DETECTION OF MALICIOUS CODE IN PDF FILES

A project report

Submitted in partial fulfilment of the requirements for the award of degree of

Bachelor of Technology

(Computer Science and Engineering)

Submitted to

LOVELY PROFESSIONAL UNIVERSITY

PHAGWARA, PUNJAB

From 10/02/2024 to 03/05/2024

SUBMITTED BY

Abhinav Kumar (12012931)
Yogesh Indoliya (12113369)

Nahita Pathania
(19372)

DETECTION OF MALICIOUS CODE IN PDF FILES

A project report

Submitted in partial fulfilment of the requirements for the award of degree of

Bachelor of Technology

(Computer Science and Engineering)

Submitted to

LOVELY PROFESSIONAL UNIVERSITY

PHAGWARA, PUNJAB

From 10/02/2024 to 03/05/2024

SUBMITTED BY

Abhinav Kumar (12012931)
Yogesh Indoliya (12113369)

Nahita Pathania
(19372)

INDIA'S LARGEST UNIVERSITY

LOVELY PROFESSIONAL UNIVERSITY

Transforming Education, Transforming India

TOPIC APPROVAL PERFORMA

School of Computer Science and Engineering (SCSE)

Program : P132-L::B.Tech. (Computer Science and Engineering) [Lateral Entry]

COURSE CODE : CSE445 **REGULAR/BACKLOG :** Regular **GROUP NUMBER :** CSERGC0445

Supervisor Name : Nahita Pathania **UID :** 19372 **Designation :** Assistant Professor

Qualification : _____ **Research Experience :** _____

SR.NO.	NAME OF STUDENT	Prov. Regd. No.	BATCH	SECTION	CONTACT NUMBER
1	Yogesh Indoliya	12113369	2021	K20NK	9808606838
2	Abhinav Kumar	12012931	2020	K20NK	7633875634
3	Umang Raj	12006480	2020	K20NK	8789639576

SPECIALIZATION AREA : Networking and Security-I **Supervisor Signature:** _____

PROPOSED TOPIC : Detection of Malicious Code in PDF files

Qualitative Assessment of Proposed Topic by PAC		
Sr.No.	**Parameter**	**Rating (out of 10)**
1	Project Novelty: Potential of the project to create new knowledge	6.67
2	Project Feasibility: Project can be timely carried out in-house with low-cost and available resources in the University by the students.	7.20
3	Project Academic Inputs: Project topic is relevant and makes extensive use of academic inputs in UG program and serves as a culminating effort for core study area of the degree program.	6.20
4	Project Supervision: Project supervisor's is technically competent to guide students, resolve any issues, and impart necessary skills.	7.53
5	Social Applicability: Project work intends to solve a practical problem.	6.87
6	Future Scope: Project has potential to become basis of future research work, publication or patent.	6.73

PAC Committee Members		
PAC Member (HOD/Chairperson) Name: Dr. Harwant Singh Arri	UID: 12975	Recommended (Y/N): Yes
PAC Member (Allied) Name: Dr. Vishu	UID: 18807	Recommended (Y/N): Yes
PAC Member 3 Name: Dr. Balraj Singh	UID: 13075	Recommended (Y/N): Yes

Final Topic Approved by PAC: **Detection of Malicious Code in PDF files**

Overall Remarks: Approved

PAC CHAIRPERSON Name: 13897::Dr. Deepak Prashar **Approval Date:** 09 Apr 2024

Declaration By Student

To whom so ever it may concern

I, **Abhinav Kumar (12012931) and Yogesh Indoliya (12113369),** hereby declare that the work done by me on "**MALICIOUS DETECTION OF PDF FILES**" under the supervision of **Nahita Pathania**, **Assistant Professor,** Lovely professional University, Phagwara, Punjab, is a record of original work for the partial fulfilment of the requirements for the award of the degree, **Computer Science and Engineering.**

Project Group Number: CSERGC0445

Name of the Student (Registration Number)

Abhinav Kumar (12012931)

Yogesh Indoliya (12113369)

Signature of the student:-

Dated:

Declaration by the Supervisor

To whom so ever it may concern

This is to certify that **Abhinav Kumar (12012931) and Yogesh Indoliya (12113369)** from Lovely Professional University, Phagwara, Punjab, has worked on "**MALICIOUS DETECTION OF PDF FILES**" under my supervision from. It is further stated that the work carried out by the student is a record of original work to the best of my knowledge for the partial fulfilment of the requirements for the award of the degree, **B.Tech.**

Name of Supervisor

UID of Supervisor

Signature of Supervisor

ACKNOWLEDGEMENT

We would like to express our special gratitude towards our teacher. **Ms. Nahita Pathania**, who gave us the golden opportunity to do this innovative project on **MALICIOUS DETECTION OF PDF FILES**, Who also helped us in completing the project. Your willingness to motivate us contributed tremendously to our project. This project helped us to do a lot of research and we came to know about so many new things. Thanks to everyone who helped us a lot in finalizing this project within the limited time frame. We want to express our gratitude to all the people who have given their heart whelming full support in making this compilation a magnificent experience. With deep sense of gratitude, we would like to thank all faculty members for their generous help in various ways for the completion of this thesis. We are extremely grateful to our faculties for their constant guidance and willingness to share their vast knowledge which made us understand this project and its manifestations in great depths and helped us to complete the assigned tasks.

We are also thankful to the H.O.D. for continuous encouragement and help. Finally, yet importantly, we would like to express my heartful thanks to my beloved parents for their blessings, my friends and classmates for their help and wishes for the successful completion of this project. Last but not the least, we thank the almighty God for reasons too numerous to mention.

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

The widespread use of digital documents, particularly PDFs, has made them a popular target for cybercriminals looking to exploit weaknesses for a variety of harmful purposes. This research digs into the difficulties of identifying malicious files encoded in PDFs and discusses the strategies, problems, and solutions for minimizing this issue. The cybersecurity system serves as the most vital system among all the systems in an organization, as it employs a network of firewalls and intrusion detection systems to transport data securely to other systems. Any threat that affects the cybersecurity system is known as a cyber-attack. Millions of people worldwide are affected by cyber-attacks each year. Cyberattacks come in many different forms, phishing, malware, ransomware, and denial of service attacks are a few examples. Cyberattack victims have a variety of symptoms, including slow computer performance, unexpected pop-ups, and frequent crashes. The leading causes of cyberattacks are weak passwords, outdated software, lack of firewalls, downloading malicious or suspicious files from compromised websites or attached emails, and other poor security practices. Invasive threat diagnosis procedures are both expensive and unpleasant. Thus, a method that can identify cyber threats non-invasively and inexpensively is required. PDFs are the most popular file type for transporting malicious files as email attachments. In the vast majority of instances, approximately 91 % of cyberattacks employing malicious code are initiated through spear-phishing emails. These attacks often leverage attachments and links laden with malicious code. This highlights the critical need for users to differentiate between these types of attacks and legitimate online content. A plethora of online tools exist that can alert users to potentially harmful websites, files, or email attachment links. However, the use of antivirus scanning tools or websites can be daunting due to their complexity and the varying levels of technical proficiency among users. To mitigate this issue, this report introduced a user-friendly web application “Malicious File Checker”, built with JavaScript, that is designed to detect malicious PDF files. This application incorporates a variety of algorithms and a cybersecurity framework to accurately determine whether an uploaded file poses a threat. The paper aims to determine an antivirus web application's work and features for giving a useful perspective toward updating cybersecurity and minimizing cyber threats. Files included with malware generally most of the time target different environments like cloud containers and traditional network traffic that will be harmful for organizations as well as for individuals. Attackers are using both vulnerabilities

that are already disclosed and ones that are not yet disclosed. Several times it has been seen that vulnerabilities were discovered years ago and have not been patched by organizations, and in other cases, the product could lack a patch because the product is at the end of its supported lifespan. Many organizations could also lack awareness of available fixes, which effectively turns an old, known vulnerability into something as risky as a zero-day threat.

Chapter 2

Review Of Literature

This report investigates cybersecurity threats and various types of attacks that pose risks to non-technical users who are not that friendly with online stuff. These individuals lack extensive knowledge of internet security practices and may inadvertently download malicious PDF files. The study emphasizes the importance of antivirus applications and solutions for identifying potentially harmful files. specifically, it addresses scenarios where users and organizations minimize or restrict themselves by clicking suspicious email links or visiting compromised websites and also maximize themselves by using the user-friendly antivirus web application Malicious File Checker, which checks the PDF files whether malicious or not and reduces the chance of unintended installation of malicious PDFs on their systems. This report introduces the work and definition of several malware or malicious activities that are now in trend and used by cyber attackers to gain unauthorized access to the victim's systems and steal important credentials or information for performing illegal activities. This report discusses the structure of PDFs with possible embedded suspicious activity and how to take precautions to make a safe environment for users as well as the organization's systems.

