

- Development of the Static Malicious Feature Extractor (SMFE) approach for the identification of malicious binaries within PDF documents, along with their associated triggering actions.
- Extraction of behavioral discrepancies utilizing the Dynamic Malicious Feature Extractor (DMFE), encompassing filesystem operations, network activities, registry activities, and program launches, to enhance the capabilities of the SMFE module in classifying malicious PDF attacks.
- Emphasis on leveraging SMFE and DMFE results along with labels extracted from VirusTotal. Machine learning-based classifiers are subsequently employed to classify malicious PDF files.
- Comparative analysis of the effectiveness of SVM, RF and KNN classifier algorithm through performance evaluation metrics such as Accuracy, Precision, Recall, and F1-score.

The paper is structured as follows: Section 2 delves into related literature, providing an in-depth exploration. Section 3 outlines the methodology, detailing the structure of malicious PDFs, the tools employed, and the features utilized for model training and assessment. Section 4 presents statistical evaluations, while Section 5 concludes the findings.

II. RELATED WORK

Malicious software typically falls into two broad categories: executable programs and payloads integrated within documents. Although conventional antivirus solutions demonstrate efficacy in identifying malware within executable programs, they may encounter challenges in addressing threats concealed within documents, notably those embedded within PDF files. Even though it might seem unimportant to regular users, malware hidden within documents is risky because it's sneaky and can avoid detection.

A. Static MalPDF Detection Based on Machine Learning

Within the domain of cybersecurity enhancement, scholars have directed their efforts towards formulating robust approaches for detecting malicious documents. The investigation conducted by Xiaofeng Lu et al. [1] introduces an innovative static and rapid detection methodology tailored to various document formats. The extracted feature set encompasses object characteristics, document anomalies, code execution indications, corrupted titles, format irregularities and inaccuracies in cross-reference tables. Malicious samples were sourced from repositories such as Contagio and VirusShare, whereas benign samples were sourced from Google and Baidu. The experimental findings exhibit promising results in terms of detection accuracy and computational efficiency, employing Decision Tree and Random Forest algorithms. Min Yu et al. [5] propose an innovative methodology utilizing multi-layered abstraction to represent document characteristics, including structure, scripting language, and vector representations. Evaluation of this model involved the meticulous curation of an extensive dataset from Virus Share and Google. Comparative analysis encompassed assessing the accuracy and false positive rates across various Machine Learning classifiers.Sultan S. Alshamrani [9] introduces a novel methodology aimed at identifying PDF malware through the application of machine learning (ML) techniques. The proposed model entails an examination of API call processes within PDF files to scrutinize the activities executed during file processing. Detection hinges upon the analysis of system calls and JavaScript files integrated within the PDF. Data mining methodologies are utilized to extract pertinent information from API requests. Subsequently, a comparative evaluation is conducted employing five distinct machine learning algorithms.

B. Static MalPDF Detection Based on Deep Learning

Young-Seob Jeong et al. [2] have devised a convolutional neural network (CNN) architecture capable of processing byte sequences extracted from non-executable streams within PDF documents. To facilitate the training and validation of the model, a dataset comprising both malicious and benign PDF files was curated, with manual annotation of byte sequences within these files.

C. Feature Selection and Machine Learning Techniques

Falah et al. [10] endeavor to augment the classification efficacy of malicious PDF files by identifying the essential

Authorized licensed use limited to: CMU Libraries - library.cmich.edu. Downloaded on December 05,2025 at 03:39:32 UTC from IEEE Xplore. Restrictions apply.

feature set requisite for precise categorization. . This effort involves extracting features using static tools, then refining them meticulously to enhance the classification process. The identified features undergo thorough evaluation, with the most significant ones chosen to build a classifier using machine learning and deep learning techniques that outperform baseline models in both accuracy and computational efficiency.

D. Malicious PDF Detection using Dynamic Analysis

Meng Xu et al. [4] present a novel method for detecting malicious PDF files, leveraging platform diversity to systematically observe the behavior of such documents. By integrating with Adobe Reader, their tool monitors the internal processing and execution of PDF files, facilitating comprehensive analysis of the impact of malicious documents on host systems. Operating as a dynamic analysis tool, it conducts document-level examinations by rendering PDF files across various platforms and multiple reader versions, discerning behavioral disparities.

