Lab 6 – Static Malware Analysis

Shrutika Joshi

University of Maryland Baltimore County

Presented To – Gina Marie

Date – 15th July 2023

Week 6 Discussion -1:

Static Malware Analysis Plan

What is malware? - Malware is any type of software that does something that causes detriment to the user, computer, or network— This malware could be any type such as viruses, trojan horses, worms, rootkits, scareware, and spyware.

What is static malware analysis? – Examining the malware file or programs without running the suspicious programs or files detected during analysis. This is the safest way to analyze files without executing code that could infect your system. In this process, the metadata of the file can be checked such as file name, type, size, and MD5 hash. Different tool can be used to analyze files statically. [1]

Initial Assessment – During the initial assessment of malware, After receiving a call from an infecting computer, at first incident gets documented such as date, time, behavior, relevant system information such as operating system, hardware configuration, error messages if there are

any, installed software, any recent user activity, potential sources of information or indicators of compromise.

Containment –

Isolate the infected computer from the network to prevent further spread of the malware and minimize potential damage.

Disconnect the computer from the internet and disable any wireless or Bluetooth connections. Change passwords of all accounts which seem to be compromised.

Static Analysis –

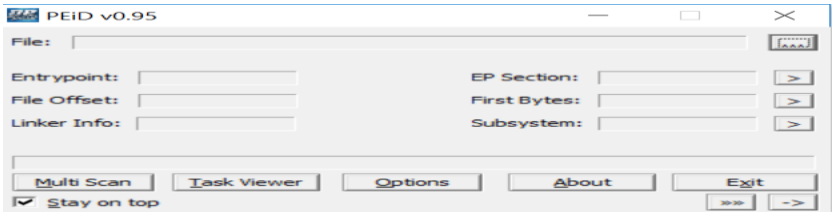
During this phase, we will analyze all the indicators of compromise such as files, URLs, IP addresses, and domains detected during the initial assessment.

For that we will collect all suspicious files or artifacts from infected computer into malware analysis VM such as Flare-VM, and REMnux we are specifically designed for malware analysis having all built-in tools to analyze malicious files.

Perform preliminary analysis of a file using scanners such as Virustotal, Hybrid-Analysis, URLscan.io to check the malicious ratings, MD5, SHA256 hash, and more details like which type of malware it is, whether it is detected by any tools.

SysinternalsSuite can be used to extract strings from executable files using the command line to further analyze the string file for suspicious strings.

You can check the file type using the tool **PEID** and can also disassemble the file using the PEID disassembler to check the programming language and source code of a malicious file.



Analyzing PE header - The programs that we run are generally stored in the executable file format. These files are portable because they can be taken to any system with the same Operating System and dependencies, and they will perform the same task on that system. Therefore, these files are called Portable Executables (PE files). [3]

To perform file-level analysis on this PE header file, you can use tools such as such as **PEView, PEStudio, and BinText**. Out of this PEStudio is a great tool to perform PE header analysis used by researchers to perform malware analysis as it shows details like the encoding, size of the string, hash details, offset in the binary where the string was found, header information and a hint to guess what the string is related to. It also has a column for a blacklist, which matches the strings against some signatures.

Also can check API calls made by malicious executable and can further check what functions these API calls can perform like CreateProcess, ShellExecute, RegCreateKey, DeleteFile which can be used by malware to create a registry, execute shell commands, processes and delete files from the system.

Citation -

1. N-able. (2019, August 13). Malware Analysis Steps. Retrieved from https://www.n-able.com/blog/malware-analysis-steps

2. Security Ninja. (2015, April 29). Malware Analysis Basics: Static Analysis. Retrieved from https://resources.infosecinstitute.com/topic/malware-analysis-basics-static-analysis/

3. Balaji, P. (2020, June 5). PEStudio: Initial Malware Assessment Made Simple. SOC Investigation. Retrieved from https://www.socinvestigation.com/pestudio-initial-malware-assessment-made-simple/

Introduction – Perform static analysis on suspicious executable files and documents which are detected during the initial assessment of an incident to check whether it is malicious or not. This analysis should be performed statically means without running the executable or opening the document files.

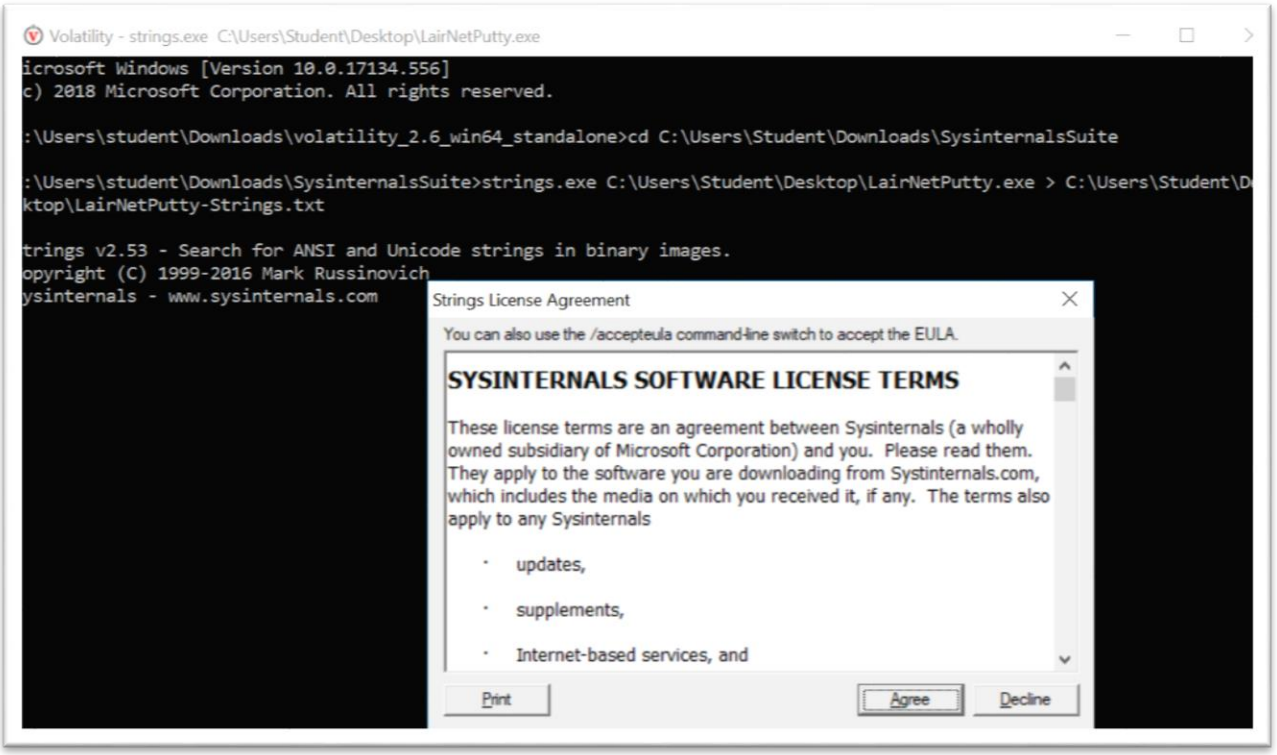
Pre-Lab – For this lab, we will be using tools like PEID, PEstudio, windows command line to run commands, Systinternalsuit tools to extract strings from a suspicious file using command line for further analysis and Notepad++ to view the strings of a file.