Most of the time when users use the internet they reach lots of websites and links that they think are safe but attackers generally use the advantages of user trust. They obfuscate malicious instructions in the form of PDF, Excel, and other files that users do not know maybe viruses and they are attracted by call-to-action logos, they download the suspicious files by mistake or forcibly by cyber attackers. after downloading a suspicious file it starts corrupting the system and gives remote access to the server hosted by cyber attackers, malicious instructions that victims install or download through legitimate sites are going to give every important function or information to their host server owned by cyber attackers they know every progress and work of the victim system and getting important credentials by keyloggers or able to control the system. that's why we introduced this simple and easy-to-understand antivirus scanning website so whenever any user contacts with a suspicious pdf or file they can just upload that suspicious file into the upload bar section and click the start link which gives the information about whether the given file is suspicious or not. A lot of research and inventions already exist on detecting malicious files, but compared to others it is easy to use and gives real-time results without wasting time. The advanced

features of PDF, such as /Launch, which can automatically run an embedded script to manage operating system-specific events, and /GoTo and /URL, which can automatically open remote resources for creating an internet risk, are also exploited by the attackers in addition to the vulnerabilities in the PDF reader. [4]. In this subject, there are two sorts of processes to detect or analyze malicious or suspicious files: dynamic analysis and static analysis. The first step is to analyze the malicious file before running it on the system, and the second step is to detect the behavior and functions or capabilities that determine how much suspicious files affect the system.

- **Cost-Benefit Analysis of Detection Solutions**
  - **Total Cost of Ownership (TCO)**

Organizations should evaluate the total cost of ownership associated with implementing and maintaining various PDF file detection solutions, considering factors such as licensing fees, hardware requirements, and ongoing operational costs.

Return on Investment (ROI)

Assessing the potential return on investment of detection solutions in terms of reduced security incidents, business continuity, and protection of sensitive data can help justify investment decisions and prioritize resource allocation. There are numerous antivirus scanning websites and tools accessible on the market that have different capabilities. hashmyfile tool generates hashes of suspicious files or compares the hashes with their database. Hex editor and CFF explorer tools provide information on types of files based on their signature (.exe, .dll, .sys). These tools are very helpful for determining whether any files are harmful or not by comparing signatures and hashes, but attackers use them again and again different approaches to bypass the security or protocols to gain unauthorized access of users systems and misuse the important information and credentials of the victim. so if attackers can progress or advance their approaches, normal users have also the right to able to know any suspicious files and use precautions, or develop the ability to use simple or easy antivirus scanning tools for detecting suspicious activity.

Chapter 3
Implementation of Project

3.1 Objective and Hypothesis:

Malware Detection: Malware detection refers to the process of identifying and mitigating malicious software (malware) on computers, networks, or devices. Malware encompasses a wide range of harmful programs designed to infiltrate, disrupt, or compromise systems. The primary goal of malware detection is to identify and prevent malware from causing harm and prevent further spread by ensuring the security and integrity of systems and data. Malware detection can be done by two methods, static analysis and dynamic malware analysis. In static malware analysis the scanning or examining of a suspicious file's code without execution by evaluating file names, hashes, and header data to detect signs of malicious intent. But in dynamic malware analysis execution of suspicious files in a sandboxed environment for observing the behavior of malicious files and their function without harming and risking the system. Effective malware detection combines expertise, technology, and multiple techniques to safeguard systems against evolving threats.

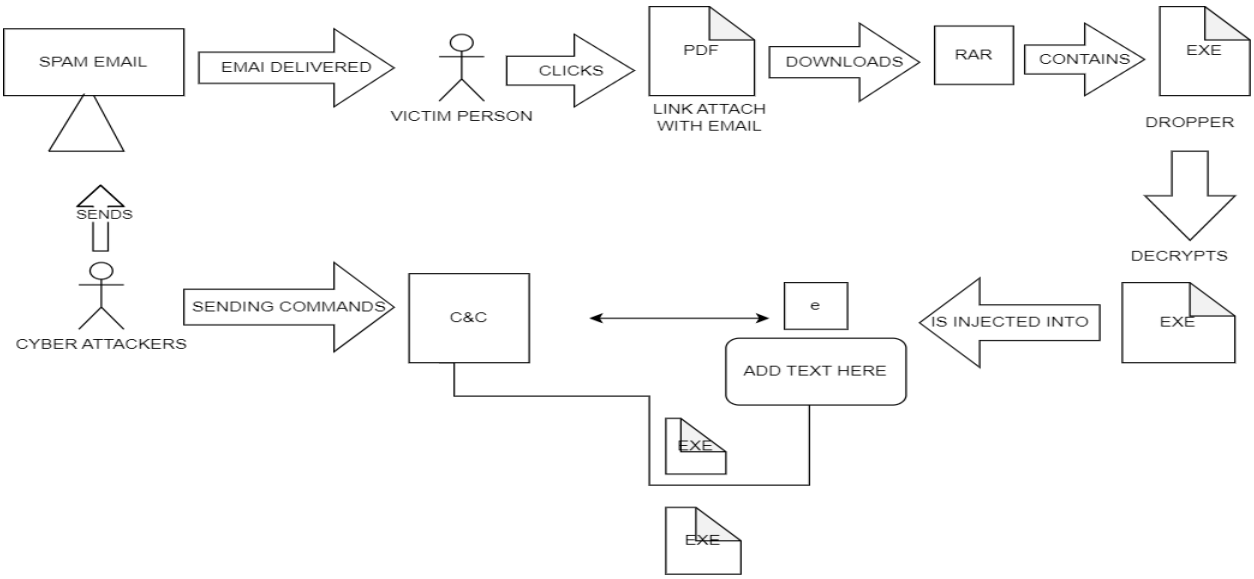


Figure 1. Flow Chart Of Malware Attacks

- Types Of Threats:
  - We have seen a boom in traditional malware techniques taking advantage of AI tools. In the present updated technological world, there are various types of threats including spear-phishing emails. Ransomware employs strong encryption algorithms (such as AES or RSA) to lock files on the victim's system. It targets various types of files, including documents, images, and databases. It enters the victim's system through various vectors like when any user unknowingly downloads infected attachments from phishing emails. Visiting any compromised websites or clicking on malicious ads. Once ransomware is activated in the victim system it ensures its survival by updating or modifying the victim system setting, creating registry entries, and establishing communication channels with command-and-control servers. It generally displays threatening messages, often accompanied by countdown timers, demanding payment. It aims to create panic and pressure the victim into paying the ransom by providing instructions on how to pay (usually in cryptocurrency like Bitcoin). It may even offer decryption keys upon payment (though this isn't guaranteed).

  - A virus is a form of malware that attaches itself to other executable files. When the infected program runs, the virus executes its code, which can be either harmless or malicious because its primary aim is to modify or delete data within the host system. It can be able to spread from the host system means, It can be able to spread from one computer to another through infected files. viruses require a host file to propagate. It remains dormant until the user interacts with the infected file also, they are very highly destructive, causing data loss or system instability. On the other hand, Worms are stand-alone malicious programs that can self-replicate and propagate independently, it doesn't need a host and replicate from one system to another system directly. Worms focus on consuming system resources (memory, bandwidth) to slow down or disrupt the system, it spread across networks, infecting multiple computers. Worms are less destructive than viruses, it can cause significant performance issues and to minimize these threats users must use antivirus and firewall tools for detection of malicious activity and control worms and viruses.

- Another threat is fileless malware it doesn't rely on standalone executable files. Instead, it leverages existing system components (such as PowerShell scripts, macros, or registry entries) to execute its payload. It operates entirely in memory, leaving no traces on the disk. This function of fileless malware makes it challenging for traditional antivirus tools to detect. It generally piggybacks on legitimate processes (PowerShell, WMI, or Windows Management Instrumentation) to execute malicious code. It uses built-in OS tools and scripts, making it harder to distinguish from legitimate activity and it avoids leaving artifacts that investigators can analyze. It can steal user's sensitive information (credentials, financial data), and it also moves laterally within the network, infecting other systems. Fileless malware poses unique challenges for defenders and the required behavior of detection is based on or depends on monitoring, memory analysis, and proactive security measures.
- Another threat in the cyber world is spyware, it silently monitors user activity, capturing sensitive information without the user's awareness means it operates discreetly. It captures everything of the victim system including passwords and messages, records of visited websites, and search queries, takes periodic screenshots of the user's or victim's screen, and sends collected data from a victim machine or system to a remote server for analysis or exploitation. Spyware remains active even after the system reboots. To avoid spyware attacks users have to regularly scan the system, caution browsing, and take the help of updated software to mitigate the impact of spyware.
- Adware is also a major attack in the cybersecurity world whose function is generating intrusive pop-up ads while users browse websites or use applications. Some disguise itself as browser extensions or plugins, injecting ads into web pages to gain unauthorized access to victim's machines. It consumes system resources (CPU, memory, and network bandwidth), slowing down the device.
- Trojans operate silently in the background without the user's knowledge. It performs a range of harmful activities by creating hidden backdoors, allowing remote attackers to access the compromised system. Trojans steal sensitive information and transmit it to the malicious servers and also some trojans can modify or delete files, disrupt system functionality, or corrupt data. It can able to turn an infected system into part of a botnet for

launching coordinated attacks. This type of malware masquerades as legitimate or desirable software, enticing users to install it, and it often hides within seemingly harmless files, such as software installers, games, or multimedia content.