Issakhani et al. [11] conducted an extraction process involving 28 static features extracted from PDF files, subsequently deploying Stacking Learning techniques for detection purposes. Notably, the methodology emphasized 12 features identified as novel contributions. Evaluation of the proposed solution involved the utilization of a newly compiled dataset named Evasive-PDFMal2022. Table I presents a comparative analysis of various approaches since 2017 employed in the field of malware detection, highlighting key aspects such as the type of analysis conducted, features utilized, and the respective publication years.

TABLE I
COMPARATIVE ANALYSIS OF APPROACHES

Author	Analysis	Features	Year
Meng Xu et al [4]	Dynamic	Syscall, Memory, Format	2017
Young et al. [2]	Static	Byte sequence	2019
Min Yu et al [5]	Static	Structure, Scripts	2019
Falah et al. [10]	Static	Structure, keywords	2021
Xiaofeng et al[1]	Static	object information	2021
Sultan S. [9]	Static	API call processes	2022
Maryam et al[11]	Static	Structure, keywords, sizes	2022

III. MAIN APPROACH

The architectural overview of the system, as illustrated in Figure 2, outlines a sequential procedure. Initially, data aggregation is conducted from two primary sources, namely MalwareBazaar and Evasive-PDFMal2022. Subsequently, a dedicated module termed staticMaliciousFeatureExtractor (SMFE) has been devised to perform static analysis on malicious PDF files. This analytical process harnesses the capabilities of the PyPDF2 Python library in conjunction with pre-existing Python scripts and utilities such as PDFid [11], Pdf-parser [11], and PeePDF [12]. Further elaboration on the functionality and application of these tools is expounded upon in the subsequent section

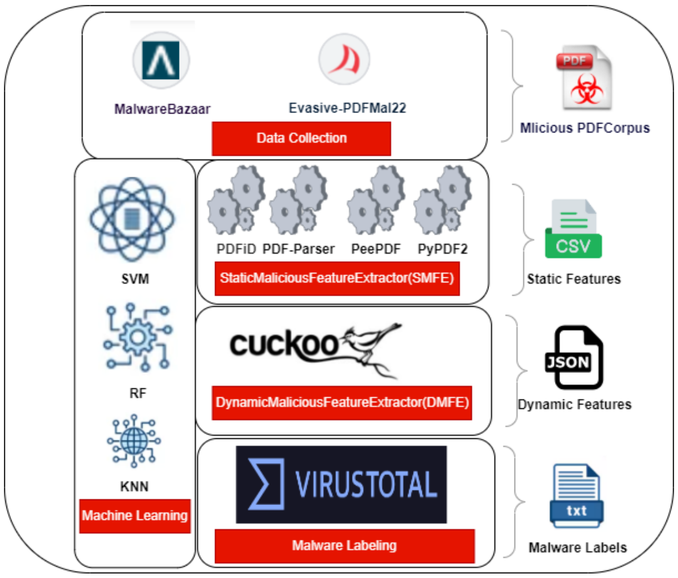


Fig. 2. Overall System Architecture

A. Data Collection

MalwareBazaar [10] serves as an online repository housing a diverse array of malware samples, fostering collaboration within the information security community, antivirus vendors, and threat intelligence providers. In this study, we obtained the comprehensive hash list comprising 771,141 malware instances up to April 2024 from MalwareBazaar. Notably, our focus was specifically on PDF files identified as malicious. To access and extract these files, we integrated the MalwareBazaar API, leveraging a registered API key obtained from the platform.Algorithm 1 depicts complete extraction process from MalwareBazaar.

Evasive-PDFMal2022 presents a dataset consisting of 10,025 records, of which 5,557 instances are categorized as malicious. Contrary to previous research endeavors [1,2,4,6,7,8] that predominantly focused on Malicious PDF Detection, our study diverges by concentrating on the classification of malicious PDF files into distinct categories. While malware detection traditionally poses a binary problem, entailing the determination of whether a file is malicious or benign, our objective extends beyond this binary paradigm to encompass multi-class classification. Therefore, we exclusively utilized the malicious subset of the Evasive-PDFMal2022 dataset for our analysis.

B. PDF Strcuture

A PDF file is composed of various discrete elements, namely the header, body, cross-reference table, and trailer:

- The header section indicates the version of the PDF file.
- The body contains the main content, including text, images, and executable code.
- The cross-reference table records the positions of all objects in the file, aiding in efficient retrieval by PDF readers.