Analysis –

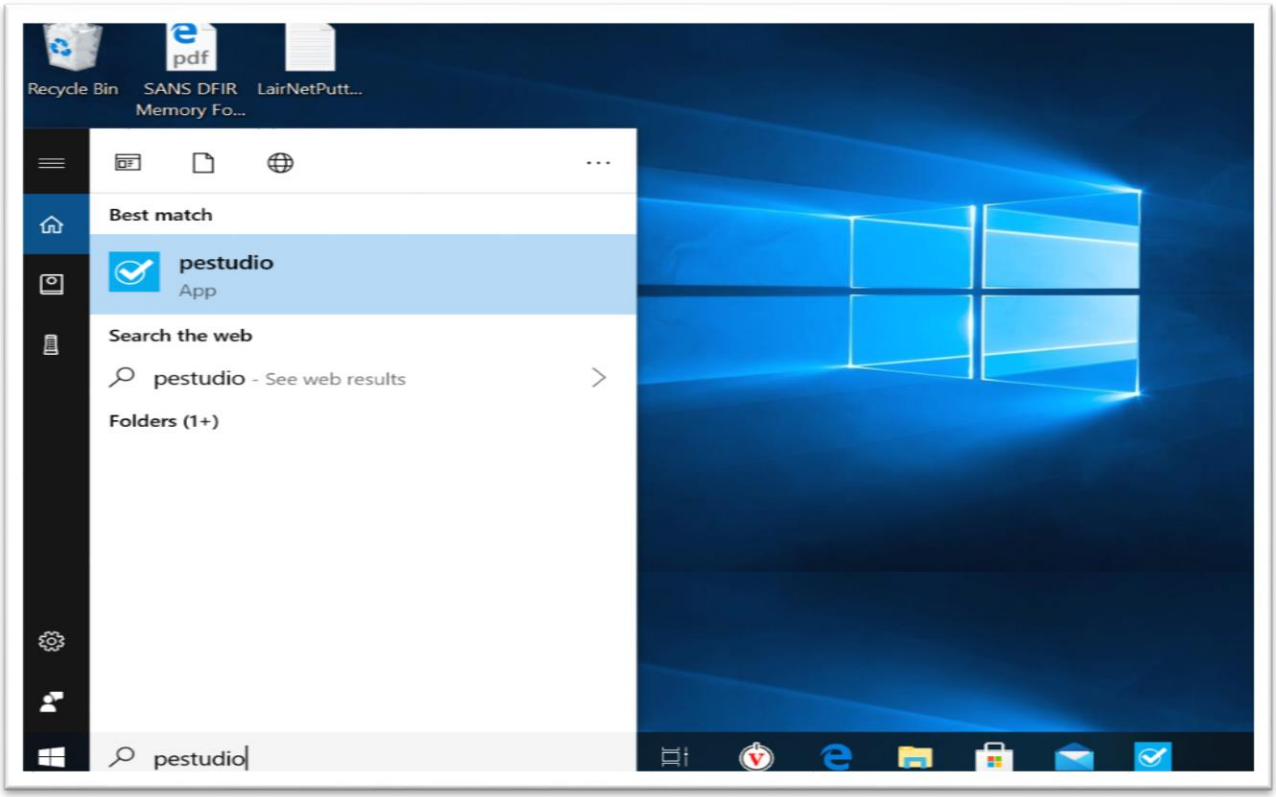
1. Analysis of Suspicious Executable

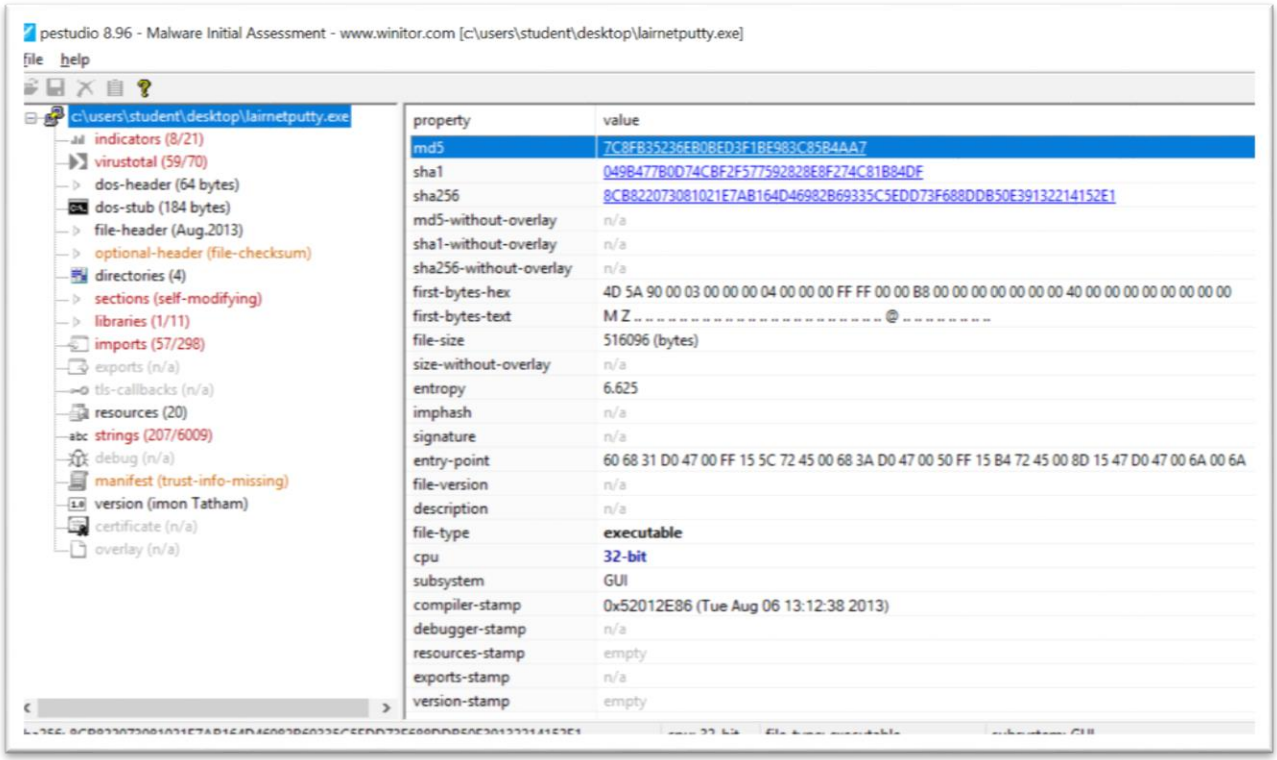
- To fetch strings from a suspicious file first open a command prompt and change the directory to C:\Users\Student\Downloads\SysinternalsSuite
- Run the below command to fetch the strings from the executable file 'LairNetPutty.exe' and add string output to a file 'C:\Users\Student\Desktop\LairNetPutty-Strings.txt"'. Then click Agree, and the string output will save to a file on the provided path. Executing the following command will perform a basic string search in binary.

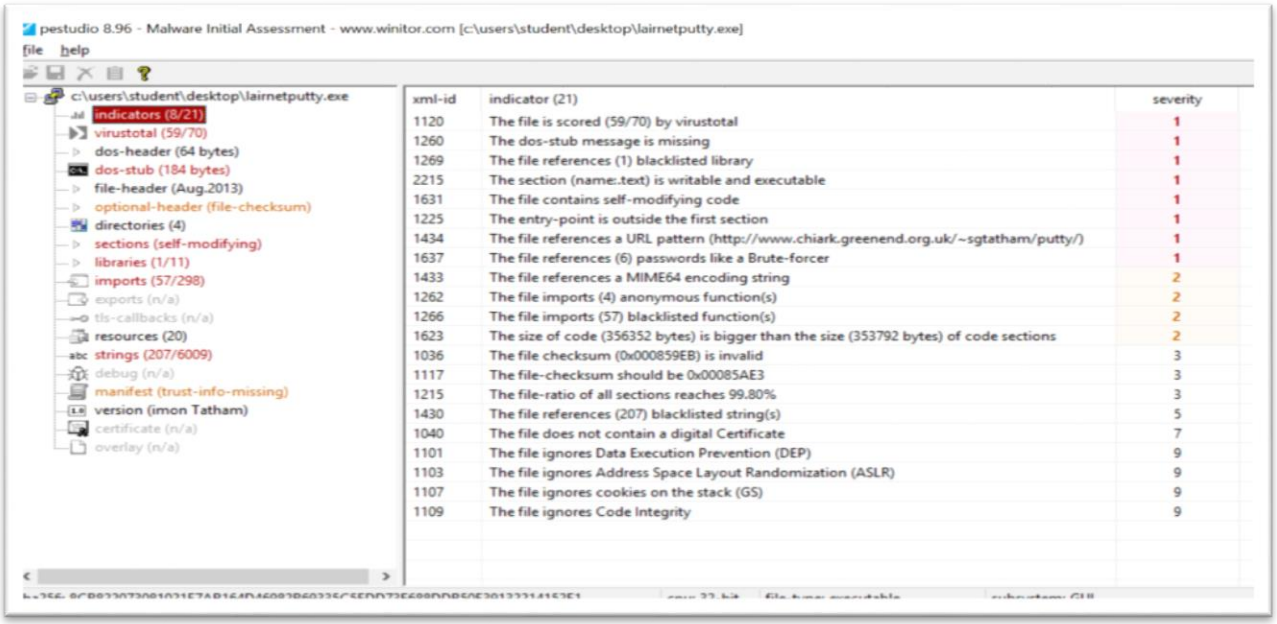
  Command - "strings.exe C:\Users\Student\Desktop\LairNetPutty.exe > C:\Users\Student\Desktop\LairNetPutty-Strings.txt"



- Now I have opened the file in Notepad++ to view the string output of the file LairNetPutty-Strings.txt.
- The alternative way of doing static analysis on file is using the PEStudio tool to view the executable file strings
- PEStudio is a popular and powerful software analysis tool that is used for statically analyzing malware and it is widely used by security researchers and malware analysts.
- Open the PEStudio from the start menu
- This tool shows details like the encoding, size of the string, hash details, offset in the binary where the string was found, header information and a hint to guess what the string is related to. It also has a column for a blacklist, which matches the strings against some signatures.


Recycle Bin
SANS DFIR Memory Fo...
LairNetPutt...
Best match
pestudio
App
Search the web
pestudio - See web results
Folders (1+)
pestudio


pestudio 8.96 - Malware Initial Assessment - www.winitor.com [c:\users\student\desktop\lairnetputty.exe]
file help
c:\users\student\desktop\lairnetputty.exe
indicators (8/21)
virustotal (59/70)
dos-header (64 bytes)
dos-stub (184 bytes)
file-header (Aug.2013)
optional-header (file-checksum)
directories (4)
sections (self-modifying)
libraries (1/11)
imports (57/298)
exports (n/a)
tls-callbacks (n/a)
resources (20)
strings (207/6009)
debug (n/a)
manifest (trust-info-missing)
version (imon Tatham)
certificate (n/a)
overlay (n/a)
property
value
md5
7C8FB35236EB0BED3F1BE983C85B4AA7
sha1
049B477B0D74CBF2F577592828E8F274C81B84DF
sha256
8CB822073081021E7AB164D46982B69335C5EDD73F688DDB50E39132214152E1
md5-without-overlay
n/a
sha1-without-overlay
n/a
sha256-without-overlay
n/a
first-bytes-hex
4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00
first-bytes-text
M Z @
file-size
516096 (bytes)
size-without-overlay
n/a
entropy
6.625
imphash
n/a
signature
n/a
entry-point
60 68 31 D0 47 00 FF 15 5C 72 45 00 68 3A D0 47 00 50 FF 15 B4 72 45 00 8D 15 47 D0 47 00 6A 00 6A
file-version
n/a
description
n/a
file-type
executable
cpu
32-bit
subsystem
GUI
compiler-stamp
0x52012E86 (Tue Aug 06 13:12:38 2013)
debugger-stamp
n/a
resources-stamp
empty
exports-stamp
n/a
version-stamp
empty

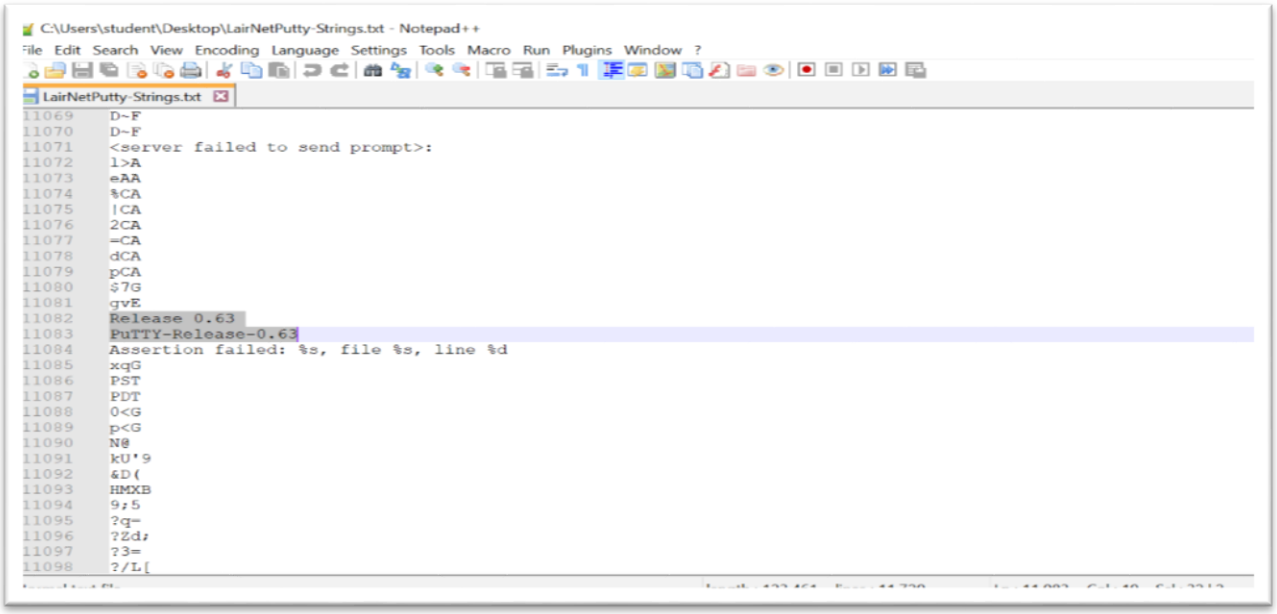


- **Review the results of running strings to answer the following:**

1. What release version of Putty does this executable contain?

- For this first open string file we previously extracted using Notepad++.