Types of Malicious PDF Files

Malicious PDFs can take different forms, including:

- PDFs that contain embedded malware
- PDFs with dangerous links or JavaScript.
- PDFs containing attack code for vulnerabilities

PDF (Portable Document Format) is a popular file format for transferring documents across multiple systems. However, its adaptability exposes it to exploitation by unscrupulous actors that use PDF vulnerabilities to deploy malware, phishing scams, and other cyber threats.

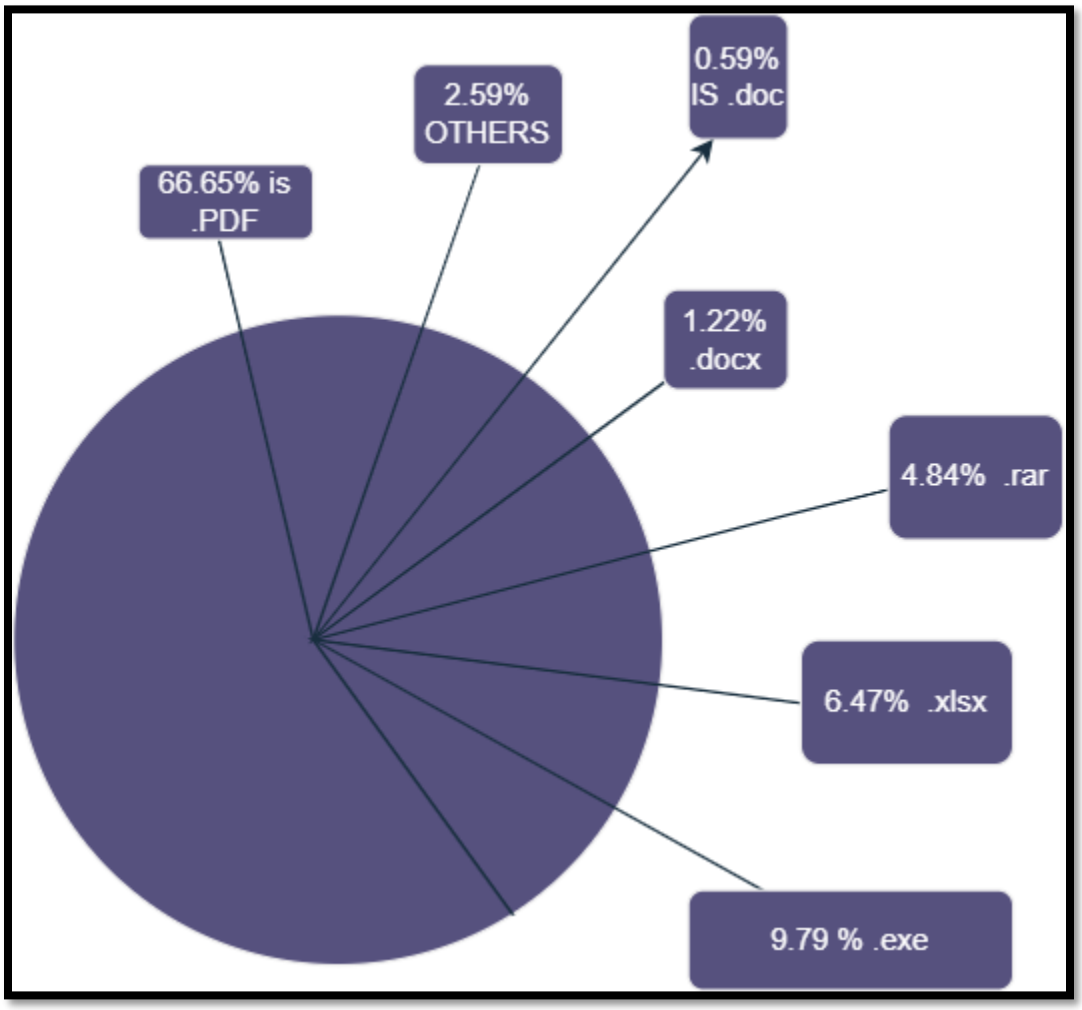


FIGURE 2. Malicious Email Attachment File Type

Figure 1 shows that even though executables are the file type of choice for attackers to inject malware, attackers are more likely to deliver malicious PDF files in email attachments. PDF accounts for approximately 66 % of all malicious email attachments. Users or victims are less suspicious of expected file types like PDFs, especially in a business context. Users may be unaware that PDF files can be used for nefarious purposes. Email remains a popular method for delivering malware. Attackers combine email attacks with social engineering tactics to increase their success rate.

Challenges in Detecting Malicious PDFs

Polymorphic malware can modify its appearance to avoid detection by typical signature-based antivirus systems, creating substantial difficulty for PDF file detection. Malicious actors use evasion strategies such as obfuscation and encryption to hide the payload within PDF files and avoid detection measures. Zero-day exploits target previously undiscovered vulnerabilities, making them difficult to detect using traditional detection methods until a patch or signature update is released. Encrypted PDFs create a challenge for detection since they require decryption before analysis, which allows attackers to hide dangerous content behind encryption.

Techniques for Detecting Malicious PDFs

Signature-based detection: Signature-based detection compares a file's digital signature to a database of known malware signatures. While it works well against known threats, it struggles with zero-day exploits and polymorphic malware.

Heuristic Analysis: Heuristic analysis is detecting unusual patterns or behaviors in PDF files that could suggest malevolent intent. This method is more adaptable than signature-based detection, but it can also result in false positives.

Sandbox Analysis: Sandbox analysis involves executing PDF files in a controlled environment to study their behavior and discover malicious activity. It provides insight into the file's runtime behavior without compromising the host system.

Advanced Techniques for Malicious PDF Detection

Behavior-based Analysis: Behavior-based analysis focuses on monitoring the runtime behavior of PDF files to detect any suspicious or malicious activities, such as file system modifications, network connections, and system registry changes.

Content Analysis: Content analysis involves examining the actual content of PDF files, including text, images, and embedded objects, to identify anomalies or malicious elements that may indicate a security threat.

Metadata Analysis: Metadata analysis entails inspecting the metadata associated with PDF files, such as author information, creation date, and software version, to uncover potential indicators of compromise or malicious intent.

Dynamic Analysis: Dynamic analysis involves executing PDF files in a dynamic analysis environment, such as a virtual machine or sandbox, to observe their behavior in real-time and detect any malicious actions or code execution attempts.

Advanced Solutions

Modern antivirus software detects and decreases dangerous PDF files by combining signature-based detection, heuristic analysis, and machine learning. IDS systems monitor network traffic for signals of malicious behavior, such as the download or transmission of suspicious PDF files, to provide an extra layer of protection against PDF-based attacks. Endpoint protection solutions include antivirus, firewall, and other security technologies that defend computers from a variety of threats, including malicious PDF files. Cloud-based security solutions provide real-time threat identification and analysis, using cloud infrastructure to detect and block harmful PDFs before they reach the endpoint.

3.2 Proposed Methodology

This research seeks to assess the risk of cybersecurity vulnerabilities among average internet users who may be unaware of potential threats. By analyzing online attacks and threats, this paper aims to identify areas where proactive security measures can mitigate risks and protect users from potential harm. The Malicious File Checker web application, built using JavaScript, provides valuable assistance to users who may not be familiar with internet security or lack extensive knowledge of online threats. The global variable function is used in the website to store the reference to the drop area where files can be uploaded. A drag-and-drop behavior function used in JavaScript-built antivirus web applications for preventing the default drag-and-drop behaviors. Additionally, a CSS class is applied to highlight the drop area when a file is dragged over it by users. The application initializes various elements (such as buttons and input fields) using event listeners to enhance user interaction. The 'handle drop' function is implemented for managing dropped files by users, when a PDF file is dropped, it displays the file name. This function ensures that only one file is accepted and verifies its validity, thereby improving the user experience during file uploads. The website alerts users if multiple files are dropped or if the file is incorrect. These functions collectively enable users to easily upload a single PDF file, receive visual feedback, and proceed with further analysis.

