The beginners Guide To – Adobe PDF Malware Reverse Engineering Part 1

By BUFFERZONE Team, 8/06/2023

Share

Target: Cybersecurity specialist

Tags: Adobe PDF, Malware, Content Disarm and Reconstruction (CDR), Reverse Engineering

An Adobe PDF (Portable Document Format) file is a file format developed by Adobe Systems. It is a versatile file format that provides an easy, reliable way to present and exchange documents digitally, regardless of the software, hardware, or operating system being used by anyone who views the document [1]. According to the detected MIME type as captured in the latest Common Crawl database, PDF is the 3rd most popular file-format on the web (after HTML and XHTML); more popular than JPEG, PNG, or GIF files [2].

The popularity of Adobe PDF files among malware authors can be attributed to several factors:

1. **Universal Use**: PDF files are commonly used around the world for sharing documents. This widespread use increases the potential pool of victims for malware authors.
2. **Complexity**: The PDF format is complex and can contain several types of embedded content, including JavaScript code, images, audio, video, and other types of media. This complexity provides many opportunities for hiding malicious code
3. **Exploitable Vulnerabilities**: PDF readers, including Adobe's own Acrobat Reader, have had vulnerabilities in the past that can be exploited. For instance, a buffer overflow vulnerability in a PDF reader can potentially be exploited by a specially crafted PDF file.[3]
4. **User Trust**: Many users trust PDF files and consider them safe. This trust can be exploited by malware authors who hide their malicious code in a harmless document [1]
5. **Phishing**: PDFs can contain hyperlinks, which can be used in phishing attacks. An unsuspecting user may be tricked into clicking a link that takes them to a malicious website.[3]

These factors combined make PDF files an attractive medium for malware authors.

PDF File Format

A PDF file, often utilized across various industries due to its compatibility and versatility, is a sophisticated amalgamation of elements. The file structure is as follows:

1. **Header**: This segment encapsulates the PDF version number, acting as a concise introduction to the file's format.
2. **Body**: The body of a PDF file is a dynamic constellation of objects forming the document's substance. This may include elements such as text, images, annotations, and form fields among others, each contributing to the file's comprehensive information.
3. **Cross-reference Table**: This intricate portion of a PDF file holds crucial details regarding the indirect objects, marking the byte offset for each from the file's onset. This in-built mechanism enables PDF readers to swiftly locate, access, and render the PDF's contents.
4. **Trailer**: The trailer houses a specific pointer leading to the cross-reference table and other exceptional objects, acting as a guide to navigate through the document's structure.

Beyond these primary elements, a PDF file may be enriched with further components like metadata, interactive aspects (such as hyperlinks and form fields), and security settings. Notably, a PDF file's capacity to encapsulate rich media content—encompassing graphics, fonts, text, and interactive elements like buttons and forms—is what sets it apart. Regardless of the device or software deployed for viewing, this ensures uniform display of the document.

The remarkable flexibility offered by the PDF file format has made it a universally adopted standard for document sharing and archiving, underlining its wide-ranging adaptability across diverse platforms. A comprehensive description of the PDF structure can be found here and contains advanced information about the PDF structure [3].

Malware Investigation Research Steps:

Investigating PDF malware requires a careful and systematic approach. Below are highly suggested steps we conduct in our research:

1. **Isolation**: Always work in a safe environment when dealing with potential malware. This usually means using a sandbox or a dedicated, isolated system that is not connected to your network. In this blog we will work inside Ubuntu Virtual Machine.
2. **Collection**: The first step is gathering potentially malicious PDF files. These can be sourced from various locations like spam emails, suspicious websites, or shared through threat intelligence feeds. We will use MalwareBazaar [4] a public malware repository to receive interesting malware for analysis.
3. **Static Analysis**: Start by examining the PDF without executing it. This includes viewing the file metadata, the structure, the embedded objects, scripts, or unusual elements. Tools like PDFiD, PDF-Parser and Pdfalyze are great