Authorized licensed use limited to: CMU Libraries - library.cmich.edu. Downloaded on December 05,2025 at 03:39:32 UTC from IEEE Xplore. Restrictions apply.

Algorithm 1 Download Malware Hashes and Extract Malicious PDFs

```
1) Retrieve Malware Hashes:
    a) Perform a GET request to obtain the full SHA256
       hashes from https://bazaar.abuse.ch/.
2) For each Hash in Full SHA256 Hashes:
    a) Perform a POST request to https://mb-api.abuse.
       ch/api/v1/ with parameters API-KEY and hash.
    b) Extract the response field from the response.
    c) Write the response content to a ZIP file.
    d) Extract the contents of the ZIP file by specifying
       PASSWORD.
    e) If the file is PDF,
        i) Store it in the Malicious PDF Directory.
       Else,
        i) Discard the file.
```

- The trailer provides essential metadata, such as the cross-reference table's location and other key components.

The practice of injecting malicious JavaScript serves as a common strategy among malicious actors due to its capacity to exert control over the behavior of a PDF file, such as effecting redirection to malicious websites. PDF readers facilitate the execution of embedded JavaScript codes, thereby enabling the dynamic invocation of functions through JavaScript Application Programming Interfaces (APIs). Typically, these JavaScript codes are encapsulated within a stream, constituting one of the PDF's object types. Streams represent sequences of binary data bytes of varying length, primarily designed to accommodate extensive image files or elements related to page composition. A graphical representation elucidating the structure of PDFs is provided in Figure 3.

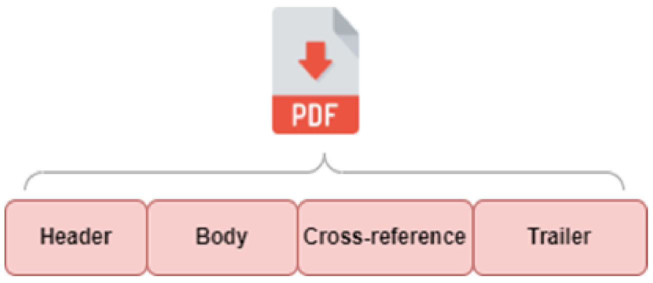


Fig. 3. PDF Structure

C. Static Malicious feature extractor (SMFE)

PDFiD tool [12] facilitates comprehensive static analysis of PDF documents. This analysis encompasses scrutiny of various structural attributes, including objects, object terminations, streams, stream terminations, cross-reference tables, trailers, and initiation markers of cross-references. Moreover, PDFiD assesses lexical elements by identifying common keywords within the document. To exemplify static feature extraction utilizing PDFiD, the malicious PDF file identified by the hash 028d311e5c1592bc4665d4d9f1744aabf429ee79be2ceb1a18d55b6935e1e5fa was subjected to analysis, with the extracted features documented in Table II.

TABLE II
COMPARATIVE ANALYSIS OF APPROACHES

PDFiD0.2.8 Fields	Malicious Sample
PDF Header:	PDF-1.5
obj	300
endobj	300
stream	218
endstream	218
xref	3
trailer	3
startxref	3
/Page	12
/Encrypt	0
/ObjStm	2
/JS	1
/JavaScript	1
/AA	1
/OpenAction	1
/AcroForm	0
/JBIG2Decode	0
/RichMedia	0
/Launch	1
/EmbeddedFile	0
/XFA	0
/URI	64
/Colors 24	0

In the extraction of malicious JavaScript embedded within PDF documents, structural elements such as objects, streams, headers, and trailers are analyzed to identify key triggering actions like /OpenAction, /Launch, and /AA. Moreover, the identification of instances of JavaScript code is facilitated through the extraction of markers such as /JS and /JavaScript. PeePDF [13] is a sophisticated tool tailored for the meticulous analysis of PDF documents. This application meticulously scrutinizes objects and streams in their original formats, delves into the intricate structures of documents, performs encoding and decoding operations on objects or variables, and simulates the execution of shellcode. PDF-Parser is equipped with the capability to parse and decompress objects, making it a valuable asset for extracting JavaScripts. An illustrative example of a stream object extracted from a malicious PDF document using PDF-Parser is presented in Table III. When the PDF document is initiated in a PDF reader, it identifies the cross-reference table and the corresponding object number associated with the "Root" attribute. As indicated in Table III, Object 1 is recognized as the Catalog, serving as the primary entry point. The "OpenAction" attribute specifies a particular region or action to execute upon document initialization. Notably, in this instance of a malicious PDF, the opening action triggers the execution of a malicious script contained within object 1293.