This antivirus application analyzes or scans uploaded PDF files and delivers results regarding their potential malicious nature. Its user-friendly interface provides ease of use, guiding users through clear messages ("The malicious File Checker uses advanced algorithms to analyze PDF files for malicious content. Simply upload your PDF file, and our system will scan it against multiple antivirus engines and other security measures. Our tool checks for various indicators of malicious activity, including suspicious file signatures, embedded scripts, known malware patterns, and more. You can trust our service to provide accurate and comprehensive analysis results") displayed on the front webpage. The underlying code is optimized for efficiency, ensuring rapid results. The user simply needs to upload any suspicious file they believe may be harmful to their system.

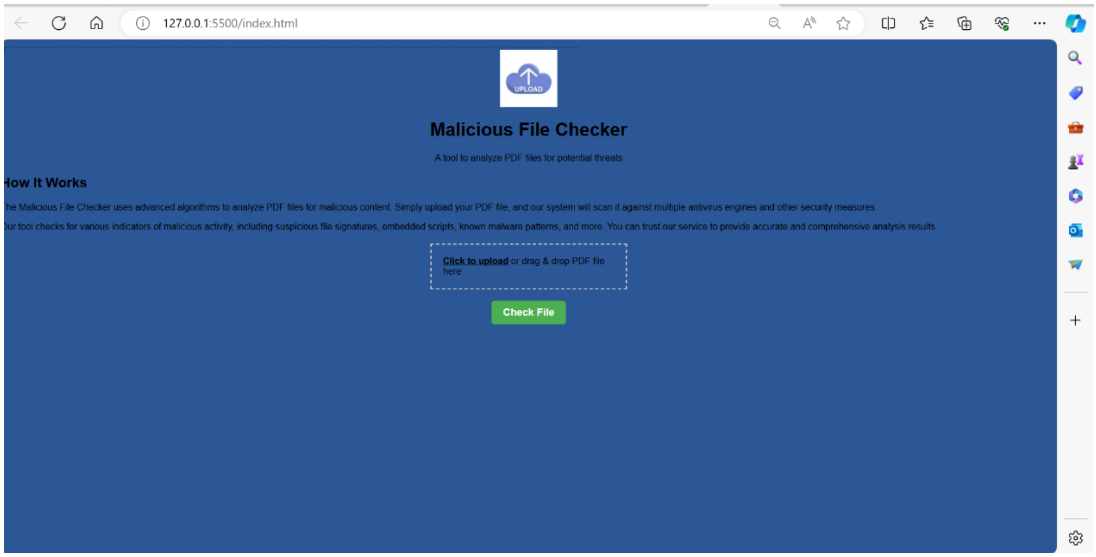


Figure 3. Front Page of Malicious File Checker

Figure 3 shows the depicted web interface showcasing a user-friendly design for a "malicious File Checker". Its purpose is to assist users in determining the safety of files they encounter. Here are the steps involved:

File upload option (users can choose between two methods):

Click to upload: By clicking this button, users are redirected to their local file storage. They can select a PDF file for analysis.

Drag and Drop: Alternatively, users can drag a suspicious file directly onto the open web application.

After selecting or dragging the file, users click the "**Check File**" button. The website initiates a rapid scan of the uploaded file, during this brief processing period, the system evaluates the file's content and structure. After all these processes within seconds, the website provides a result to the user accordingly whether the uploaded file is malicious or not. This intuitive interface aims to empower users by simplifying the process of assessing file security. By following these steps, individuals can make informed decisions about the files they encounter online.

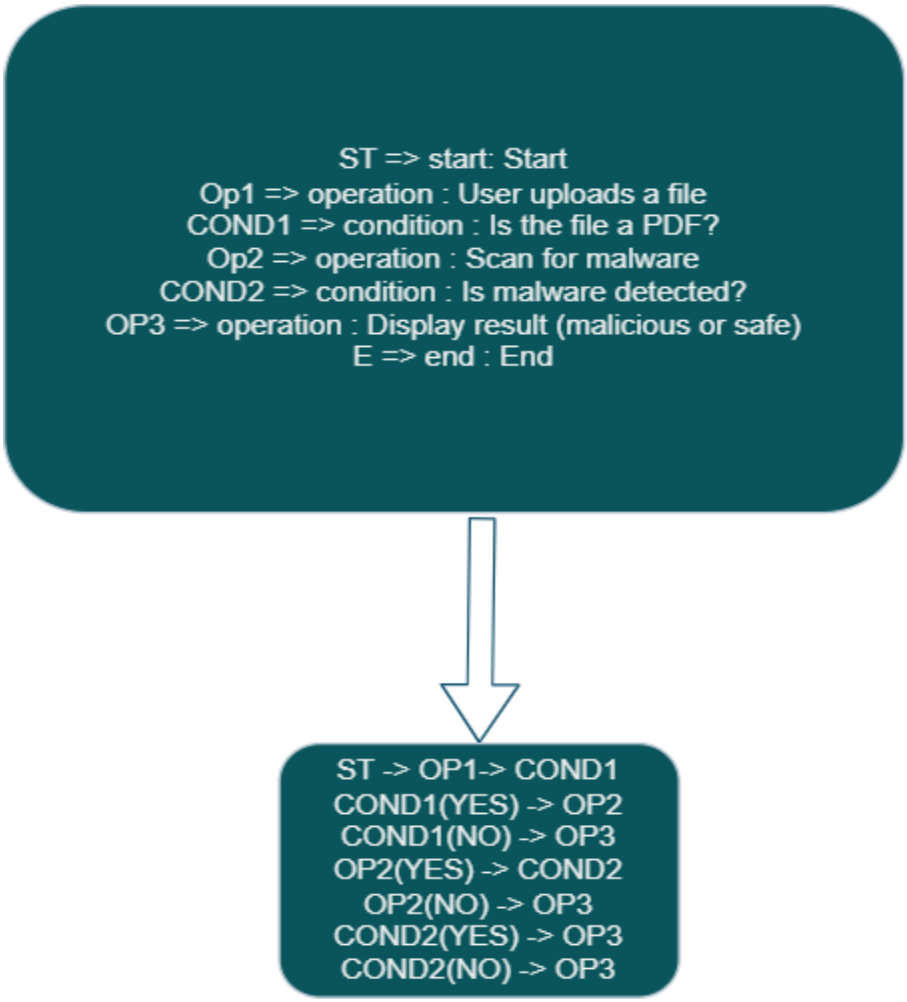


Figure 4. Flowchart of the Existing System

Figure 4. and Figure **4.1** shows a normal working flowchart of an application. When a PDF file is uploaded, the antivirus application calculates its MD5 hash, which acts as a fingerprint for the file. If the same PDF is encountered again, the application recalculates its MD5 hash and compares it with the stored hash. similar to MD5, the checker computes the SHA-1 hash for the uploaded file and it serves as another unique identifier for the file, generally, it means that when the user uploads a PDF file, the Malicious File Checker computes the hash values (MD5, SHA-1, and SHA-256) and it serves as a signature for the uploaded file. After all this, the application compares the signatures with known malicious hashes in its database, if a match occurs the file is flagged as potentially malicious or shows an error and if a match does not occur then the file is considered clean. These cryptographic hash functions inserted in the application play a crucial role in verifying and detecting any unauthorized alterations or malware within PDF files.

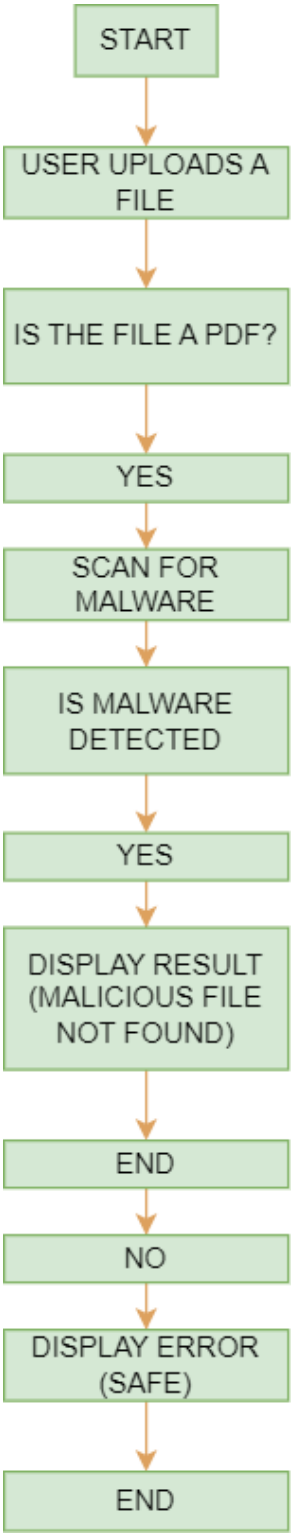


Figure 4.1 Flowchart of Existing System

In the realm of cybersecurity, assessing the integrity of files, especially those suspected of being malicious is paramount. By scrutinizing files for potential threats, organizations as well as any individual can safeguard their system and mitigate the impact of malware attacks. several effective approach involves employing a malicious file checker, which follows a well-defined process:

(a) Hash Calculation: Upon user file upload, the checker computes a cryptographic hash (such as SHA-256) for the uploaded file. This hash serves as a unique identifier for the file.

