PDF Malware Analysis
(Project Documentation)

1. Project Overview / Description

This project focuses on developing a practical toolkit and methodology to analyze malicious PDF files.

PDF malware is commonly used for phishing, exploits, JavaScript-based attacks, and embedded payloads.

will learn how attackers weaponize PDF documents and how defenders analyze and detect these malicious artifacts.

The project involves static and behavioral analysis of PDF files, extraction of embedded objects, JavaScript analysis,metadata inspection, and identifying suspicious structures used by threat actors.

2. Practical Motivation

PDF files are one of the most exploited document formats. Attackers frequently use them in:

- Phishing campaigns

- Malware delivery

- Exploit kits

- Social engineering attacks

This project gives hands-on experience with malware analysis techniques used by SOC analysts and incident responders.

They will practically inspect malicious PDFs, extract objects, review code, and generate security reports.

3. Project Objectives

1. Teach how PDF malware works in real attack scenarios.

2. Build an automated PDF analysis toolkit (static analysis).

3. Extract and analyze embedded content such as JavaScript, URLs, and objects.

4. Detect suspicious indicators like obfuscation, shellcode, and exploit attempts.

5. Produce a structured malware analysis report with findings and mitigation steps.

4. Practical Scope of the Project

The toolkit will practically perform the following checks:

A. Metadata Extraction:

- Retrieve author, creation dates, modification info.

- Identify suspicious metadata anomalies.

B. Embedded Object Extraction:

- Extract streams, images, URLs, JavaScript, and embedded files.

- Identify compressed or encrypted objects.

C. JavaScript Analysis:

- Detect suspicious JavaScript embedded in PDF.

- Identify obfuscation patterns (eval, unescape, base64 decode).

- Inspect suspicious actions triggered on document open.

D. Indicator Detection:

- Find URLs/IPs inside PDF objects.

- Detect exploit attempts targeting vulnerabilities (e.g., CVE-2010-0188).

- Identify embedded executables or shellcode.

E. Automated Risk Scoring:

- Assign severity scores based on detected artifacts.

F. Report Generation:

- Summary of findings

- Indicators of Compromise (IOCs)

- Risk assessment

- Mitigation recommendations

5. Tools & Technologies Used (Practical Tools)

Programming Languages:

- Python (recommended)

- Bash scripting (optional)

Malware Analysis Tools Will Practically Use:

- pdfid (detect keywords related to attacks)

- pdf-parser.py (deep object parsing)

- peepdf (advanced PDF analysis)

- xpdf / strings (content inspection)

- qpdf (decompression for analysis)

Additional Tools:

- VirusTotal (for hash scanning)

- Any.run or similar sandbox (optional and safe use only)

Documentation Tools:

- MS Word / Google Docs

- Draw.io for flowcharts

6. Practical Techniques Will Implement

Static Analysis Techniques:

- Parsing PDF structure

- Extracting suspicious JavaScript

- Identifying embedded objects or executables

- Finding network IOCs

- Detecting obfuscated code

Threat Hunting Techniques:

- Matching patterns with known exploits

- Identifying malicious indicators

- Chain-of-attack understanding

Blue Team Techniques:

- Recognizing malicious PDFs

- Email filtering strategies

- Hardening systems against PDF exploits

- Reporting and documenting malware analysis findings

7. Workflow / Architecture (Practical Explanation)

The toolkit workflow must follow these steps:

STEP 1: Load PDF File

- Extract metadata and structural layout.

STEP 2: Object Enumeration

- Identify number of objects.

- Highlight suspicious ones (JavaScript, embedded files).

STEP 3: Keyword-Based Detection

- Look for keywords like /JavaScript, /OpenAction, /JS, /EmbeddedFile.

STEP 4: Deep Parsing

- Extract object content.

- Analyze JavaScript, encoded streams, compressed data.

STEP 5: IOC Extraction

- URLs, IP addresses, encoded payloads.

STEP 6: Risk Analysis

- Assign severity based on findings.

STEP 7: Report Generation

- Provide clear security summary and mitigation steps.

8. Detailed Flowchart (Text Version for Word File)

START

↓

Load PDF → Extract Metadata

↓

Scan for Suspicious Keywords

↓

Enumerate Objects & JavaScript

↓

Extract and Decode Streams

↓

Identify IOCs & Exploit Indicators

↓

Risk Scoring Engine

↓

Generate Final Malware Analysis Report

↓

END

9. Expected Practical Output

must provide:

- A working PDF malware analysis toolkit.

- Real analysis done on sample malicious PDF files (safe samples only).

- Screenshots of analysis tools running.

- A structured malware analysis report.

Expected Toolkit Output Includes:

- Suspicious JavaScript detection

- Extracted URLs/IPs

- Embedded files list

- Detected exploit attempts

- Severity scoring

10. Learning Outcomes (Practical Skills)

will learn:

- How PDF malware functions internally.

- How attackers hide payloads inside PDFs.

- How defenders analyze documents safely.

- How to extract indicators used in threat intelligence.

- Report-making skills used in SOC and IR teams.

11. Student Deliverables

1. Complete project documentation (Word/PDF)

2. Automated PDF malware analysis toolkit

3. Screenshots of malware analysis performed

4. Flowcharts and working diagrams

5. Final project PPT