**Frontiers in Computing and Intelligent Systems**
ISSN: 2832-6024 | Vol. 5, No. 2, 2023

# Survey of Malicious PDF Attacks

**Shuhan Wang ***

Department of Computer Science, Southwest Jiaotong University, Chengdu 610031, China
*** Corresponding author Email:** sc21s2w@leeds.ac.uk

**Abstract:** In recent years, malicious documents have gained widespread attention as one of the primary vectors for Advanced Persistent Threat (APT) attacks. These malicious document attacks employ various sophisticated techniques, including stream object attacks, embedded JavaScript, information entropy attacks, and machine learning. Therefore, it is essential to pay attention to the existing research findings and trends. Stream object attacks leverage stream objects in PDF or Office documents to hide malicious code, bypassing traditional detection methods. Embedded JavaScript executes malicious actions, such as downloading malware or launching network attacks. Information entropy attacks utilize the information entropy characteristics within documents to detect concealed malicious activities. This article examines and summarizes the current status and progress in each domain. In conclusion, it reviews and proposes research related to embedded JavaScript, information entropy attacks, and machine learning.

**Keywords:** Entropy Attack; Embedded JavaScript Attack; Stream Object Attack; Feature Extraction.

## 1. Introduction

In today's digital age, the threat of malicious documents is becoming increasingly severe. Malicious documents refer to files that are designed to deceive, infect, or attack computer systems and users, such as malware, viruses, spyware, etc. These malicious documents can be disseminated through emails, downloads, web links, or portable devices, posing a significant threat to the information security of individual users, businesses, and organizations. With the widespread application of Artificial Intelligence (AI) across various fields, we have enjoyed many conveniences. However, this has also brought about an increase in malicious attacks, especially threats propagated through malicious documents.

The purpose of this review is to explore the issues, methods, and latest developments in malicious document detection. This aims to enhance the understanding of the threat of malicious documents and provide researchers, security experts, and decision-makers with a comprehensive overview of malicious document detection.

Detecting malicious documents is vital for protecting individual privacy, corporate secrets, and national security. By accurately and efficiently detecting malicious documents, we can take preventive measures at an early stage, reducing the spread and impact of malware, and preventing data breaches, financial losses, and other severe consequences like cybercrimes.

Currently, methods for analyzing malicious PDF document detection primarily fall into two categories: static analysis and dynamic analysis.

Static analysis methods determine the nature of a document without opening it, by analyzing embedded code, content, structure, and other features in the document. Furthermore, they employ multi-level abstraction methods that abstract and vectorize the document structure and scripting languages, utilizing machine learning for computation.

Dynamic analysis, on the other hand, entails opening the document within a virtual environment and monitoring its behavior during execution to determine if it's malicious. Depending on the principle of analysis, dynamic methods can be divided into document emulation and code simulation techniques [1].

Researchers have also started to integrate the behavioral characteristics of the document when opened with its inherent features for analysis, leading to the development of combined static and dynamic malicious document detection methods.

This review encompasses topics like social engineering tactics, cyberattacks, and vulnerability exploitation. It can reveal patterns and trends in the spread of malicious documents, enabling better prevention and response to their dissemination. This also aids others in gaining a deeper understanding of how malicious documents operate and the extent of their threats. By sharing this information, everyone can remain vigilant about emerging threats and take timely preventive measures.

## 2. Relation Work

### 2.1. Stream Object

The Stream Object Attack is a malicious document attack technique commonly found in PDF files. This attack leverages the characteristics of the PDF format by tampering with or manipulating the stream objects within a PDF document to facilitate malicious actions. In PDF files, a stream object is a data structure used to store document content, including text, images, fonts, and more. Attackers can exploit vulnerabilities or weaknesses in PDF readers by modifying or adding malicious stream objects, aiming to compromise or exploit the user's system.

### 2.2. Entropy

The entropy attack is a cryptographic attack method aimed at deciphering passwords or extracting keys by analyzing the information entropy within a cryptographic system. Information entropy is a measure of the amount of information present in the keys or plaintext of a cryptographic system. Attackers exploit the characteristics of information entropy to guess or infer key details in the cryptographic system. Entropy attacks typically operate based on principles of statistical analysis and probabilistic inference. Attackers might use a plethora of known plaintext-ciphertext pairs, and by analyzing patterns, frequencies, and statistical features

within this dataset, they strive to crack the keys or cryptographic algorithms within the system.

## 2.3. Embedded JavaScript

JavaScript attacks embedded in malicious documents are a common cyber security threat. Attackers take advantage of JavaScript's versatility and broad browser support to embed malicious code into documents, facilitating a range of malicious activities. This type of attack capitalizes on users' trust in documents. By deceiving users into opening a malicious document or visiting a malicious webpage, the embedded JavaScript code is triggered and executed. Once executed, the malicious JavaScript can carry out various malevolent actions such as stealing sensitive information, hijacking user sessions, implanting malware, attacking other websites or systems, and more.