(b) Virus Total API Integration: The checker then initiates an API request to Virus Total, a reputable online service that aggregates threat intelligence from various antivirus engines. Using the calculated hash, it queries the virus total's extensive database.

(c) Malware Assessment: The API response provides crucial insights. It indicates whether the given PDF file is malicious or benign. If any malicious detections are found, the checker promptly alerts the users with an appropriate visual indicator (such as a red cross).

(d) Detailed Information Displays: Beyond the binary verdict, the checker enriches the user experience by presenting comprehensive details about the file:

File Attributes: Name, type, and size.

Temporal context: First seen date and last seen date.

Usage Metrics: Submission count and detection count.

Keeping PDF reader software and operating systems up to date with the latest security patches and updates helps mitigate the risk of exploitation by known vulnerabilities. Educating users about the potential risks associated with opening or downloading PDF files from untrusted sources and encouraging safe browsing practices can help prevent inadvertent exposure to malicious content. Implementing a multi-layered defense strategy that combines signature-based detection, heuristic analysis, sandboxing, and machine learning capabilities provides comprehensive protection against a wide range of PDF-based threats. Participating in threat intelligence sharing initiatives and collaborating with industry peers and security vendors enables organizations to stay informed about emerging threats and enhance their detection and response capabilities.

The web application "**Malicious File Checker**" developed using JavaScript, is designed to detect malicious PDF files. it is a sophisticated system that focuses on analyzing and understanding the

structure and content of PDF files. it uses this knowledge to anticipate and identify potential security threats. it processes a collection of PDF files, extracting crucial data and attributes from each file like file size, and specific byte sequences commonly associated with malicious files. then these extracted data are used for analysis and to predict whether a PDF file is malicious or not. It is capable of identifying patterns and associations within the dataset that may have previously gone unnoticed. This will provide accurate predictions for new, unseen PDF files. it undergoes a cleaning phase where any missing or incomplete data is addressed or not for ensure the analysis and predictions are based or depend upon complete accurate data. This JavaScript-based web application serves as a robust tool for detecting malicious PDF files, providing valuable insights and predictions based on comprehensive data analysis. It's generally a testament to the power of data-driven approaches in enhancing cybersecurity measures. The principles of adaptive detection, dynamic strategy modification, continuous learning, error correction, and reinforcement learning of a malicious file checker website can be effectively applied to the field of cybersecurity, particularly in the development of web applications for detecting malicious files. applications can learn and adapt their detection strategies over time, improving their performance and accuracy as applications interact with more and more files. this web application malicious file checker continues to learn and adapt detection strategy over time. It selects the course of action that provides the greatest benefit, which in this context would be accurately detecting malicious files while minimizing false positives and negatives. Application is capable of correcting mistakes made during its learning process. Involving in adjusting the detection strategy if it finds certain types of files are consistently being misclassified. reinforcement is a versatile approach used in conjunction with other techniques to enhance the performance of the detection system.

Continuous Improvement and Adaptation: Integrating threat intelligence feeds and indicators of compromise (IOCs) into detection solutions enables organizations to proactively identify and respond to emerging PDF-based threats in real time. Adopting an adaptive security architecture that can dynamically adjust detection and response mechanisms based on evolving threat landscapes and organizational risk profiles enhances resilience and agility in combating malicious PDF files. Establishing a feedback loop for gathering insights from security incidents, false positives, and evasion attempts enables continuous improvement of detection algorithms and strategies to stay ahead of adversaries.

Ethical Considerations in Malicious PDF Detection: When analyzing PDF files for malicious content, it's essential to ensure the privacy of users' sensitive information contained within the documents, adhering to ethical principles and legal requirements. Security researchers and vendors should follow responsible disclosure practices when identifying and reporting vulnerabilities in PDF reader software to facilitate timely patching and minimize the risk of exploitation by malicious actors.

Chapter 4

Results and Implementation

4.1 Screenshot of Programs

Script.js code

```
let dropArea; // Declare dropArea globally

function preventDefaults(e) {
    e.preventDefault();
    e.stopPropagation();
}

function highlight() {
    dropArea.classList.add('highlight');
}

function unhighlight() {
    dropArea.classList.remove('highlight');
}

async function handleDrop(e) {
    const dt = e.dataTransfer;
    const files = dt.files;

    // Ensure that only one file is dropped
    if (files.length !== 1) {
        alert('Please drop only one file');
        return;
    }

    const file = files[0];
    const fileType = file.type;
```

Figure 5.Code of the Application

```
    // Check if the dropped file is a PDF
    if (fileType !== 'application/pdf') {
        alert('Please drop a PDF file');
        return;
    }

    // Display the file name
    const fileNameDisplay = document.getElementById('fileNameDisplay');
    fileNameDisplay.textContent = `Uploaded File: ${file.name}`;

    // Proceed with file upload and analysis
    const fileInput = document.getElementById('fileInput');
    fileInput.files = files;
    checkFile();
}

document.addEventListener('DOMContentLoaded', () => {
    const checkBtn = document.getElementById('checkFileBtn');
    const fileInput = document.getElementById('fileInput');
    const dropArea = document.getElementById('dropArea');
    const dropText = document.querySelector('.click-to-upload');
    const fileNameDisplay = document.getElementById('fileNameDisplay'); // Get the file name display element

    function displayFileName(fileName) {
        fileNameDisplay.textContent = `Uploaded File: ${fileName}`; // Display the file name
    }
```

Figure 6 Code of the Application

```
    function clearFileName() {
        fileNameDisplay.textContent = ''; // Clear the file name display
    }

    // Prevent default drag behaviors
    ['dragenter', 'dragover', 'dragleave', 'drop'].forEach(eventName => {
        dropArea.addEventListener(eventName, preventDefaults, false);
    });

    // Highlight drop area when file is dragged over it
    ['dragenter', 'dragover'].forEach(eventName => {
        dropArea.addEventListener(eventName, highlight, false);
    });

    // Unhighlight drop area when file is dragged out of it
    ['dragleave', 'drop'].forEach(eventName => {
        dropArea.addEventListener(eventName, unhighlight, false);
    });

    // Handle dropped files
    dropArea.addEventListener('drop', handleDrop, false);

    // Click event listener for 'click to upload' text
    dropText.addEventListener('click', () => {
        fileInput.click(); // Trigger file input when 'click to upload' text is clicked
    });
```

Figure 7 Code of the Application

```
// Change event listener for file input
fileInput.addEventListener('change', () => {
    const files = fileInput.files;
    if (files.length === 1) {
        const fileType = files[0].type;
        if (fileType !== 'application/pdf') {
            alert('Please select a PDF file');
            fileInput.value = ''; // Clear the file input
            clearFileName(); // Clear the file name display
            return;
        }
        displayFileName(files[0].name); // Display the file name
        checkFile();
    } else {
        alert('Please select only one file');
        fileInput.value = ''; // Clear the file input
        clearFileName(); // Clear the file name display
    }
});

// Function to handle file drop
async function handleDrop(e) {
    const dt = e.dataTransfer;
    const files = dt.files;
```

Figure 8

```
// Function to handle file drop
async function handleDrop(e) {
    const dt = e.dataTransfer;
    const files = dt.files;

    // Ensure that only one file is dropped
    if (files.length !== 1) {
        alert('Please drop only one file');
        return;
    }

    const file = files[0];
    const fileType = file.type;

    // Check if the dropped file is a PDF
    if (fileType !== 'application/pdf') {
        alert('Please drop a PDF file');
        return;
    }

    // Proceed with file upload and analysis
    fileInput.files = files;
    displayFileName(file.name); // Display the file name
    checkFile();
}
```

Figure 9

```
async function checkFile() {
    const apiKey = '47b4e237eb761d6421af3440c0f82a321049a3c1ffbd08b22dc9af713f7cbcfe';
    const fileInput = document.getElementById('fileInput');
    const modal = document.getElementById('myModal');
    const resultDiv = document.getElementById('result');
    resultDiv.innerHTML = '';

    const file = fileInput.files[0];
    if (!file) {
        alert('Please select a file');
        return;
    }

    const hash = await calculateFileHash(file);
    if (!hash) {
        alert('Unable to calculate hash');
        return;
    }

    try {
        const response = await fetch(`https://www.virustotal.com/api/v3/files/${hash}`, {
            headers: {
                'x-apikey': apiKey
            }
        });
        const data = await response.json();
```