Authorized licensed use limited to: CMU Libraries - library.cmich.edu. Downloaded on December 05,2025 at 03:39:32 UTC from IEEE Xplore. Restrictions apply.

TABLE III
STATIC ANALYSIS FOR MALICIOUS PDF JAVASCRIPT CALL WITH ACTIONS

PDF Object	Object Value
obj 1293	Type: /Action Referencing: /Launch c:/windows/system32/cmd.exe if exist " Mis Documentos/blague.pdf (cd "Mis Documentos")) (start blague.pdf)
obj 1	Type: /Catalog Referencing: 1293 0 /OpenAction 1293 0 R
xref	/Size 1288 /Root 1 0 R /Info 275 0 R

D. Dynamic Malicious feature extractor (SMFE)

Run-time loading is a technique employed to obscure malicious activities by dynamically loading code segments during program execution, thereby complicating detection by static analysis tools. For instance, this approach may involve initiating the retrieval of a malicious payload once the document is accessed on the target system. Various obfuscation methods, such as encryption and modifications to the code structure, are utilized to obscure the true intent of the malicious source code during static analysis.In response to this challenge, our investigation focuses on monitoring the run-time operations conducted by Adobe Acrobat Reader across various stages of document processing, encompassing loading, parsing, rendering, and script execution. External behaviors, including filesystem operations, memory dumps, network activities, registry interactions, and process operations, are meticulously documented using Cuckoo Sandbox. In our experimental setup, Cuckoo Sandbox is deployed on a Kali Linux machine, functioning as a central hub for initiating malware analyses on Windows-based host machines. These host machines operate within a VirtualBox environment running Windows, and are specifically designated for the execution of suspicious PDFs and JavaScript files. Following the execution of each sample, the host machine is reverted to its snapshot state, ensuring its readiness for subsequent analyses. To expedite the analysis process for a substantial volume of PDF malware samples, we concurrently execute analyses on five host machines.

E. Virustotal Labeling

To ensure the accuracy of our model training and evaluation, we meticulously label our dataset leveraging VirusTotal. As an online service, VirusTotal facilitates file or hash analysis, providing comprehensive reports from various antivirus engines, including details on registry and file access. With the VirusTotal Public API v2.0, application developers can analyze up to 500 files per day for free. By submitting hashes or files with our VirusTotal API key, we obtain JSON objects containing antivirus engine results. Our identification of malware families relies on processing these JSON results obtained through the VirusTotal API. Presently, our dataset comprises 14,088 malicious PDF files. For instance, consider a malicious PDF instance 028d311e5c1592bc4665d4d9f1744aabf429ee79be2ceb1a18d55 b6935e1e5fa. Upon verification, 41 out of 61 VirusTotal assessments classify it as malicious. Further parsing reveals that 11 out of 41 identify it as Exploit.PDF-Dropper.Gen.

IV. STATISTICAL EVALUATION

According to the design goal of the classification system, classification Accuracy, Precision, Recall and F1-Score of classifiers are evaluated. In scenarios where all documents within the dataset are identified as malicious PDFs and there are no instances of benign PDFs, the absence of True Negatives (TN) and False Positives (FP) simplifies the accuracy metric. In such cases, accuracy can be succinctly expressed as the ratio of correctly identified malicious PDFs (True Positives, TP) to the total number of documents within the test dataset (TP + FN).

$$Accuracy = \frac{TP}{(TP + FN)} \tag{1}$$

Precision in this context measures the proportion of correctly identified malicious PDFs (True Positives, TP) to the total number of documents classified as malicious by the model (True Positives plus False Positives, TP + FP). Mathematically, precision is calculated as follows:

$$Precision = \frac{TP}{(TP + FP)} \tag{2}$$

Recall, also known as sensitivity or true positive rate, measures the proportion of correctly identified malicious PDFs (True Positives, TP) to the total number of actual malicious PDFs in the dataset (True Positives plus False Negatives, TP + FN). Mathematically, recall is calculated as follows:

$$Precision = \frac{TP}{(TP + FN)} \tag{3}$$

The F1 score is a harmonic mean of precision and recall, providing a single metric to evaluate the balance between precision and recall. It is calculated as follows:

$$F1Score = 2x\frac{Precision + Recall}{(PrecisionRecall)} \tag{4}$$

Table IV illustrates the performance of difference classifiers.