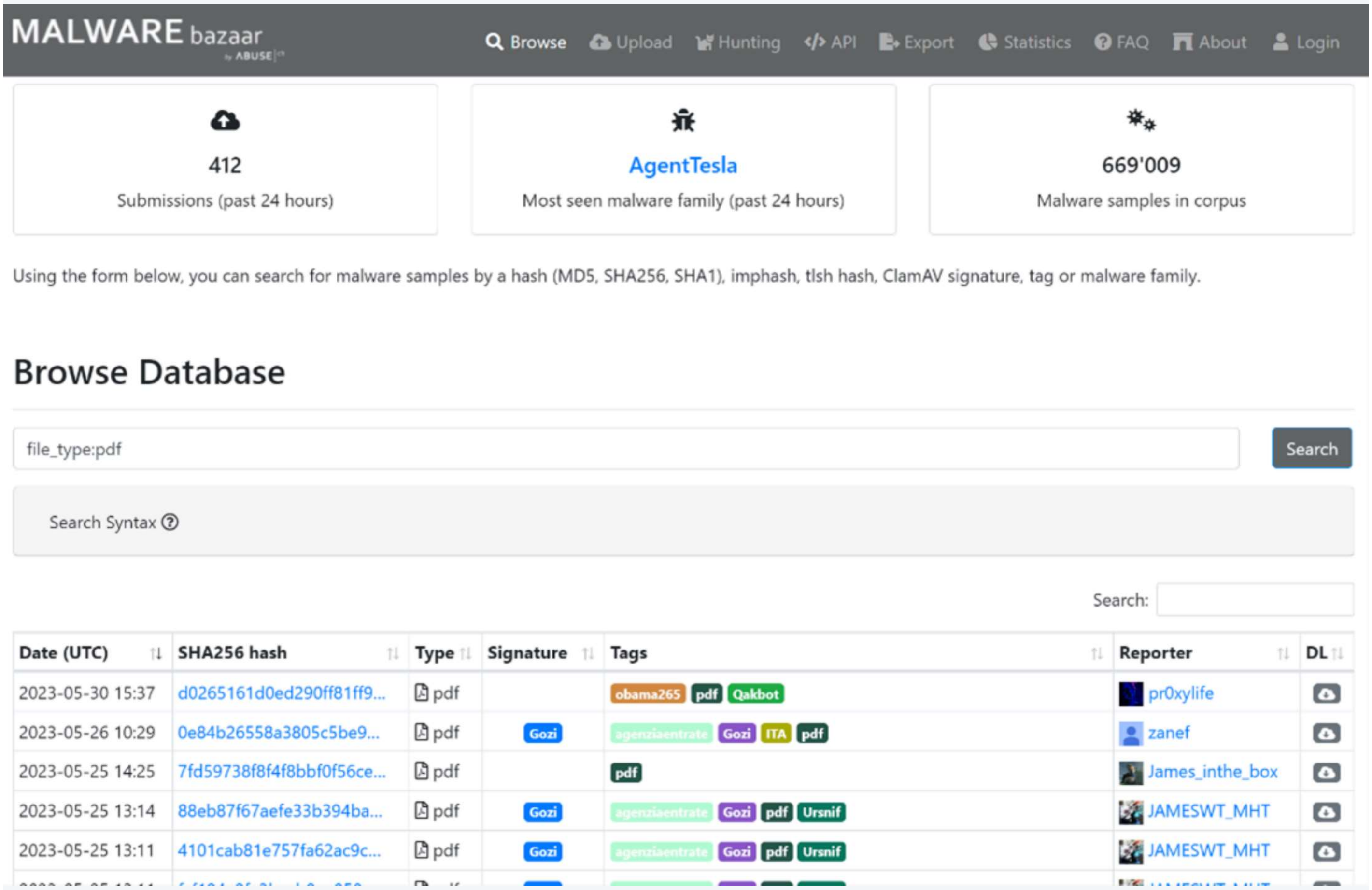
sources for static analysis information. This is the focus of our blog.