Figure 10

```
        if (response.ok) {
            displayResult(data);
            modal.style.display = "block"; // Show modal after displaying result
        } else {
            alert('Error: ' + data.error.message);
        }
    } catch (error) {
        console.error('Error:', error);
        alert('An error occurred, please try again later.');
    }
}

function displayResult(data) {
    const resultDiv = document.getElementById('result');

    // Check if any malicious things were found
    const hasMalicious = data.data.attributes.total > 0;

    // Display relevant information from the API response
    const fileInfo = data.data.attributes;
    let message = '';

    if (hasMalicious) {
        // Display error message with red cross icon
        message = `<div style="color: red; font-size: 24px;">&#10060; Possible malicious things found!</div>`;
```

Figure 11

```
} else {
    // Display success message with green checkmark icon
    message = `<div style="color: green; font-size: 24px;">&#10004; No malicious things found!</div>`;
}

const resultHTML = `
<h2>File Information:</h2>
${message}
<ul>

<li><strong>Name:</strong> ${fileInfo.names.join(', ')}</li>
<li><strong>Size:</strong> ${fileInfo.size} bytes</li>
<li><strong>Type:</strong> ${fileInfo.type}</li>
<li><strong>First Seen:</strong> ${fileInfo.first_seen}</li>
<li><strong>Last Seen:</strong> ${fileInfo.last_seen}</li>
<li><strong>Times Submitted:</strong> ${fileInfo.times_submitted}</li>
<li><strong>Times Detected:</strong> ${fileInfo.times_detected}</li>
<li><strong>Number of Engines Detected:</strong> ${fileInfo.last_analysis_results ? Object.keys(fileInfo.last_analysis_results).length
<li><strong>Reputation:</strong> ${fileInfo.reputation}</li>
<li><strong>Trusted Votes:</strong> ${fileInfo.trusted_votes}</li>
<li><strong>Malicious Votes:</strong> ${fileInfo.malicious_votes}</li>
<li><strong>Category:</strong> ${fileInfo.category}</li>
<li><strong>Last Modification Date:</strong> ${fileInfo.last_modification_date}</li>
<li><strong>Total Votes:</strong> ${fileInfo.total_votes}</li>
<li><strong>MD5:</strong> ${fileInfo.md5}</li>
<li><strong>SHA-1:</strong> ${fileInfo.sha1}</li>
<li><strong>SHA-256:</strong> ${fileInfo.sha256}</li>
```

Figure 12

```
<li><strong>Magic:</strong> ${fileInfo.magic}</li>
<li><strong>Entropy:</strong> ${fileInfo.entropy}</li>
<li><strong>First Bytes (hex):</strong> ${fileInfo.first_bytes}</li>
<li><strong>Times Analysed:</strong> ${fileInfo.times_analysed}</li>
<li><strong>Downloadable:</strong> ${fileInfo.downloadable}</li>
<li><strong>Upload Timestamp:</strong> ${fileInfo.upload_timestamp}</li>
<li><strong>Scan Date:</strong> ${fileInfo.scan_date}</li>
<li><strong>Positives:</strong> ${fileInfo.positives}</li>
<li><strong>Total:</strong> ${fileInfo.total}</li>
<li><strong>Tags:</strong> ${fileInfo.tags}</li>
<li><strong>Similar Samples:</strong> ${fileInfo.similar_samples}</li>
<li><strong>Meaningful Name:</strong> ${fileInfo.meaningful_name}</li>
<li><strong>Period:</strong> ${fileInfo.period}</li>
<li><strong>Size Lit Endian:</strong> ${fileInfo.size_little_endian}</li>
<li><strong>Signature Exists:</strong> ${fileInfo.signature_exists}</li>
<li><strong>Detected Engines:</strong> ${fileInfo.detected_engines}</li>
<li><strong>Submitted From:</strong> ${fileInfo.submitted_from}</li>
<li><strong>Verdict:</strong> ${fileInfo.verdict}</li>
<li><strong>Meaningful Name Unicode:</strong> ${fileInfo.meaningful_name_unicode}</li>
<li><strong>First Seen ITW:</strong> ${fileInfo.first_seen_itw}</li>
<li><strong>File Type Extension:</strong> ${fileInfo.file_type_extension}</li>
<li><strong>Analysis Start Time:</strong> ${fileInfo.analysis_start_time}</li>
<li><strong>Analysis End Time:</strong> ${fileInfo.analysis_end_time}</li>
<li><strong>Number of ITW Names:</strong> ${fileInfo.num_itw_names}</li>
<li><strong>Times Executed:</strong> ${fileInfo.times_executed}</li>
<li><strong>Original Signature:</strong> ${fileInfo.original_signature}</li>
```

Figure 13

```
<li><strong>Size Lit Endian:</strong> ${fileInfo.size_little_endian}</li>
<li><strong>Signature Exists:</strong> ${fileInfo.signature_exists}</li>
<li><strong>Detected Engines:</strong> ${fileInfo.detected_engines}</li>
<li><strong>Submitted From:</strong> ${fileInfo.submitted_from}</li>
<li><strong>Verdict:</strong> ${fileInfo.verdict}</li>
<li><strong>Meaningful Name Unicode:</strong> ${fileInfo.meaningful_name_unicode}</li>
<li><strong>First Seen ITW:</strong> ${fileInfo.first_seen_itw}</li>
<li><strong>File Type Extension:</strong> ${fileInfo.file_type_extension}</li>
<li><strong>Analysis Start Time:</strong> ${fileInfo.analysis_start_time}</li>
<li><strong>Analysis End Time:</strong> ${fileInfo.analysis_end_time}</li>
<li><strong>Number of ITW Names:</strong> ${fileInfo.num_itw_names}</li>
<li><strong>Times Executed:</strong> ${fileInfo.times_executed}</li>
<li><strong>Original Signature:</strong> ${fileInfo.original_signature}</li>
<li><strong>PE Compile Time:</strong> ${fileInfo.pe_compile_time}</li>
<li><strong>URLs:</strong> ${fileInfo.urls}</li>
<li><strong>File Name:</strong> ${fileInfo.file_name}</li>
<li><strong>File Size:</strong> ${fileInfo.file_size}</li>
<li><strong>Resource:</strong> ${fileInfo.resource}</li>
<li><strong>Packer:</strong> ${fileInfo.packer}</li>
<li><strong>Code Size:</strong> ${fileInfo.code_size}</li>
<li><strong>Preferred Dumper:</strong> ${fileInfo.preferred_dumper}</li>

<!-- Add more information as needed -->
</ul>
`;
```

Figure 14

```
    resultDiv.innerHTML = resultHTML;

    // Show modal after displaying result
    const modal = document.getElementById('myModal');
    modal.style.display = "block";

    // Close modal after 10 seconds
    setTimeout(() => {
        modal.style.display = "none";
    }, 15000);
}


function calculateFileHash(file) {
    return new Promise((resolve, reject) => {
        const reader = new FileReader();
        reader.onload = function(event) {
            const arrayBuffer = event.target.result;
            crypto.subtle.digest('SHA-256', arrayBuffer).then(hashBuffer => {
                const hashArray = Array.from(new Uint8Array(hashBuffer));

                const hashHex = hashArray.map(byte => byte.toString(16).padStart(2, '0')).join('');
                resolve(hashHex);
            }).catch(error => reject(error));
        };
```

Figure 15

```
        reader.onerror = function(error) {
            reject(error);
        };
        reader.readAsArrayBuffer(file);
    });
}

// Get the modal
const modal = document.getElementById('myModal');

// Get the <span> element that closes the modal
const span = document.getElementsByClassName('close')[0];

// When the user clicks the 'x' button, close the modal
span.onclick = function() {
    modal.style.display = "none";
}

// When the user clicks anywhere outside of the modal, close it
window.onclick = function(event) {
    if (event.target == modal) {
        modal.style.display = "none";
    }
}
```