TABLE IV
STATISTICAL EVALUATION OF MACHINE LEARNING CLASSIFIERS

Model	Accuracy	Precision	Recall	F1-Score
RF	0.961	0.969	0.96	0.964
SVM	0.932	0.941	0.91	0.925
KNN	0.924	0.951	0.92	0.935

Comparative evaluation with other PDF malware classification techniques indicates that RF classifier aligns best with our proposed system.

Authorized licensed use limited to: CMU Libraries - library.cmich.edu. Downloaded on December 05,2025 at 03:39:32 UTC from IEEE Xplore. Restrictions apply.

V. CONCLUSION

In conclusion, documents play a crucial role in information dissemination due to their efficiency, convenience, and security. Malicious documents, particularly those in PDF format, pose significant threats as they can exploit vulnerabilities in document parsing software to compromise operating systems. By understanding and effectively analyzing the complexities of PDF malware, we can better mitigate the risks associated with malicious documents and enhance cybersecurity measures. However, it is worth noting a limitation of this approach, particularly concerning highly sophisticated PDF malware meticulously crafted to evade both static and dynamic detection methods. Such malware often incorporates intelligent code designed to detect sandbox analysis and may employ tactics such as delaying execution or extending analysis time to evade detection. While sandbox analysis typically involves an adjustable analysis time, commonly set to five minutes, these advanced malware variants pose challenges to traditional detection methods. In the future, we aim to extend our research to encompass malicious Word, Excel, and PowerPoint documents, as these file formats are also commonly utilized in document sharing. By expanding our analysis to include these formats, we can better understand and address the security challenges posed by various types of malicious documents. This broader scope will enable us to develop more comprehensive detection and mitigation strategies to safeguard against threats across different document types.

REFERENCES

[1] Lu, X. Wang, F. Jiang, C. Lio, P. A Universal Malicious Documents Static Detection Framework Based on Feature Generalization. Appl. Sci. 2021, 11, 12134.

[2] Young-Seob Jeong, Jiyoung Woo, Ah Reum Kang, ”Malware Detection on Byte Streams of PDF Files Using Convolutional Neural Networks”, Security and Communication Networks, vol. 2019, Article ID 8485365, 9 pages, 2019

[3] Adobe Acrobat Reader’s vulnerability by year, Available online at ”https://www.cvedetails.com/product/32069/Adobe-Acrobat-Reader-Dc.html?vendorid=53’

[4] M. Xu, T. Kim, PlatPal: Detecting malicious documents with platform diversity, in: USENIX Security Symposium, 2017, pp. 271–287

[5] M. Yu, J. Jiang, G. Li, C. Lou, Y. Liu, C. Liu, W. Huang, Malicious documents detection for business process management based on multi-layer abstract model, Future Gener. Comput. Syst. 99 (2019) 517–526.

[6] CIC-EvasivePDF2022 PDF dataset by year 2024, Available online http://205.174.165.80/CICDataset/CIC-EvasivePDF2022/Dataset/

[7] MalwareBazaar dataset available online in April 2024 at https://bazaar.abuse.ch

[8] Virustotal availabale online in April 2024 at https://www.virustotal.com

[9] Sultan S. Alshamrani, ”Design and Analysis of Machine Learning Based Technique for Malware Identification and Classification of Portable Document Format Files”, Security and Communication Networks, vol. 2022, Article ID 7611741, 10 pages, 2022.

[10] Ahmed Falah, Lei Pan, Shamsul Huda, Shiva Raj Pokhrel, Adnan Anwar, Improving malicious PDF classifier with feature engineering: A data-driven approach, Future Generation Computer Systems, Volume 115,2021,Pages 314-326, ISSN 0167-739X,

[11] Maryam Issakhani, Princy Victor, Ali Tekeoglu, and Arash Habibi Lashkari1, “PDF Malware Detection Based on Stacking Learning”, The International Conference on Information Systems Security and Privacy, February 2022

[12] D. Stevens, PDF tools, 2024, Available online at https://blog.didierstevens.com/programs/pdf-tools

[13] J.M. Esparza, peepdf - PDF analysis tool, 2024, Available online at https://eternal-todo.com/tools/peepdf-pdf-analysis-tool

Authorized licensed use limited to: CMU Libraries - library.cmich.edu. Downloaded on December 05,2025 at 03:39:32 UTC from IEEE Xplore. Restrictions apply.