# 3. Attack

## 3.1. Streaming Object Attack

There is an analysis method that can successfully extract JavaScript code nested within stream objects and, combined with Support Vector Machine (SVM) detection techniques, can detect these codes, achieving the detection of malicious PDF documents.

The system consists of four modules: User Shell Control Interface, Sample Analysis Module, SVM Detection Module, and Shellcode Analysis Module.

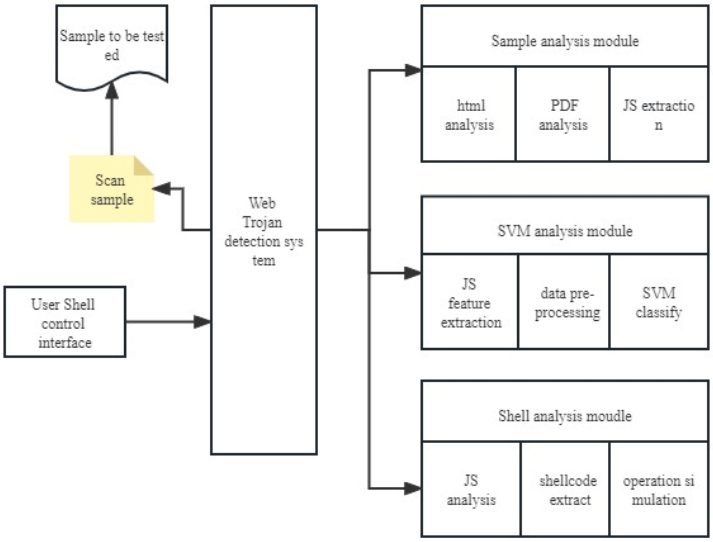


**Figure 1.** Net horse detection system overall framework

The entire system process is as follows: For the input sample to be tested, the system first determines whether its type is HTML or PDF, and based on the type, it invokes the corresponding analysis module for sample parsing.

For HTML samples, the system checks if the page's Iframe tags contain suspicious links and extracts the content within the <script></script> tags on the current page. If a suspicious link is discovered, the system further invokes a page downloader to retrieve the corresponding page, then repeatedly carries out page type determination, suspicious link extraction, page parsing, and JavaScript extraction processes.

For PDF samples, the system conducts static stream object analysis to locate JS stream objects and extracts the embedded JavaScript code. Then, using the feature classifier based on the Support Vector Machine, it assesses the maliciousness of the JavaScript code extracted from the sample analysis module. If the JavaScript code is determined to be a malicious segment, the system will call upon a JS engine to de-obfuscate that JavaScript code. If the de-obfuscated code contains shellcode, the system further invokes a shellcode simulation tool to emulate its behavior, aiming to obtain detailed information about the malicious script's functionality [2].

## 3.2. Embedded JavaScript

For detecting embedded JavaScript, one can employ specific models. The training phase for a malicious document detection model generally involves three steps. Initially, JavaScript code is extracted from benign PDF files using a PDF parser. Subsequently, a composite JavaScript feature is constructed by extracting semantic and malicious-related information from the gleaned JavaScript code, combining it into a token sequence. These token sequences are then transformed and encoded through an N-gram model, yielding the composite JavaScript feature. Finally, a one-class Support Vector Machine is utilized to train on these features, resulting in the malicious PDF document detection model [3].

It's important to note that the effectiveness of this method hinges on accurately extracting JavaScript code from PDF documents. However, some attackers might employ tactics to obfuscate the JavaScript code, making it elusive for standard parsers, thus rendering this approach ineffective. During the malicious PDF document detection phase, parsers such as Peepdf and PJscan can be used to analyze PDF documents, determining the presence of JavaScript code. If concealed JavaScript code is detected within a PDF, it is deemed malicious. The Spider Monkey tool can be used to convert JavaScript code into Token representations, where 86 tokens are defined to represent keywords, operators, and data within the JavaScript code.

Semantic and malicious-associated information are extracted and combined into Token sequences. Using the N-gram model, these Token sequences are transformed into sets of Token segments of length N. Binary encoding is applied to these Token segment sequences, resulting in vectorized JavaScript features. These JavaScript features are then normalized in preparation for subsequent malicious PDF document detection and classification.