4. **Dynamic Analysis**: This involves monitoring the behavior of the PDF file when it is opened. You would typically use a sandbox environment for this, which can safely log the actions of the file, such as network connections, file system modifications, or registry changes. Many evasive behaviors are discovered during dynamic analysis that can highlight behavior that we missed during the static analysis, or we are unfamiliar with. This part will be outside of this blog's focus.
5. **Payload Extraction**: If the PDF has an embedded payload, this will need to be extracted for further analysis. This could be another file, a script, or something else. Payload extraction can be done as part of the static analysis or part of the dynamic analysis features.
6. **Code Analysis**: If the PDF includes embedded or obfuscated code, such as JavaScript or shellcode, this will need to be analyzed. This involves de-obfuscating the code, understanding its functionality, and identifying any potential exploits or vulnerabilities it might use. This will be done as part of the static analysis investigation we will conduct.
7. **Threat Intelligence Correlation**: Correlate the information collected about the PDF malware with threat intelligence data. This can give information on the possible threat actors, campaigns, their methods, or whether this malware has been observed before. This step is done after the collection and during the static and dynamic analysis. When we discover Information of Compromise (IOC) which are a list of drop file (sha256 /MD5 hash representation), URL's, IP addressed in the file we can enhance our understanding of the file capabilities based on threat intelligence.
8. **Reporting**: Finally, document your findings. This report should detail the characteristics of the malware, how it works, its impact, and recommended mitigation strategies.

Remember to always stay safe when investigating potential malware, and only do so in a controlled and isolated environment. It is important to keep systems and software up to date to protect against known vulnerabilities that malware often exploits. This tutorial is for educational purposes only. Please take full responsibility while handling dangerous malicious files.

Collection:

In this blog, we will fetch a potentially suspicious file from MalwareBazaar and examine the PDF file jointly (remember to operate within a virtual machine). Utilizing the "file_type:pdf" filter, we will obtain the most recently uploaded PDF files within the framework. Let us proceed with the download of the latest file, with the sha256 hash: d0265161d0ed290ff81ff99e4571de9b709b357c9e663ad2b4519b68497705f5.



412
Submissions (past 24 hours)

AgentTesla
Most seen malware family (past 24 hours)

669'009
Malware samples in corpus

Using the form below, you can search for malware samples by a hash (MD5, SHA256, SHA1), imphash, tlsh hash, ClamAV signature, tag or malware family.