Figure 16

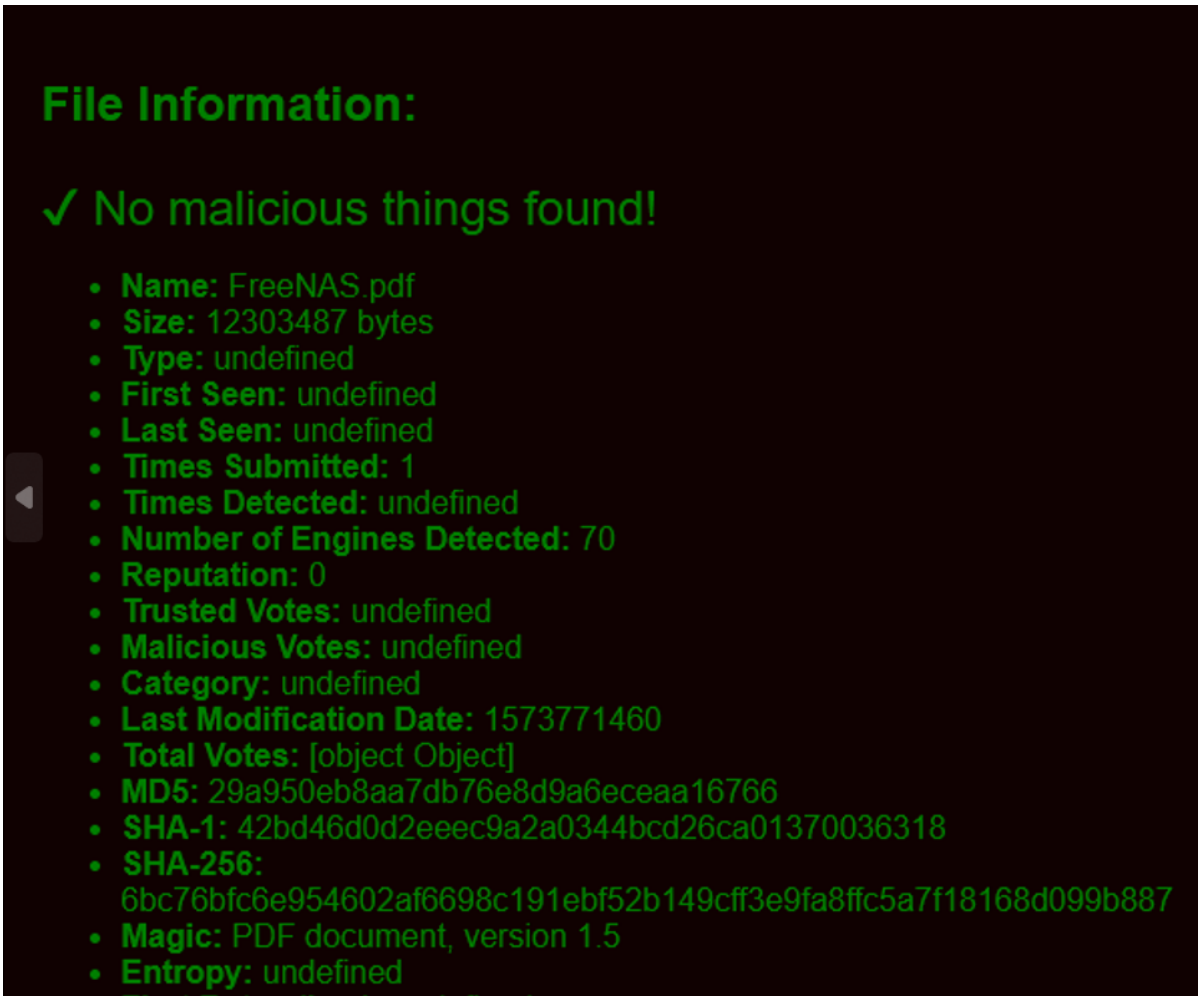


Figure 17

Figure 17 shows an antivirus web application result after scanning the uploaded file in which every information of the PDF file is shown whether a PDF file is malicious or not.

This result that shows after the detection of an uploaded file, after analyzing the file the web application gives the result about whether the malicious file is infected or not. the result shown to the users included with description or information about the file like name of the file, size of the file. number of engines applied or detected are also visible in the result. last modification date and many other information about the file are included with the results and the user can see this information very clearly.

The outcomes of this study should not have any negative impact on the cybersecurity business; that is, the results of this study should not be used to disparage specific antivirus providers as being

less qualitative than others based only on the findings. This is primarily due to compliance with the regulations and conditions of service suggested by the virus total.

The code we used during the construction of the website communicates with the virus total application programming interface for detection. It sends a calculated hash of the user's uploaded files and get a response from the application programming interface with detailed information including indicators, detection rates, and more.

As I mentioned above about the analysis process, Static analysis operates by deconstructing code into manageable segments, facilitating the identification of malicious code lines. However, if the code under scrutiny has been obfuscated, static analysis may not produce accurate results. On the other hand, dynamic analysis, which involves the deliberate execution of the file to study its impact on the operating system and the system's response to the file's execution, can be considered riskier. Despite the potential risks, it provides valuable insight into the real-time behavior of the file within a system environment. These two methods together form a comprehensive approach to analyzing and securing software systems[7]. These all important facts are considered during the building of the antivirus scanning website. Users can manually close the modal window using the close button or by clicking outside the modal area. this website addresses security or privacy to the user's credentials and sensitive information The given image in Figure 3 and Figure 4 is a result that shows after the detection of an uploaded file, after analyzing the file the web application gives the result about whether the malicious file is infected or not. the result shown to the users included with description or information about the file like name of the file, size of the file. number of engines applied or detected are also visible in the result. last modification date and many other information about the file are included with the results and the user can see this information very clearly.

The outcomes of this study should not have any negative impact on the cybersecurity business; that is, the results of this study should not be used to disparage specific antivirus providers as being less qualitative than others based only on the findings. This is primarily due to compliance with the regulations and conditions of service suggested by the virus total.

The code we used during the construction of the website communicates with the virus total application programming interface for detection. It sends a calculated hash of the user's uploaded files and get a response from the application programming interface with detailed information including indicators, detection rates, and more.

As I mentioned above about the analysis process, Static analysis operates by deconstructing code into manageable segments, facilitating the identification of malicious code lines. However, if the code under scrutiny has been obfuscated, static analysis may not produce accurate results. On the other hand, dynamic analysis, which involves the deliberate execution of the file to study its impact on the operating system and the system's response to the file's execution, can be considered riskier. Despite the potential risks, it provides valuable insight into the real-time behavior of the file within a system environment. These two methods together form a comprehensive approach to analyzing and securing software systems[7]. These all important facts are considered during the building of the antivirus scanning website. Users can manually close the modal window using the close button or by clicking outside the modal area. this website addresses security or privacy to the user's credentials and sensitive information..

Style.css code :

```css
body {
    margin: 0;
    font-family: Arial, sans-serif;
    background-color: #2b5797; /* VirusTotal blue */
}


.background-img {
    height: 100vh;
    display: flex;
    justify-content: center;
    align-items: center;
}


.container {
    text-align: center;
}

.drop-area {
    border: 2px dashed #ccc;
    padding: 20px;
    margin: 20px auto;
    cursor: pointer;
    max-width: 300px;
}
```

Figure 18.

```css
.click-to-upload {
    font-weight: bold; /* Make the text bold */
    text-decoration: underline; /* Underline the text */
    cursor: pointer; /* Change cursor to pointer on hover */
}

.drop-text {
    margin: 0;
}

.drop-area input {
    display: none;
}

#checkBtn {
    padding: 10px 20px;
    background-color: #4CAF50;
    color: white;
    border: none;
    cursor: pointer;
}

#checkBtn:hover {
    background-color: #45a049;
}
```

Figure 19

```css
/* Modal styles */
.modal {
    display: none; /* Hidden by default */
    position: fixed; /* Stay in place */
    z-index: 1; /* Sit on top */
    left: 0;
    top: 0;
    width: 100%; /* Full width */
    height: 100%; /* Full height */
    overflow: auto; /* Enable scroll if needed */
    background-color: rgb(0,0,0); /* Fallback color */
    background-color: rgba(0,0,0,0.4); /* Black w/ opacity */
}

/* Header styles */
header {
    text-align: center;
}

#logo {
    width: 100px; /* Adjust size as needed */
    height: 100px; /* Adjust size as needed */
}

.description {
    margin-bottom: 20px;
}
```

Figure 20

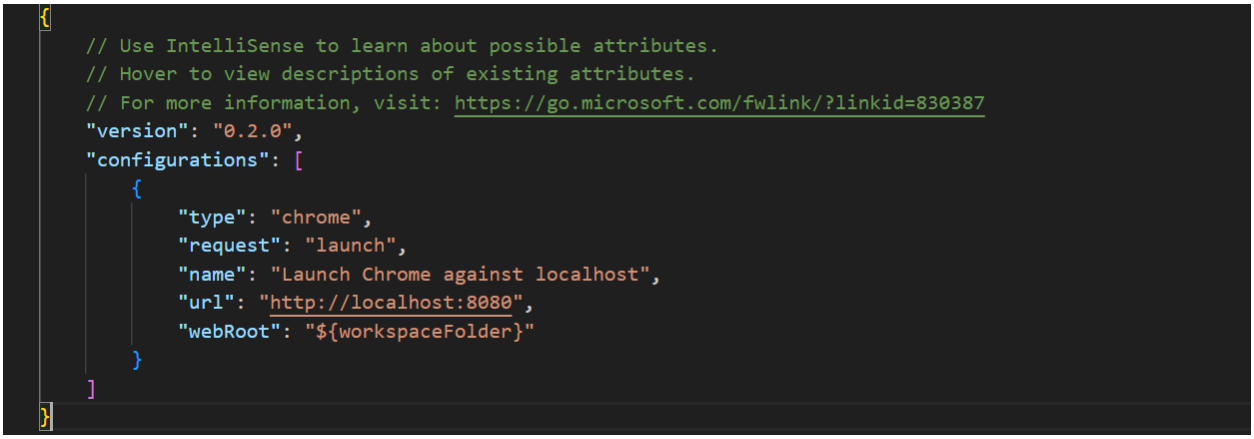
```css
.modal-content {
    background-color: #100101;
    margin: 15% auto;
    padding: 20px;
    border: 1px solid #888;
    width: 80%;
    max-width: 600px;
    color: green; /* Set font color to green */
}



.close {
    color: #aaa;
    float: right;
    font-size: 28px;
    font-weight: bold;
}

.close:hover,
.close:focus {
    color: black;
    text-decoration: none;
    cursor: pointer;
}
```