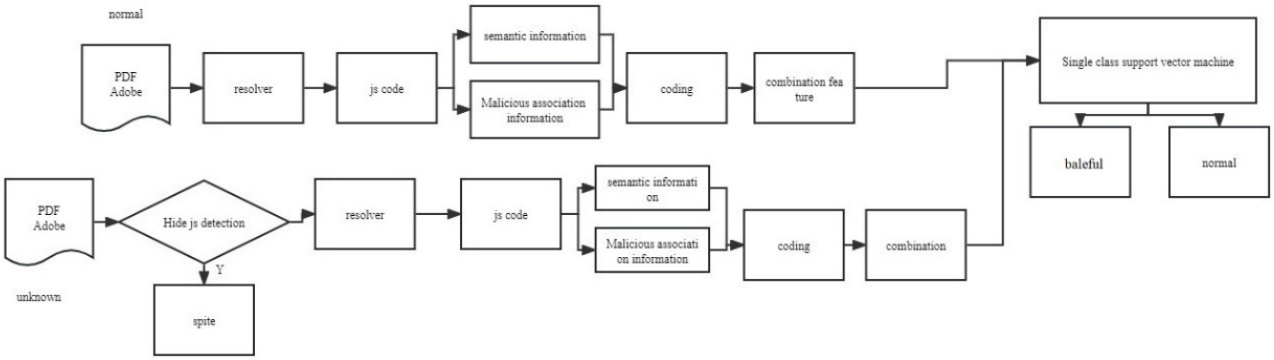


**Figure 2.** Detection model of JavaScript feature and single classification support vector machine

Additionally, attackers can also employ other methods to evade this detection model, such as targeting parsers to hinder JavaScript code extraction or intentionally causing classifiers to misclassify extracted code as benign. The robustness of the model can also be tested by carrying out mimicry attacks or anti-mimicry attacks, altering malicious samples to resemble benign samples, or injecting malicious content into benign samples, turning them into malicious documents. Furthermore, an efficient method to produce malicious samples can be implemented by directly constructing JavaScript code in the feature space, achieving the objective of rapidly generating attack samples.

In conclusion, the training phase of the malicious document detection model encompasses the extraction of JavaScript code, feature construction, and the training of the Support Vector Machine; while the detection phase includes preliminary detection, JavaScript code extraction, feature construction, and classification using the Support Vector Machine. Attackers could potentially evade this detection model by methods like obscuring JavaScript code, targeting parsers, or launching mimicry attacks.

## 3.3. Entropy

Network full-flow analysis technology integrates Deep Flow Inspection (DFI) and Deep Packet Inspection (DPI) techniques, aiming to address data features associated with network intrusion attacks and 0-day vulnerability threats. Extracting data information from different network layers, link layers, and application layers, facilitates the identification of anomalous network traffic packets and comprehensive network data flow.

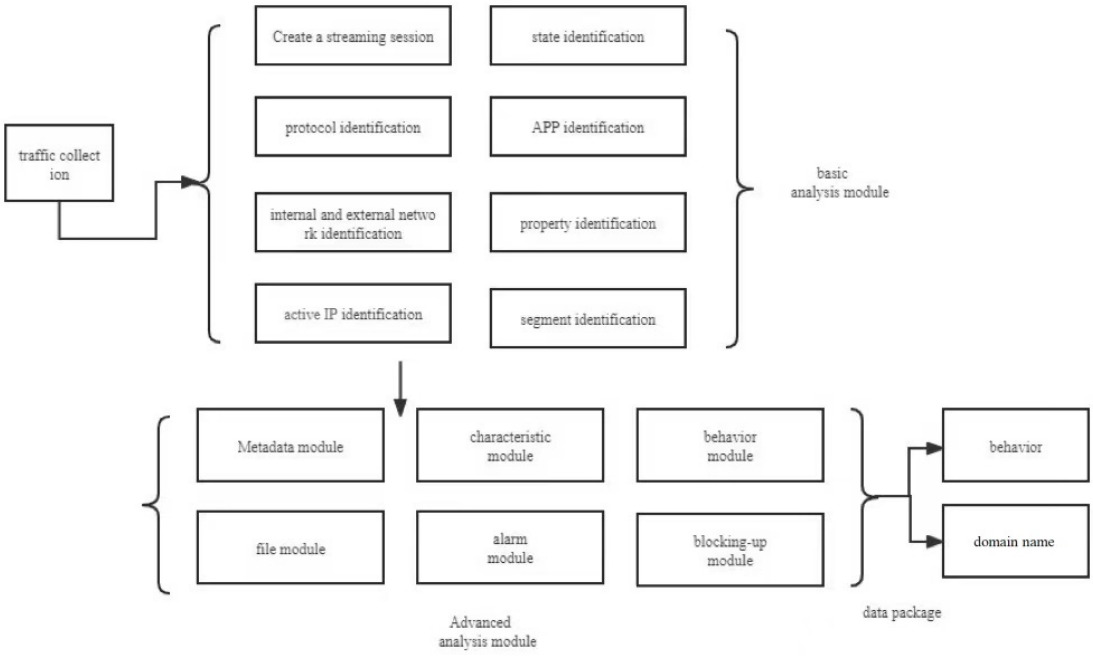


**Figure 3.** The execution flow of network full traffic analysis technology

The main components of network full-flow analysis technology can be summarized into three points:

Network Full-flow Collection: It senses the IP and MAC addresses of different traffic data. It gathers and integrates comprehensive network traffic, total packets, total bytes, IF endpoints, and other multi-dimensional traffic information.