  The putty release version is 0.63



2. What email addresses are contained in relation to an encryption cipher?

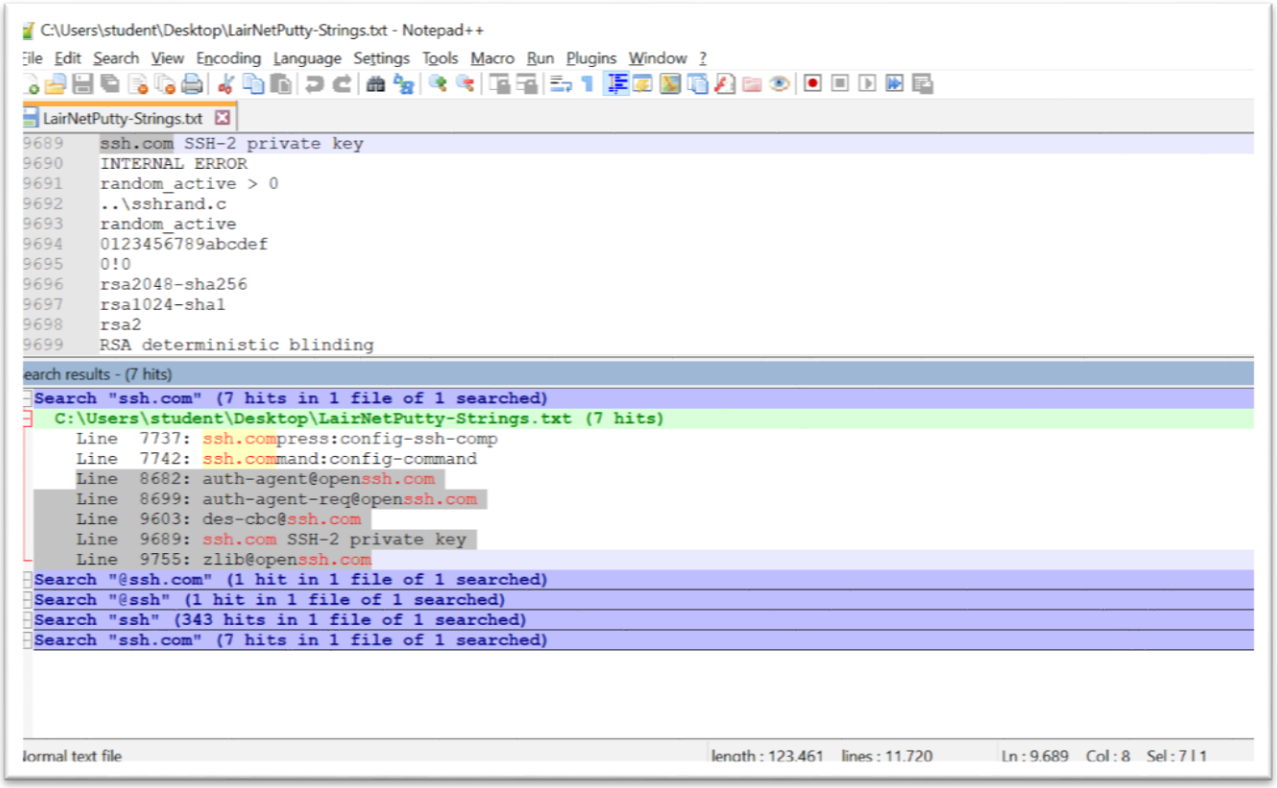
- There are 4 email addresses that are related to encryption cipher.

  auth-agent@openssh.com

  auth-agent-req@openssh.com

  des-cbc@ssh.com

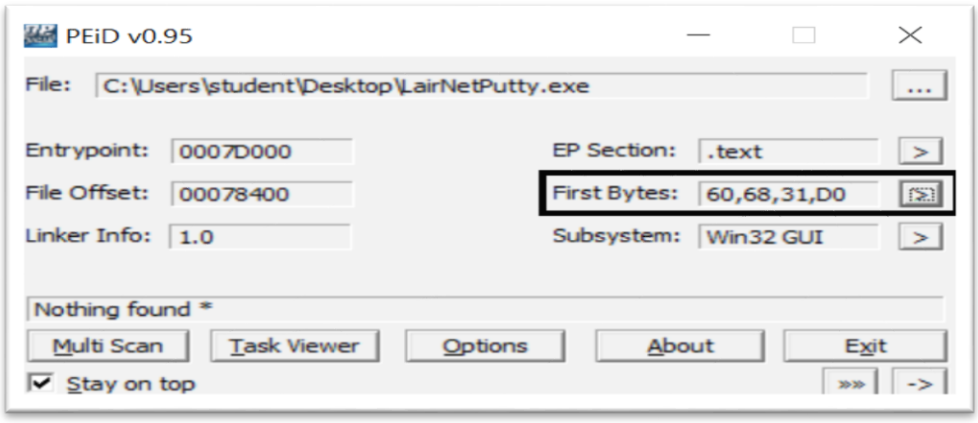
  zlib@openssh.com



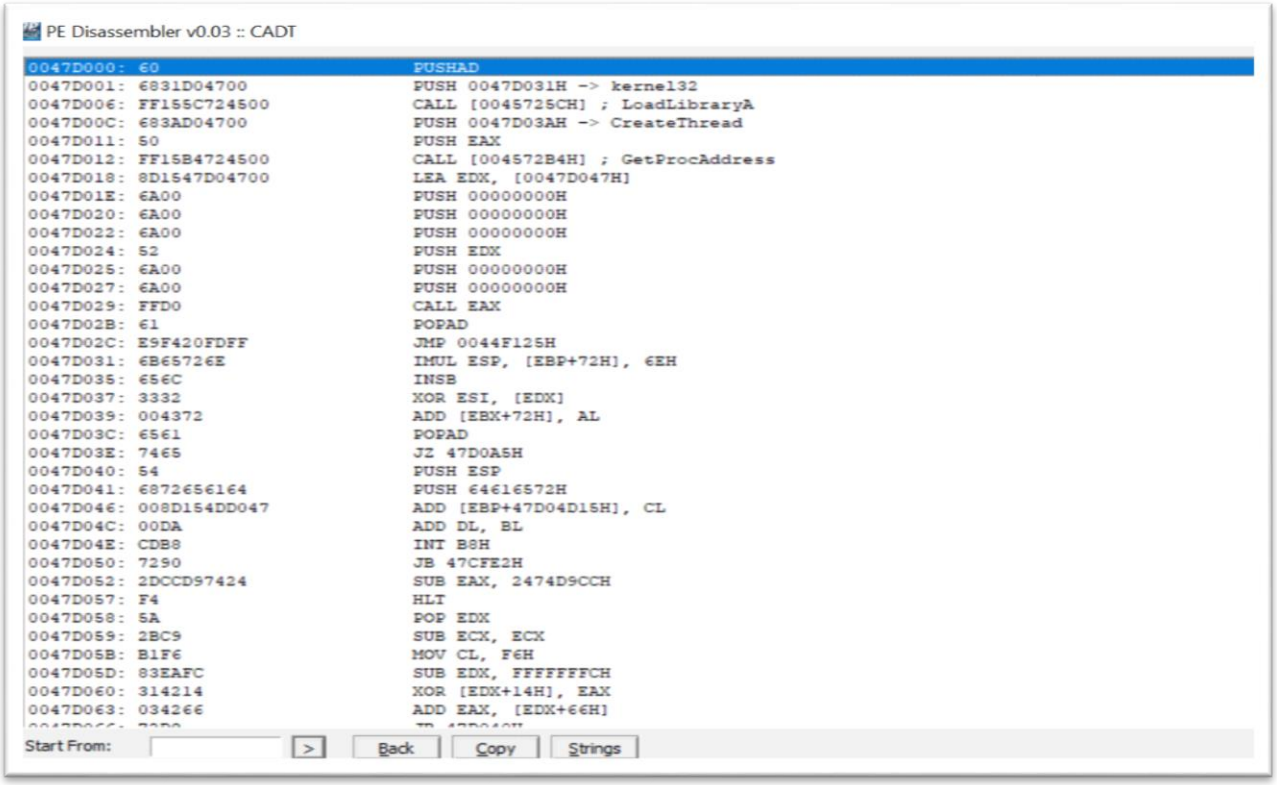
3. What programming language does this executable appear to be based on?

- I have used the PEiD tool's disassembler option to disassemble the source code of the executable file.

- For that open the PEiD tool from the start menu and browse the LairNetPutty.exe file. It will show the File Offset, Entrypoint, and First Bytes of the file. Now further expand the First Bytes option to view the source code of a file.



- Below is a source code of a file that is disassembled by the PE disassembler. By looking at the code below it looks like assembly language.



4. What are some function names used within the executable? Do any appear to be malicious?

- Below are some of the functions which are used within an executable file that hackers can use for malicious reasons

  CreateProcessA: This function is used to create a new process. Malware can abuse this function to execute additional malicious processes or launch other files.

  CreateThread: Malware often uses this function to create new threads, allowing for concurrent execution and potential evasion techniques.

  RegCreateKeyA: This function is used to create a registry key. Malware can utilize this function to create or modify registry entries to achieve persistence or make system-level changes.