Figure 21

```css
#fileNameDisplay {
    margin-top: 10px;
    font-weight: bold;
    color: white; /* Set font color to white */
}


#checkFileBtn {
    display: block;
    margin: 20px auto; /* Center the button */
    padding: 10px 20px;
    font-size: 18px;
    font-weight: bold;
    background-color: #4CAF50; /* Green background */
    color: white;
    border: none;
    border-radius: 5px;
    cursor: pointer;
}

#checkFileBtn:hover {
    background-color: #45a049; /* Darker green background on hover */
}

#fileInfo {
    margin-top: 10px;
    font-weight: bold;
}
```

Figure 22

launch.json code:



```
{
    // Use IntelliSense to learn about possible attributes.
    // Hover to view descriptions of existing attributes.
    // For more information, visit: https://go.microsoft.com/fwlink/?linkid=830387
    "version": "0.2.0",
    "configurations": [
        {
            "type": "chrome",
            "request": "launch",
            "name": "Launch Chrome against localhost",
            "url": "http://localhost:8080",
            "webRoot": "${workspaceFolder}"
        }
    ]
}
```

Figure 23

4.2 Discussion

Enhanced threat intelligence tools will help enterprises keep ahead of developing PDF-based attacks by combining and evaluating threat data from several sources in real time. Express a personal reflection: Malicious PDF files offer a serious threat to businesses and individuals, necessitating effective detection and mitigation solutions to combat increasing cyber threats. By knowing the methodologies, problems, and solutions for identifying malicious PDFs, stakeholders may take proactive steps to improve their security posture and reduce the risk of exploitation. AI and machine learning technologies will continue to play an important role in enhancing the identification and mitigation of malicious PDF files, as they use powerful algorithms to scan massive datasets and identify emerging risks. Blockchain technology has the potential to improve PDF file security by enabling tamper-proof document verification and secure distribution procedures. The rapidly evolving threat landscape, including advancements in attack techniques and the emergence of new vulnerabilities, presents ongoing challenges for organizations in detecting and mitigating malicious PDF files. The adoption of emerging technologies, such as artificial intelligence, blockchain, and quantum computing, introduces both opportunities and risks for PDF file security, necessitating continuous research and innovation to stay ahead of

adversaries. The increasing complexity of regulatory compliance requirements, coupled with the global nature of PDF file security threats, poses challenges for organizations in maintaining compliance and adapting to evolving regulatory frameworks. The persistent shortage of skilled cybersecurity professionals and the widening skills gap exacerbates the challenges faced by organizations in effectively detecting and responding to PDF-based threats, highlighting the need for investment in training and talent development initiatives. Research into next-generation detection techniques, such as quantum-resistant cryptography and biologically inspired algorithms, holds promise for enhancing the resilience of PDF file security against future threats. Human-centric security solutions that focus on understanding and addressing user behavior, cognition, and decision-making processes offer potential avenues for mitigating social engineering attacks and improving overall PDF file security. Research into privacy-preserving technologies, such as homomorphic encryption and differential privacy, enables organizations to protect sensitive information contained within PDF files while maintaining compliance with data privacy regulations.

Chapter 5

CONCLUSION

In conclusion, the JavaScript-based antivirus Malicious File Checker has produced positive results for suspicious or malicious PDF prediction. With the use of given applications, prevention and early detection may be used to accurately identify those who have a higher risk of suffering cyber-attacks. However, it is crucial to remember that this JavaScript-based application should not replace traditional cybersecurity measures or be the sole factor in making security decisions.

Additional research is required to address issues with bias and fairness, and to ensure that this antivirus web application performs well across a variety of systems and networks. Overall, it holds great promise for enhancing our capacity to identify and prevent cybersecurity attacks. The suggested approach can be used by any non-technical individuals without access to sophisticated cybersecurity services or experts. Malicious File Checker helps cybersecurity professionals make quick decisions. This web application has certain limitations in that it will only determine the pdf format only. This antivirus makes its impossible to gauge the degree of severity of a cybersecurity attack.

REFERENCES

1. Ramanathan, M., & Kankanhalli, M. S. (2013). Malicious PDF detection using metadata and structural features. *Information Management & Computer Security*, 21(3), 218-234.

2. Fattori, A., Cavallaro, L., & Balzarotti, D. (2010, April). A generic approach to detect pdf-based malware. In *Proceedings of the 17th Annual Network and Distributed System Security Symposium* (pp. 1-17).

3. Fortinet. (2020). FortiGuard Labs Global Threat Landscape Report Q4 2019. Retrieved from https://www.fortinet.com/content/dam/fortinet/assets/threat-reports/fortiguard-labs-global-threat-landscape-report-q4-2019.pdf

4. Sood, A. K., & Enbody, R. J. (2013). PDFrate: A tool for detecting malicious PDF files. *Journal of Computer Virology and Hacking Techniques*, 9(4), 241-254.

5. Symantec. (2020). Internet Security Threat Report, Volume 25. Retrieved from https://www.symantec.com/content/dam/symantec/docs/reports/istr-25-2020-en.pdf

6. Lin, Y., Guo, W., & Zou, W. (2014). A Survey of Detection Methods on Malicious PDF Files. *International Journal of Computer Theory and Engineering*, 6(2), 118-123.

7. McAfee. (2019). McAfee Labs Threats Report: August 2019. Retrieved from https://www.mcafee.com/enterprise/en-us/assets/reports/rp-quarterly-threats-aug-2019.pdf

8. Nissim, N., Bohadana, M., & Elovici, Y. (2016). Malware detection by eating a whole executable. *IEEE Transactions on Information Forensics and Security*, 11(7), 1571-1584.

9. Cisco Talos. (2020). Threat Landscape Report: 2020 Midyear Update. Retrieved from https://www.cisco.com/c/dam/m/digital/elq-cmcglobal/witb/documents/cisco-talos-2020-threat-landscape-report.pdf

10. Li, J., Wu, Y., Lin, Y., & Wang, C. (2013, October). Malicious PDF document detection based on characteristic signals. In *2013 6th International Congress on Image and Signal Processing* (pp. 277-281). IEEE.

[11] “Virus total intelligence.” [Online]. https://www.virustotal.com/gui/ intelligence-overview Available:

[12] Can Pdf have a viruses. How to Detect and Resolve it. Youtube link https://youtu.be/TjZsnKZqjbo?si=wPhD1zJoS_r487m6

[13] How Hackers Launch PDF Virus File And How We Can Protect Ourselves! (Cybersecurity) https://youtu.be/3RSn9JwnWlQ?si=_RkP6F9RWBAi4PG5

[14] K.vanLiebergen, J. Caballero, P. Kotzias, and C. Gates, “A deep dive into virustotal: Characterizing and clustering a massive file feed,” 2022.

RP

ORIGINALITY REPORT

4%	3%	2%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	fastercapital.com Internet Source	2%
2	www.semanticscholar.org Internet Source	1%
3	Sonal Dabral, Amit Agarwal, Manish Mahajan, Sachin Kumar. "Chapter 14 Malicious PDF Files Detection Using Structural and Javascript Based Features", Springer Science and Business Media LLC, 2017 Publication	1%
4	br.hinative.com Internet Source	<1%
5	docplayer.net Internet Source	<1%
6	dokumen.pub Internet Source	<1%
7	Yuning Cui, Yi Sun, Jun Luo, Yonghui Huang, Yuxuan Zhou, Xuelei Li. "MMPD: A Novel Malicious PDF File Detector for Mobile Robots", IEEE Sensors Journal, 2020	<1%

	Publication	
8	Karima Belmabrouk. "chapter 11 Cyber Criminals and Data Privacy Measures", IGI Global, 2023 Publication	<1 %
9	www.enisa.europa.eu Internet Source	<1 %

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		