Network Protocol Identification: Utilizing IP protocols and TCP/UDP protocols, operations are conducted according to the rules of the network protocol stack. It provides an in-depth view of network traffic and new sessions' data packet compositions over any given time range. This involves detecting, parsing, and bifurcating abnormal network traffic. By splitting and combining the analysis of various data packets and data streams, genuine information about user

intrusion attacks and network vulnerability threats is procured.

Deep Decoding and Restoration of Network Data Flow: The DFI data flow detection engine technology is employed to inspect traffic metrics of different network segments and network application entities. This includes the detection of network protocols, IP addresses, MAC addresses, and IP conversations, followed by issuing alerts. The DPI packet detection technology, in conjunction with the parsing recognition capability of the network protocol stack, is used to analyze network node ports, IP addresses, data packet flags, and data packet payloads. This helps in identifying concealed novel threat attacks within the network data flow [4].

Moreover, an abnormal threat intelligence analysis model is established to cater to the attack chain targeting network intrusions. This model connects different network nodes and the progression levels of intrusion attacks, thereby crafting a comprehensive attack chain execution process. The model encompasses components like intrusion target detection, malicious attack code detection, attack code distribution, Trojan program or link control, and extraction of anomalous attack traffic data. By analyzing intrusion attackers and abnormal threat elements, this model is adept at severing the attack chain nodes and propagation paths before the successful implementation of an attack, thus successfully thwarting threats and safeguarding network security.

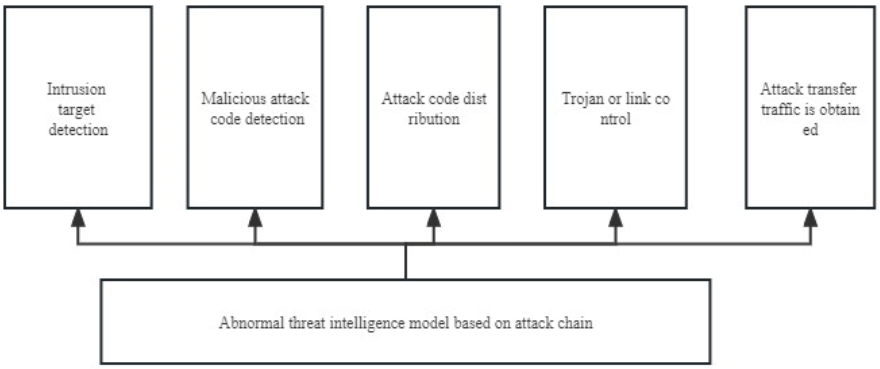


**Figure 4.** Abnormal threat intelligence analysis model based on attack chain

First and foremost, intrusion target detection is carried out using network full-flow analysis technology, encompassing port scanning and node scanning. This aims to identify external attackers and exploitable data resources, thereby completing the probe into the network system nodes.

Subsequently, malicious attack codes and their distribution are detected for the targeted network nodes. This involves detecting the malicious code developed and executed by attackers as well as identifying malicious code disseminated through web vulnerabilities or phishing techniques.

Following that, after the attacker breaches the target network, there's the detection of the utilization of Trojan programs or link controls. This includes identifying the installation of malicious software or Trojan programs and the execution of malicious codes. It also involves analyzing the execution of attack commands by monitoring the attacker's links to the target host.

Lastly, concerning each stage of the attack chain, the anomalous traffic data undergoes data attribute feature extraction and the reconstruction of the attack process. This facilitates indirect determination of the progression of network threat attacks and the degree of harm to the target network.

# References

[1] Yue, M. (2021). A survey of research on malicious document detection. Journal of Cyber Security, 6(03):54-76.

[2] Yang, S. J. (2014). Detection of malicious web pages and PDF documents based on SVM model. Technical Research, 26-45.

[3] Gu, J. X. (2021). Malicious PDF document detection based on positive sample and single classification algorithm. Technical Research, 20-40.

[4] Yuan, H. H. (2022). Research on abnormal threat detection based on network full traffic analysis technology. Technical Research, 11: 137-140.