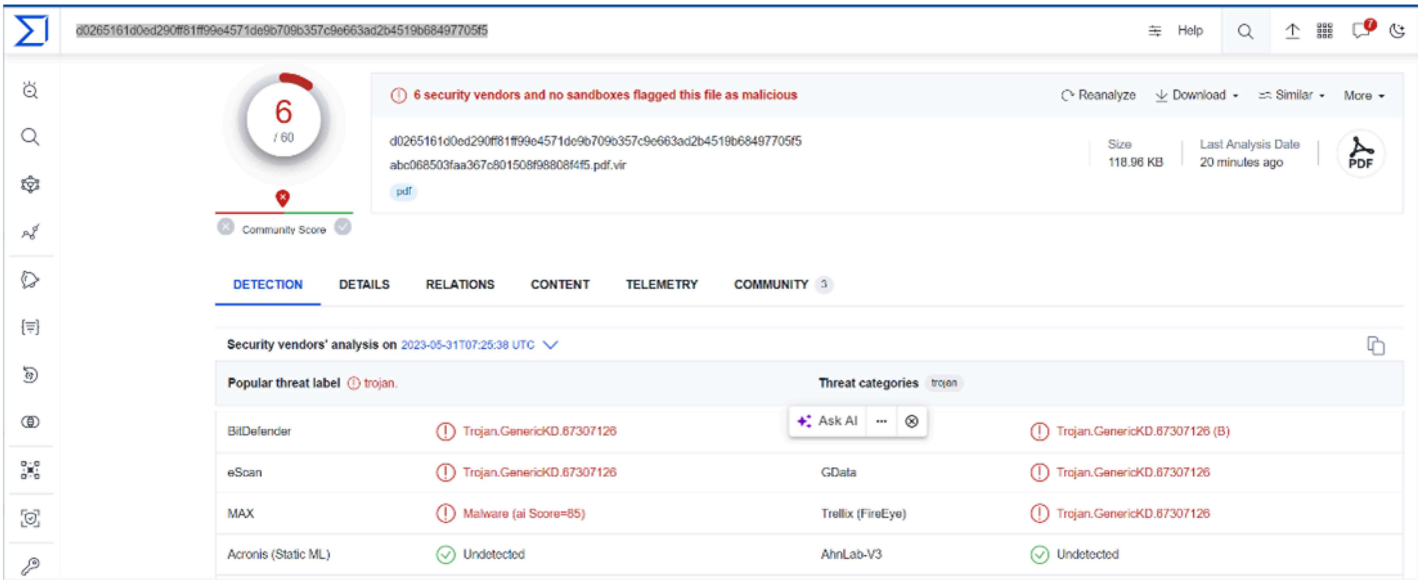
Browse Database

file_type:pdf

Search:

Date (UTC)	SHA256 hash	Type	Signature	Tags	Reporter	DL
2023-05-30 15:37	d0265161d0ed290ff81ff9...	pdf		obama265 pdf Qakbot	pr0xylife	
2023-05-26 10:29	0e84b26558a3805c5be9...	pdf	Gozi	agenziaentrate Gozi ITA pdf	zanef	
2023-05-25 14:25	7fd59738f8f4f8bbf0f56ce...	pdf		pdf	James_inthe_box	
2023-05-25 13:14	88eb87f67aefe33b394ba...	pdf	Gozi	agenziaentrate Gozi pdf Ursnif	JAMESWT_MHT	
2023-05-25 13:11	4101cab81e757fa62ac9c...	pdf	Gozi	agenziaentrate Gozi pdf Ursnif	JAMESWT_MHT	

This file has a low detection rate now in VirusTotal [7]:



d0265161d0ed290ff81ff99e4571de9b709b357c9e663ad2b4519b68497705f5

6 / 60

Community Score

6 security vendors and no sandboxes flagged this file as malicious

d0265161d0ed290ff81ff99e4571de9b709b357c9e663ad2b4519b68497705f5
abc068503faa367c801508f98808f4f5.pdf.vir
pdf

Size 118.96 KB

Last Analysis Date 20 minutes ago

Security vendors' analysis on 2023-05-31T07:25:38 UTC

Popular threat label trojan.

Threat categories trojan

BitDefender	Trojan.GenericKD.67307126		Trojan.GenericKD.67307126 (B)
eScan	Trojan.GenericKD.67307126	GData	Trojan.GenericKD.67307126
MAX	Malware (ai Score=85)	Trellix (FireEye)	Trojan.GenericKD.67307126
Acronis (Static ML)	Undetected	AhnLab-V3	Undetected

This indicates that the file is a new attack or could be a false positive. To verify we will start our static analysis.