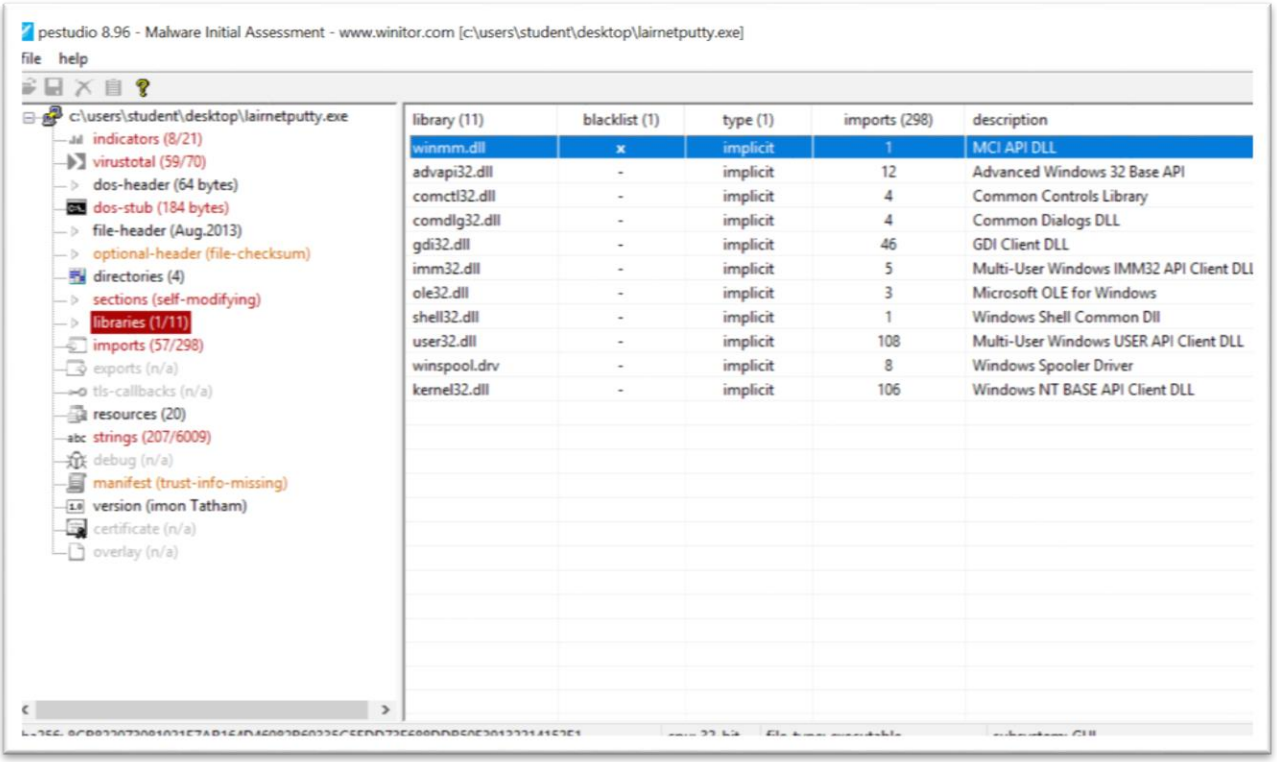
  ShellExecuteA: This function is used to execute commands or open files using the default associated program. Malware can exploit this function to execute arbitrary commands or launch malicious files.

  DeleteFileA: Malware can use this function to delete files from the system, potentially removing critical system files or user data.

5. What are some Windows DLLs that are referenced?

- DLL stands for Dynamic Link Library. The Dynamic Link Library file contains instructions and rules that other programs on a computer or device use to run and function efficiently. [1]
- The below snapshot contains .DLL files detected in the executable file LairNetPutty.exe. Out of which winmm.dll is detected as blacklisted by PEStudio.
- The Winmm.dll (Windows Multimedia API) is a dynamic link library file in the Windows operating system. It Provides functions for controlling multimedia devices and playing audio

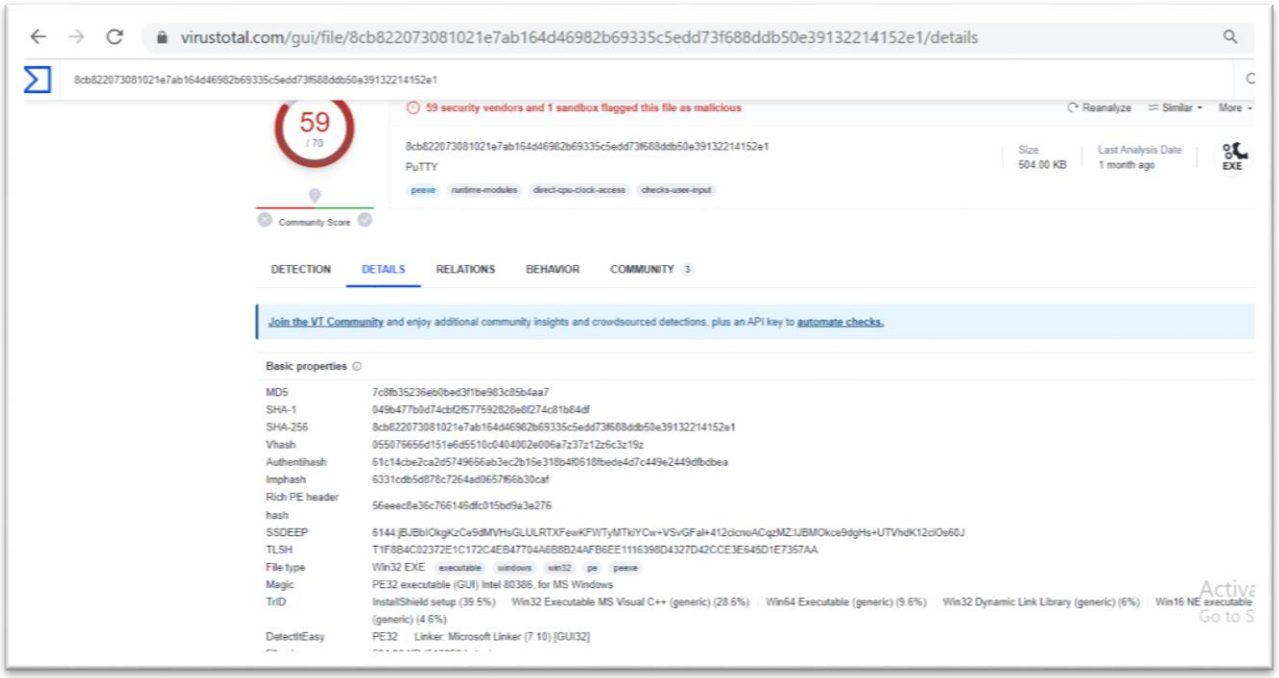
and video files using the Media Control Interface (MCI) in Windows. While the "winmm.dll" file itself is a legitimate component of the Windows operating system it can be used for malware persistence by injecting code into the memory space of ‘winmm.dll’. Also if there are any security vulnerabilities associated with this file, attackers can exploit it. It is also called DLL hijacking. This type of attack can be used for data exfiltration, privilege escalation, and establishing persistence on an account which makes it a serious threat to organizations.



- Now upload a hash of the LairNetPutty.exe file to VirusTotal.com to answer the following:

1. Does the executable appear to be malicious?

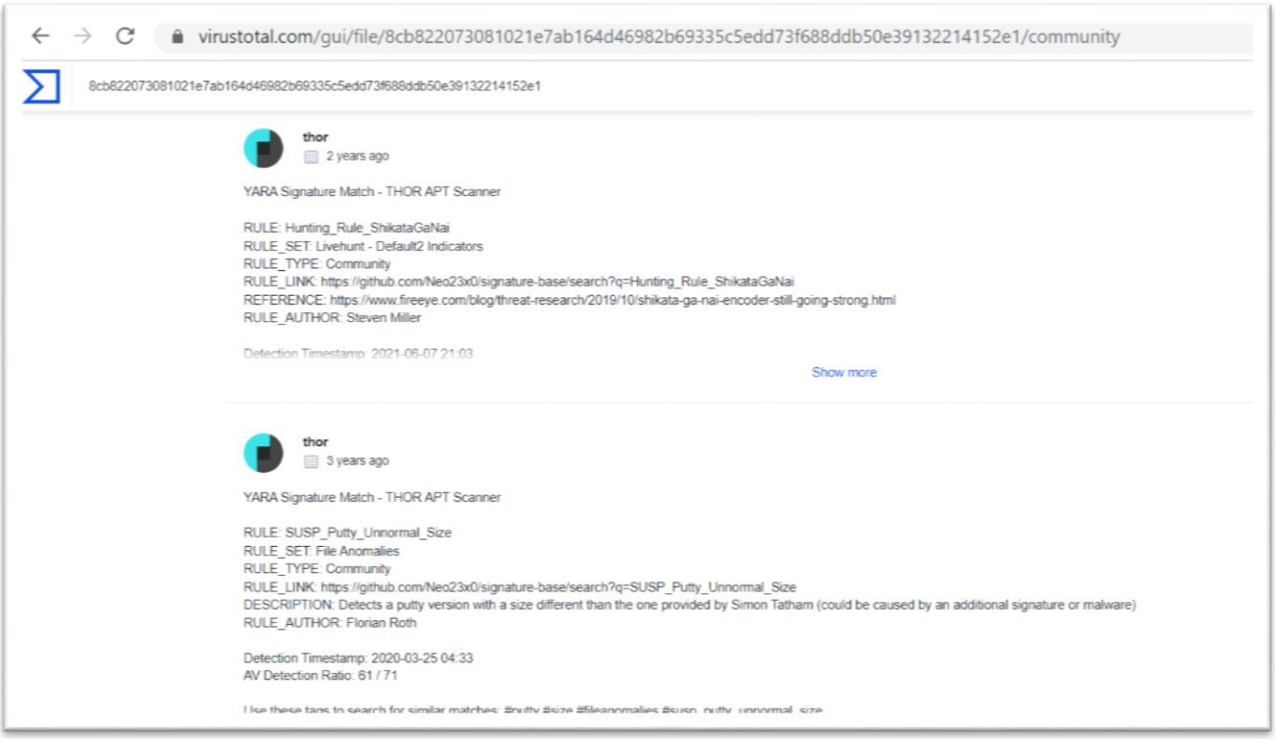
- Yes, as per the VirusTotal result, the executable appears to be malicious. A total of 59 out of 70 security vendors detected this executable as malicious and it is detected as a Trojan name trojan.rozena/swrort.