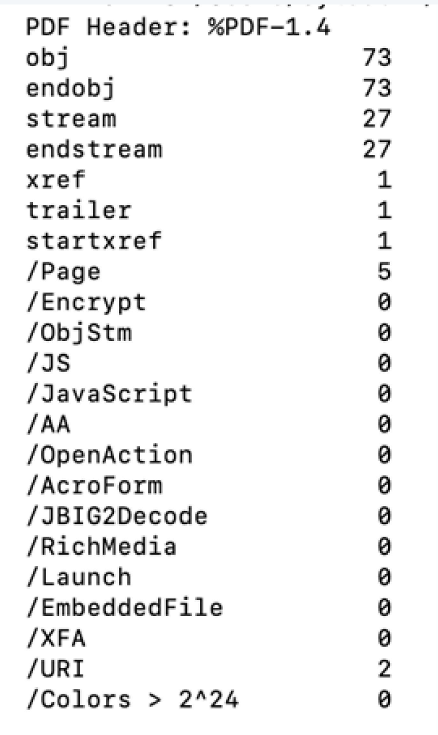
Reverse Engineering PDF File Using Static Analysis

In this blog we will focus on PDFiD and PDF-Parser from Didier Stevens [5] that created a well-known suite of research tools. We encourage you to follow him and check his website and tutorials. Furthermore, we will use a new tool called Pdfalyzer [6] that automates PDF research process and provides highly detailed and visible information.

PDFiD

PDFiD will scan a file to look for certain PDF keywords, allowing you to identify PDF documents that contain JavaScript or execute an action when opened for example. PDFiD will also handle name obfuscation. However, PDFiD is not a detection tool it is a visibility tool, and it is a great first step to start our investigation.

By running python pdfid.py <file> we will get the following output:



```
PDF Header: %PDF-1.4
obj                    73
endobj                 73
stream                 27
endstream              27
xref                    1
trailer                 1
startxref               1
/Page                   5
/Encrypt                0
/ObjStm                 0
/JS                     0
/JavaScript             0
/AA                     0
/OpenAction             0
/AcroForm               0
/JBIG2Decode            0
/RichMedia              0
/Launch                 0
/EmbeddedFile           0
/XFA                    0
/URI                    2
/Colors > 2^24          0
```

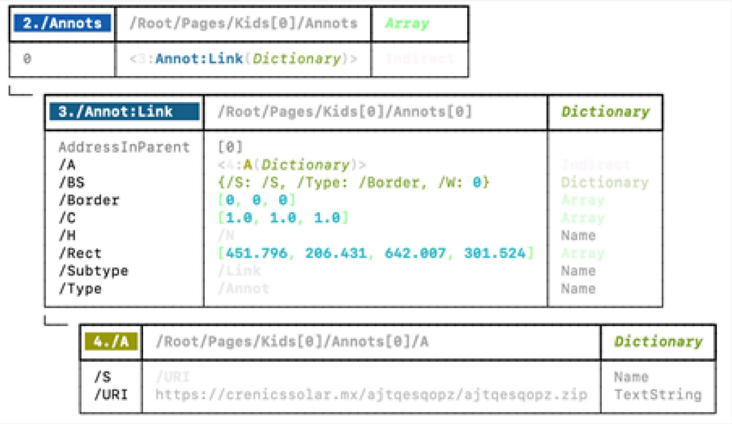
We can learn that there are 73 objects, 27 streams, 5 pages and 2 URI. Since URI are common in PDF files we will drill down on the objects and investigate further.

Pdfalyze

Pdfalyze is a new tool in the research community and has a lot of value. It visualizes the PDF file structure and using Yara signatures and many more forensic capabilities.

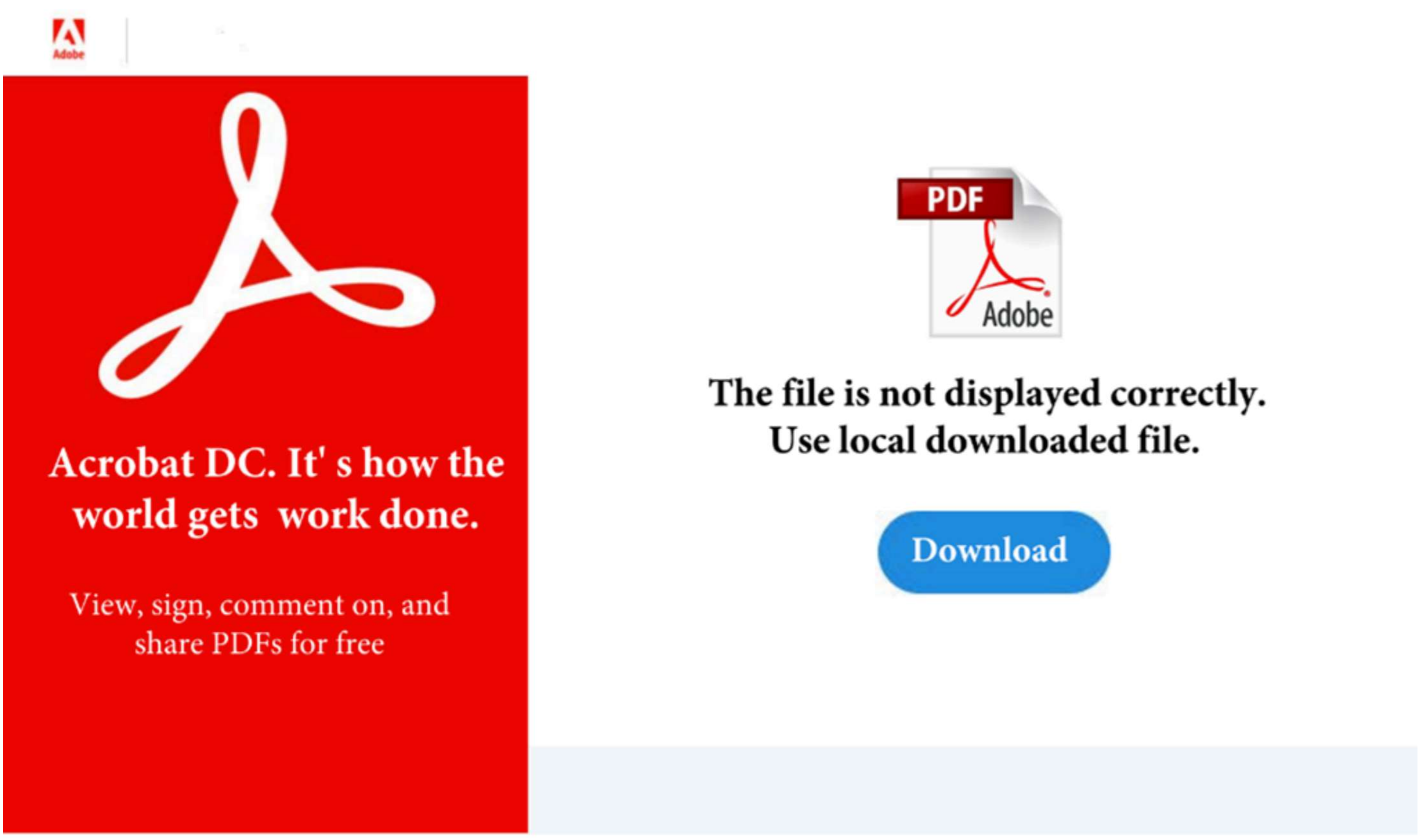
By running Pdfalyze <file>

We will get the tree of the file structure and since we have multiple objects and streams, we will start the investigation with the URI.



```
┌──────────────┬────────────────────────────────┬──────────┐
│ 2./Annots    │ /Root/Pages/Kids[0]/Annots     │ Array    │
├──────────────┼────────────────────────────────┼──────────┤
│ 0            │ <3:Annot:Link(Dictionary)>     │ Indirect │
└──────────────┴────────────────────────────────┴──────────┘
└──
    ┌──────────────────┬────────────────────────────────────────────┬────────────┐
    │ 3./Annot:Link    │ /Root/Pages/Kids[0]/Annots[0]              │ Dictionary │
    ├──────────────────┼────────────────────────────────────────────┼────────────┤
    │ AddressInParent  │ [0]                                        │            │
    │ /A               │ <4:A(Dictionary)>                          │ Indirect   │
    │ /BS              │ {/S: /S, /Type: /Border, /W: 0}            │ Dictionary │
    │ /Border          │ [0, 0, 0]                                  │ Array      │
    │ /C               │ [1.0, 1.0, 1.0]                            │ Array      │
    │ /H               │ /N                                         │ Name       │
    │ /Rect            │ [451.796, 206.431, 642.007, 301.524]       │ Array      │
    │ /Subtype         │ /Link                                      │ Name       │
    │ /Type            │ /Annot                                     │ Name       │
    └──────────────────┴────────────────────────────────────────────┴────────────┘
    └──
        ┌────────┬────────────────────────────────────────────────────┬────────────┐
        │ 4./A   │ /Root/Pages/Kids[0]/Annots[0]/A                    │ Dictionary │
        ├────────┼────────────────────────────────────────────────────┼────────────┤
        │ /S     │ /URI                                               │ Name       │
        │ /URI   │ https://crenicssolar.mx/ajtqesqopz/ajtqesqopz.zip  │ TextString │