2. Based on the analysis, what does it appear this executable appears to be?

- Based on the analysis this executable appears to be a Trojan name ‘trojan.rozena/swrort’. trojan.rozena/swrort is a Trojan which is a detection for files that try to connect to a remote server. Once connected, an attacker can perform malicious routines such as downloading other files. They can be installed from a malicious site or used as payloads of exploit files. And once executed it connects to a remote server and attacker can perform malicious routines such as downloading other malware and executing them.
- As per the YARA signature match it is also detected as a ‘THOR APT scanner’. [2]

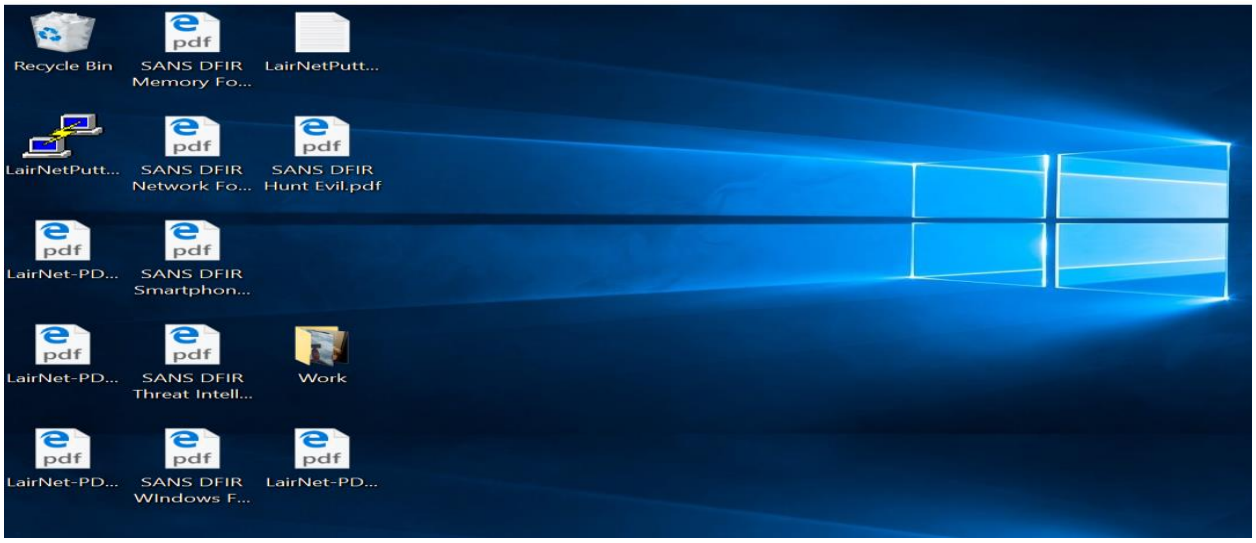
- YARA Signature – YARA is a tool designed to help malware researchers identify and classify malware samples. This tool is used to perform signature-based detection of a malware. [5]



3. Analysis of Suspicious PDFs

1. On your desktop, there are multiple PDF documents that are identified as "questionable".

- Just by looking at the filenames we can’t be sure which files could be malicious. Although, Putty application PDF files could be malicious as well as have to check SANS files.



2. Using PDFStreamDumper (found on your Start Menu), analyze the different files to identify which may be malicious and list them.

- By looking at the files below files seems suspicious, although detailed analysis of every object should be performed to make the decision on this.

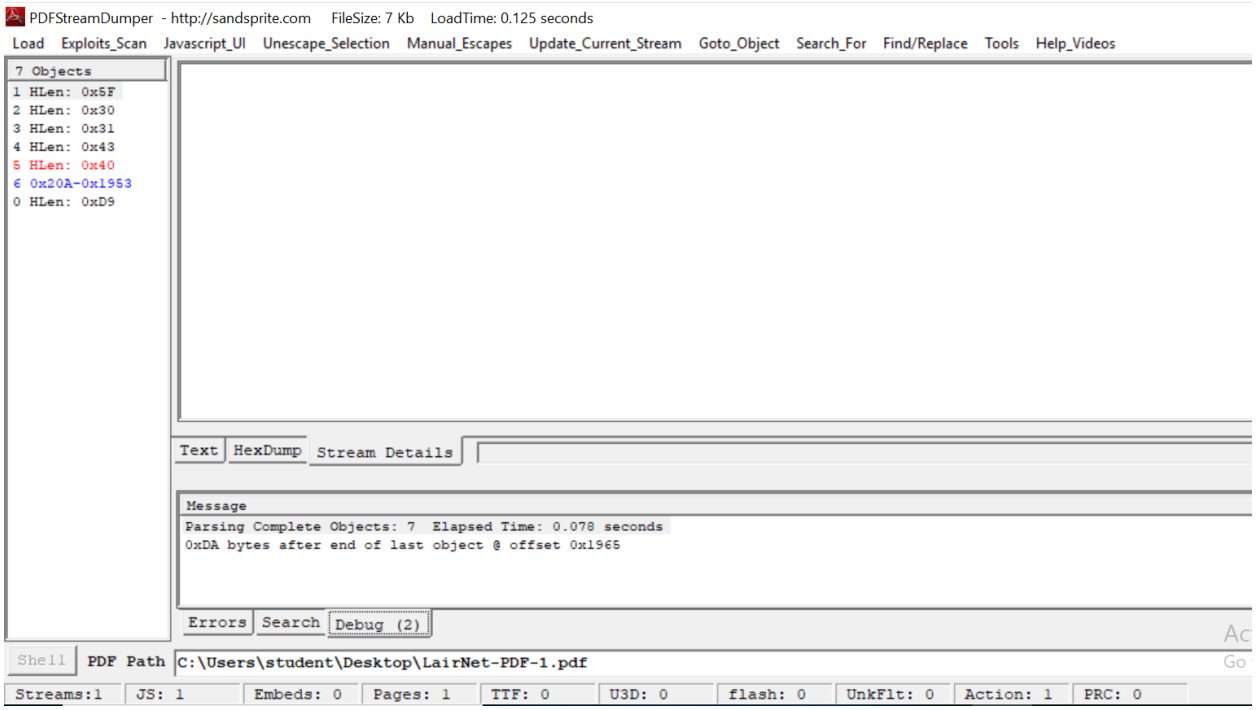
  LairNet-PDF-2.pdf

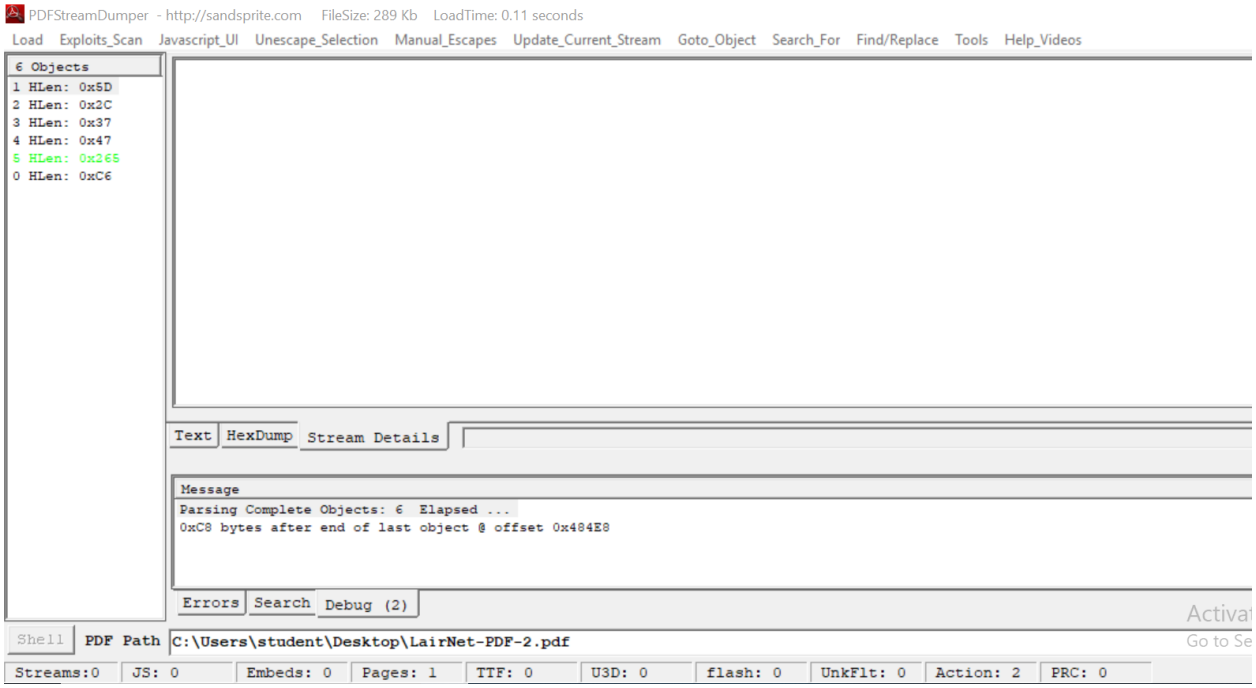
  LairNet-PDF-3.pdf

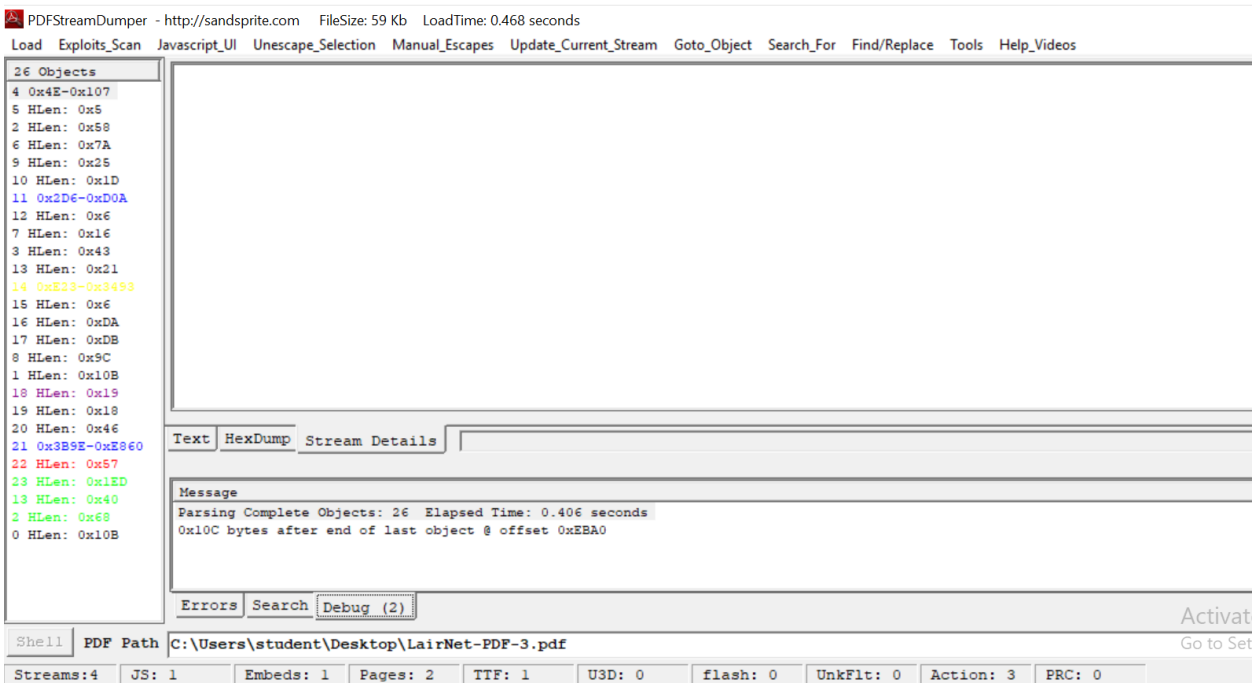
  LairNet-PDF-4.pdf

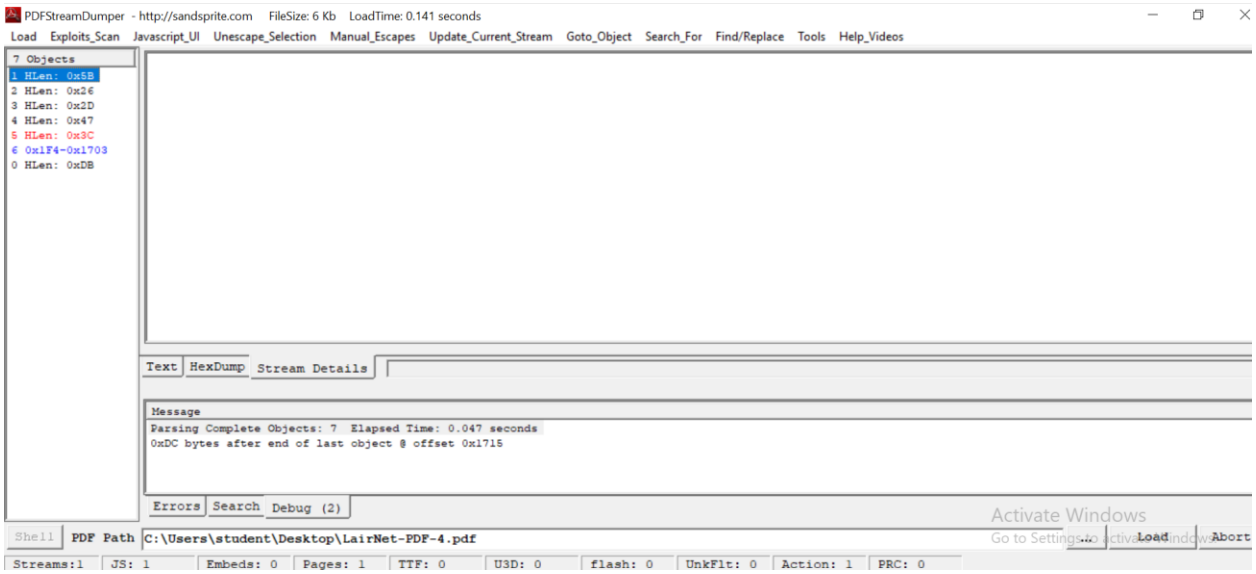
  SANS DFIR Hunt Evil.pdf

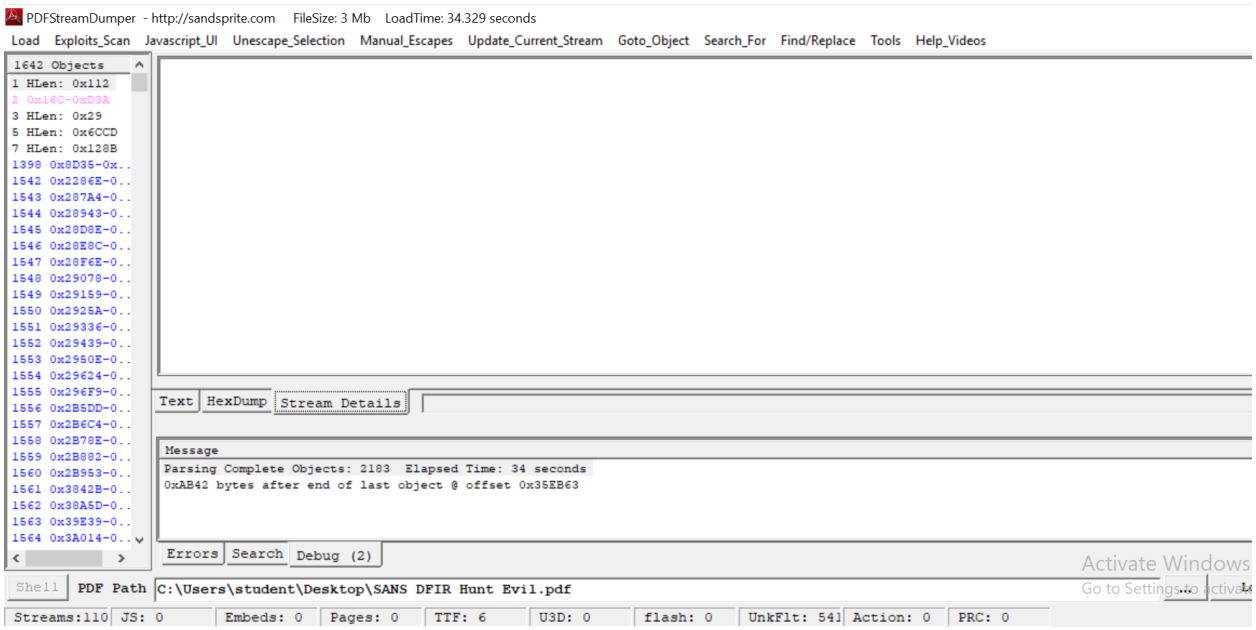
3. After Opening PDFStreamDumper, Click "Load - PDF File" and select one PDF document at a time.

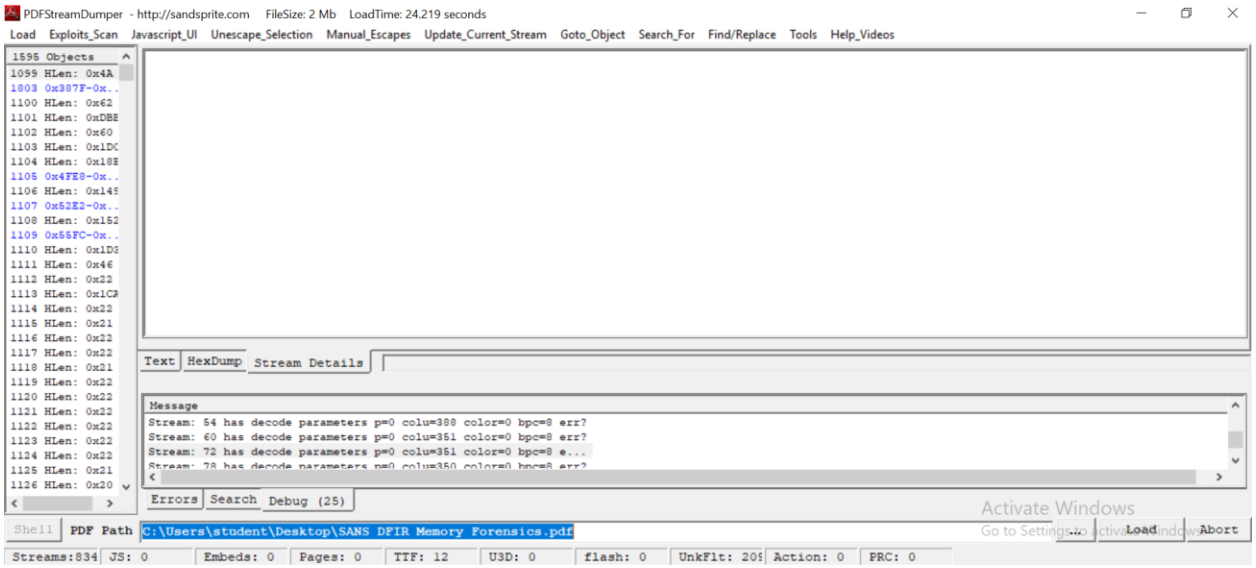

PDFStreamDumper - http://sandsprite.com FileSize: 7 Kb LoadTime: 0.125 seconds
Load Exploits_Scan Javascript_UI Unescape_Selection Manual_Escapes Update_Current_Stream Goto_Object Search_For Find/Replace Tools Help_Videos
7 Objects
1 HLen: 0x5F
2 HLen: 0x30
3 HLen: 0x31
4 HLen: 0x43
5 HLen: 0x40
6 0x20A-0x1953
0 HLen: 0xD9
Text HexDump Stream Details
Message
Parsing Complete Objects: 7 Elapsed Time: 0.078 seconds
0xDA bytes after end of last object @ offset 0x1965
Errors Search Debug (2)
Shell PDF Path C:\Users\student\Desktop\LairNet-PDF-1.pdf
Streams:1 JS: 1 Embeds: 0 Pages: 1 TTF: 0 U3D: 0 flash: 0 UnkFlt: 0 Action: 1 PRC: 0

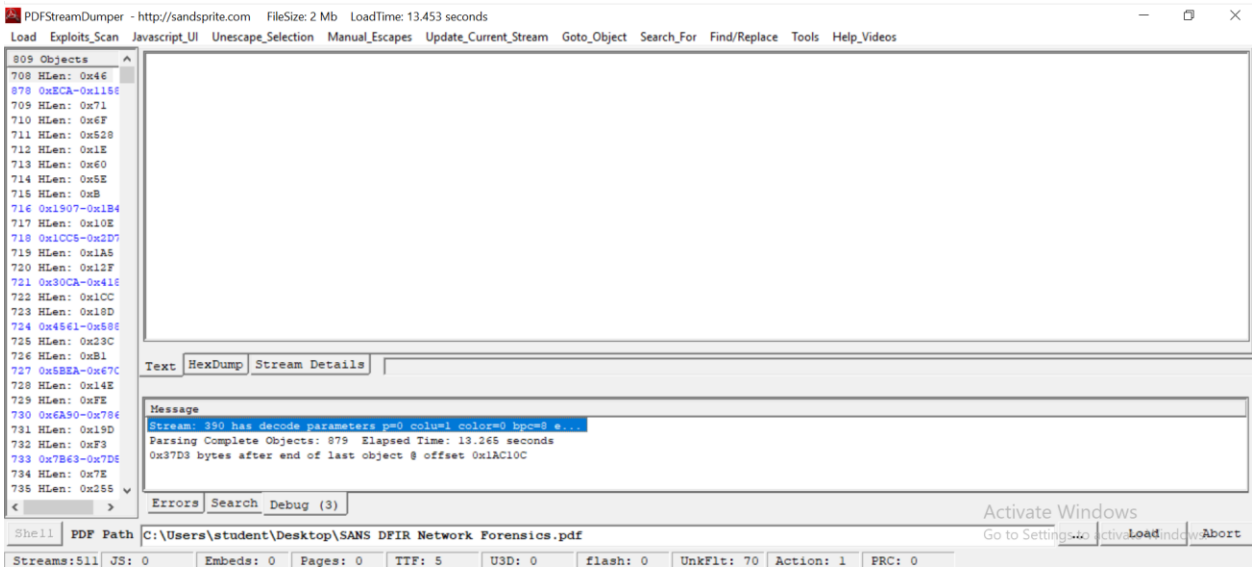

PDFStreamDumper - http://sandsprite.com FileSize: 289 Kb LoadTime: 0.11 seconds
Load Exploits_Scan Javascript_UI Unescape_Selection Manual_Escapes Update_Current_Stream Goto_Object Search_For Find/Replace Tools Help_Videos
6 Objects
1 HLen: 0x5D
2 HLen: 0x2C
3 HLen: 0x37
4 HLen: 0x47
5 HLen: 0x265
0 HLen: 0xC6
Text HexDump Stream Details
Message
Parsing Complete Objects: 6 Elapsed ...
0xC8 bytes after end of last object @ offset 0x484E8
Errors Search Debug (2)
Shell PDF Path C:\Users\student\Desktop\LairNet-PDF-2.pdf
Streams:0 JS: 0 Embeds: 0 Pages: 1 TTF: 0 U3D: 0 flash: 0 UnkFlt: 0 Action: 2 PRC: 0