        └────────┴────────────────────────────────────────────────────┴────────────┘
```

We can observe that in object 2 we have /Annots stream that contains a link. This link is downloaded when the user clicks on the button in the document. Although this file is new from today the link is already down by the hosting site so we cannot download the zip file for investigation, but it could be a malicious object. To lure the user to download the file it uses a lure image:


Adobe
Acrobat DC. It' s how the world gets work done.
View, sign, comment on, and share PDFs for free
PDF
Adobe
The file is not displayed correctly. Use local downloaded file.
Download

This is a classical attack vector in PDF files. The user is not aware of what content is hidden behind the button.

Pdf-parser.py

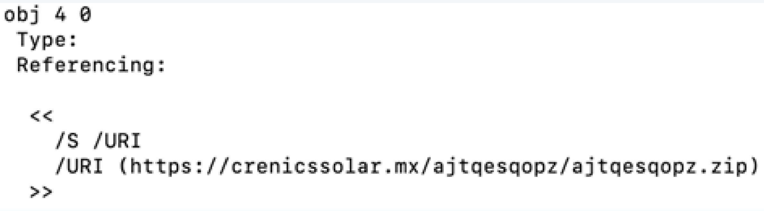
It is a simple and efficient tool for object extraction from PDF files.

Although in this case Pdfalyze showed us the object we wanted. For some use cases it is simpler to use pdf-parser.

The usage is as follows:

python pdf-parser.py -f -o 4 -d extract_URLI <file>

Were the object we extract is 4 and we saved the object if it exists in extract_URL as a file.



```
obj 4 0
 Type:
 Referencing:

  <<
    /S /URI
    /URI (https://crenicssolar.mx/ajtqesqopz/ajtqesqopz.zip)
  >>
```

We trust you found the novice's PDF guide Part-1 informative, and we will soon release Part-2. Kindly visit our website for upcoming blog entries.

References

[1] Adobe PDF, https://www.adobe.com/acrobat/about-adobe-pdf.html

[2] Common Crawl data statistics, https://commoncrawl.github.io/cc-crawl-statistics/plots/mimetypes.

[3] Dubin, Ran. "Content Disarm and Reconstruction of PDF Files." *IEEE Access* (2023).

[4] MalwareBazaar, Public Malware Repository, https://bazaar.abuse.ch/

[5] Didier Stevens, PDF tools, https://blog.didierstevens.com/programs/pdf-tools/

[6] Pdfalyzer,https://github.com/michelcrypt4d4mus/pdfalyzer

[7]
VirusTotal, https://www.virustotal.com/gui/file/d0265161d0ed290ff81ff99e4571de9b709b357c9e663ad2b4519b68497705f5[8]
Yara, https://virustotal.github.io/yara/