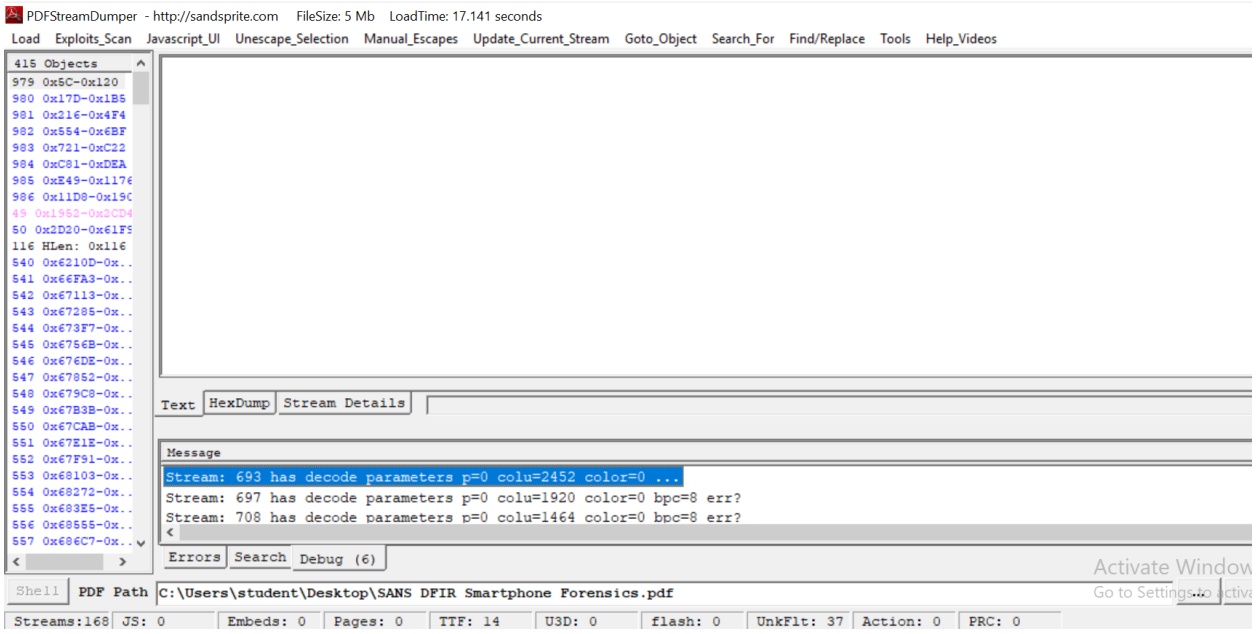

PDFStreamDumper - http://sandsprite.com FileSize: 59 Kb LoadTime: 0.468 seconds
Load Exploits_Scan Javascript_UI Unescape_Selection Manual_Escapes Update_Current_Stream Goto_Object Search_For Find/Replace Tools Help_Videos
26 Objects
4 0x4E-0x107
5 HLen: 0x5
2 HLen: 0x58
6 HLen: 0x7A
9 HLen: 0x25
10 HLen: 0x1D
11 0x2D6-0xD0A
12 HLen: 0x6
7 HLen: 0x16
3 HLen: 0x43
13 HLen: 0x21
14 0xE23-0x3493
15 HLen: 0x6
16 HLen: 0xDA
17 HLen: 0xDB
8 HLen: 0x9C
1 HLen: 0x10B
18 HLen: 0x19
19 HLen: 0x18
20 HLen: 0x46
21 0x3B9E-0xE860
22 HLen: 0x57
23 HLen: 0x1ED
13 HLen: 0x40
2 HLen: 0x68
0 HLen: 0x10B
Text HexDump Stream Details
Message
Parsing Complete Objects: 26 Elapsed Time: 0.406 seconds
0x10C bytes after end of last object @ offset 0xEBA0
Errors Search Debug (2)
Shell PDF Path C:\Users\student\Desktop\LairNet-PDF-3.pdf
Streams:4 JS: 1 Embeds: 1 Pages: 2 TTF: 1 U3D: 0 flash: 0 UnkFlt: 0 Action: 3 PRC: 0

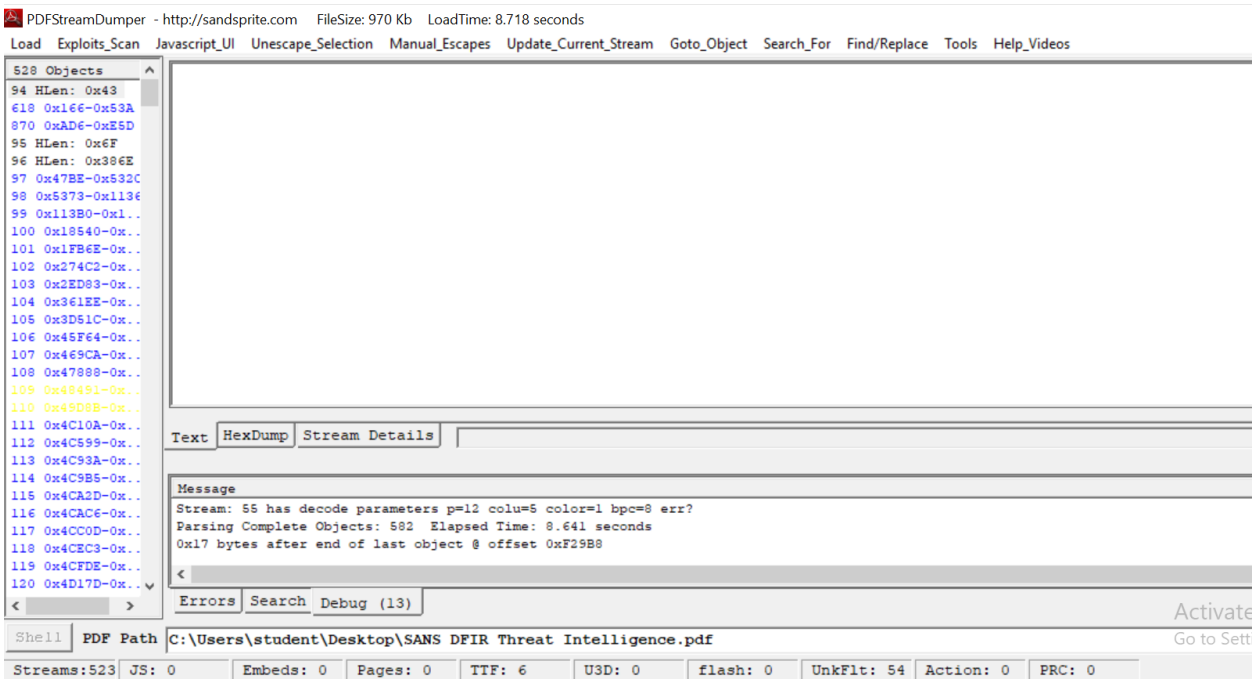

PDFStreamDumper - http://sandsprite.com FileSize: 6 Kb LoadTime: 0.141 seconds
Load Exploits_Scan Javascript_UI Unescape_Selection Manual_Escapes Update_Current_Stream Goto_Object Search_For Find/Replace Tools Help_Videos
7 Objects
1 HLen: 0x5B
2 HLen: 0x26
3 HLen: 0x2D
4 HLen: 0x47
5 HLen: 0x3C
6 0x1F4-0x1703
0 HLen: 0xDB
Text HexDump Stream Details
Message
Parsing Complete Objects: 7 Elapsed Time: 0.047 seconds
0xDC bytes after end of last object @ offset 0x1715
Errors Search Debug (2)
Shell PDF Path C:\Users\student\Desktop\LairNet-PDF-4.pdf
Load Abort
Streams:1 JS: 1 Embeds: 0 Pages: 1 TTF: 0 U3D: 0 flash: 0 UnkFlt: 0 Action: 1 PRC: 0

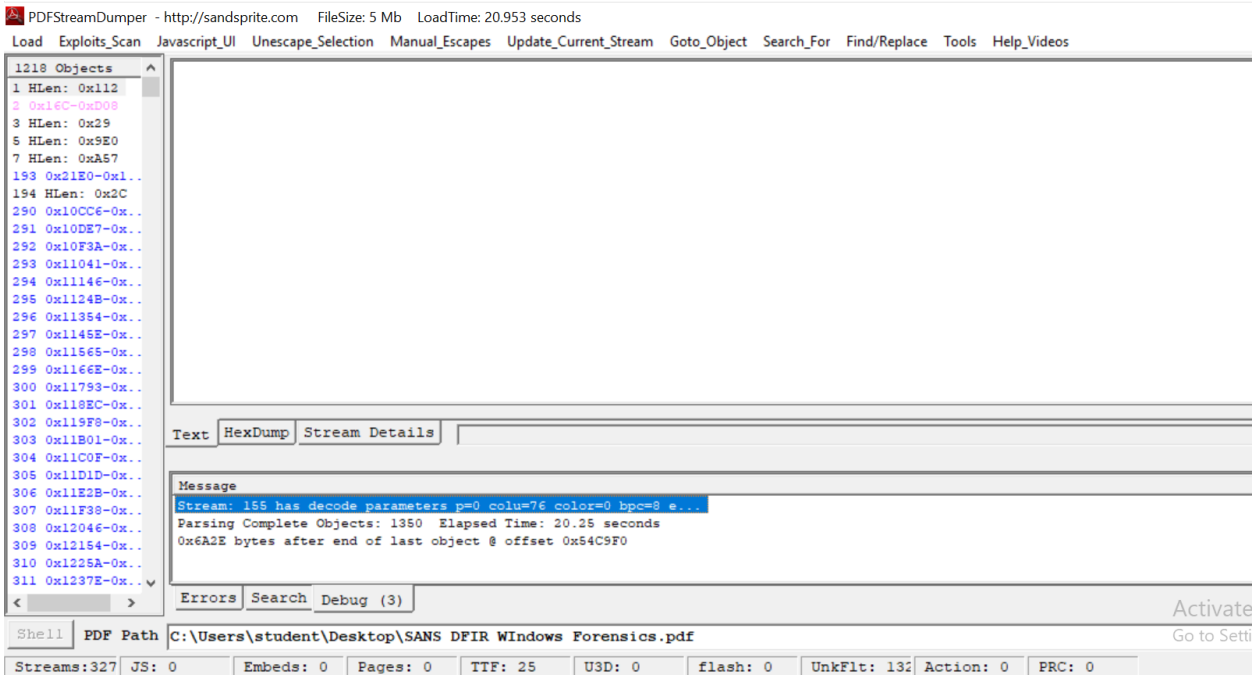

PDFStreamDumper - http://sandsprite.com FileSize: 3 Mb LoadTime: 34.329 seconds
Load Exploits_Scan Javascript_UI Unescape_Selection Manual_Escapes Update_Current_Stream Goto_Object Search_For Find/Replace Tools Help_Videos
1642 Objects
1 HLen: 0x112
2 0x16C-0xD3A
3 HLen: 0x29
5 HLen: 0x6CCD
7 HLen: 0x128B
1398 0x8D35-0x..
1542 0x2286E-0..
1543 0x287A4-0..
1544 0x28943-0..
1545 0x28D8E-0..
1546 0x28E8C-0..
1547 0x28F6E-0..
1548 0x29078-0..
1549 0x29159-0..
1550 0x2925A-0..
1551 0x29336-0..
1552 0x29439-0..
1553 0x2950E-0..
1554 0x29624-0..
1555 0x296F9-0..
1556 0x2B5DD-0..
1557 0x2B6C4-0..
1558 0x2B78E-0..
1559 0x2B882-0..
1560 0x2B953-0..
1561 0x3842B-0..
1562 0x38A5D-0..
1563 0x39E39-0..
1564 0x3A014-0..
Text HexDump Stream Details
Message
Parsing Complete Objects: 2183 Elapsed Time: 34 seconds
0xAB42 bytes after end of last object @ offset 0x35EB63
Errors Search Debug (2)
Activate Windows
Go to Settings...
Shell PDF Path C:\Users\student\Desktop\SANS DFIR Hunt Evil.pdf
Streams:110 JS: 0 Embeds: 0 Pages: 0 TTF: 6 U3D: 0 flash: 0 UnkFlt: 54 Action: 0 PRC: 0


PDFStreamDumper - http://sandsprite.com FileSize: 2 Mb LoadTime: 24.219 seconds
Load Exploits_Scan Javascript_UI Unescape_Selection Manual_Escapes Update_Current_Stream Goto_Object Search_For Find/Replace Tools Help_Videos
1595 Objects
1099 HLen: 0x4A
1803 0x387F-0x..
1100 HLen: 0x62
1101 HLen: 0xDBE
1102 HLen: 0x60
1103 HLen: 0x1DC
1104 HLen: 0x18E
1105 0x4FE8-0x..
1106 HLen: 0x145
1107 0x52E2-0x..
1108 HLen: 0x152
1109 0x55FC-0x..
1110 HLen: 0x1D8
1111 HLen: 0x46
1112 HLen: 0x22
1113 HLen: 0x1CA
1114 HLen: 0x22
1115 HLen: 0x21
1116 HLen: 0x22
1117 HLen: 0x22
1118 HLen: 0x21
1119 HLen: 0x22
1120 HLen: 0x22
1121 HLen: 0x22
1122 HLen: 0x22
1123 HLen: 0x22
1124 HLen: 0x22
1125 HLen: 0x21
1126 HLen: 0x20
Text HexDump Stream Details
Message
Stream: 54 has decode parameters p=0 colu=388 color=0 bpc=8 err?
Stream: 60 has decode parameters p=0 colu=351 color=0 bpc=8 err?
Stream: 72 has decode parameters p=0 colu=351 color=0 bpc=8 e...
Errors Search Debug (25)
Activate Windows
Go to Settings...
Shell PDF Path C:\Users\student\Desktop\SANS DFIR Memory Forensics.pdf
Load Abort
Streams:834 JS: 0 Embeds: 0 Pages: 0 TTF: 12 U3D: 0 flash: 0 UnkFlt: 205 Action: 0 PRC: 0


PDFStreamDumper - http://sandsprite.com FileSize: 2 Mb LoadTime: 13.453 seconds
Load Exploits_Scan Javascript_UI Unescape_Selection Manual_Escapes Update_Current_Stream Goto_Object Search_For Find/Replace Tools Help_Videos
809 Objects
Text HexDump Stream Details
Message
Stream: 390 has decode parameters p=0 colu=1 color=0 bpc=8 e...
Parsing Complete Objects: 879 Elapsed Time: 13.265 seconds
0x37D3 bytes after end of last object @ offset 0x1AC10C
Errors Search Debug (3)
Shell PDF Path C:\Users\student\Desktop\SANS DFIR Network Forensics.pdf
Load Abort
Streams:511 JS: 0 Embeds: 0 Pages: 0 TTF: 5 U3D: 0 flash: 0 UnkFlt: 70 Action: 1 PRC: 0


PDFStreamDumper - http://sandsprite.com FileSize: 5 Mb LoadTime: 17.141 seconds
Load Exploits_Scan Javascript_UI Unescape_Selection Manual_Escapes Update_Current_Stream Goto_Object Search_For Find/Replace Tools Help_Videos
415 Objects
Text HexDump Stream Details
Message
Stream: 693 has decode parameters p=0 colu=2452 color=0 ...
Stream: 697 has decode parameters p=0 colu=1920 color=0 bpc=8 err?
Stream: 708 has decode parameters p=0 colu=1464 color=0 bpc=8 err?
Errors Search Debug (6)
Shell PDF Path C:\Users\student\Desktop\SANS DFIR Smartphone Forensics.pdf
Streams:168 JS: 0 Embeds: 0 Pages: 0 TTF: 14 U3D: 0 flash: 0 UnkFlt: 37 Action: 0 PRC: 0



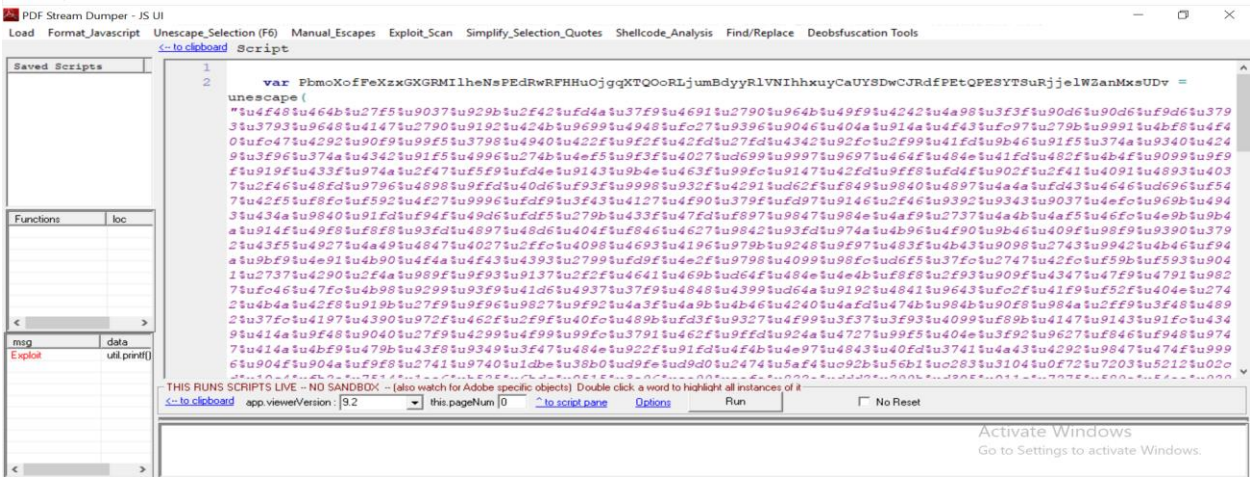


4. Manually review the contents of each "Object" to see if you can identify any questionable content, or strings of interest. [4]

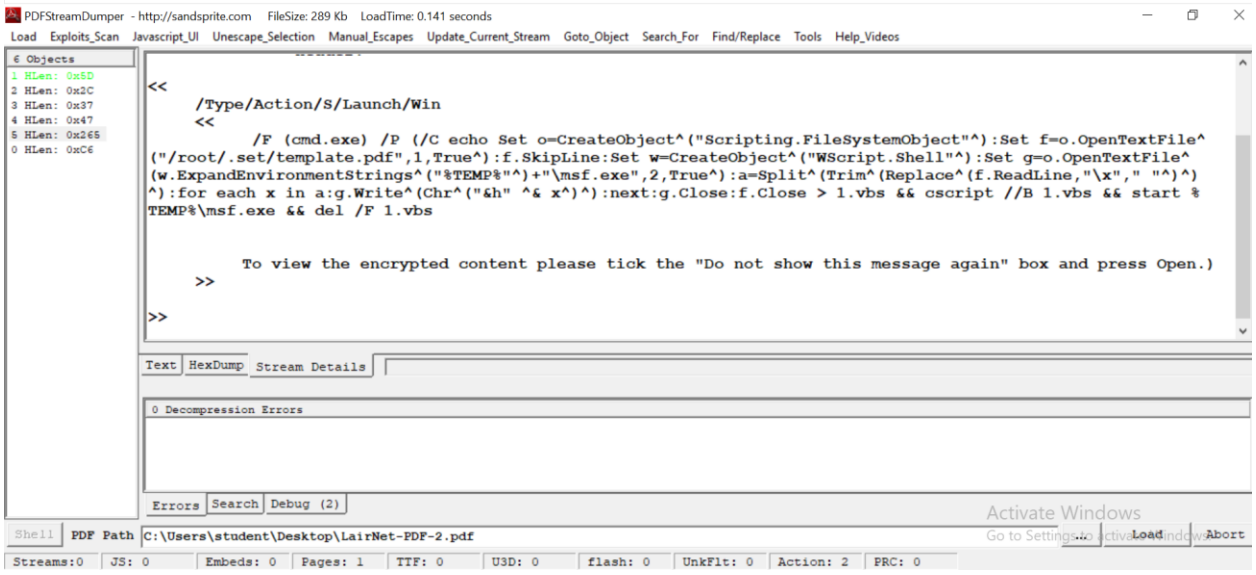
- Manually review each object and if Javascript is detected in the object then click on the Javascript_UI option to clearly view the detected javascript.

- After reviewing all the below files some files contain some embedded XML image content and javascript code executing commands on the user's computer once the PDF files are open.
- What is PDF Injection – In PDF injection attackers inject javascript code or any XML code into a PDF file. Once the user opens the PDF file this added code gets executed and performs actions as stated in the added code like executing commands on the user's computer. [3]

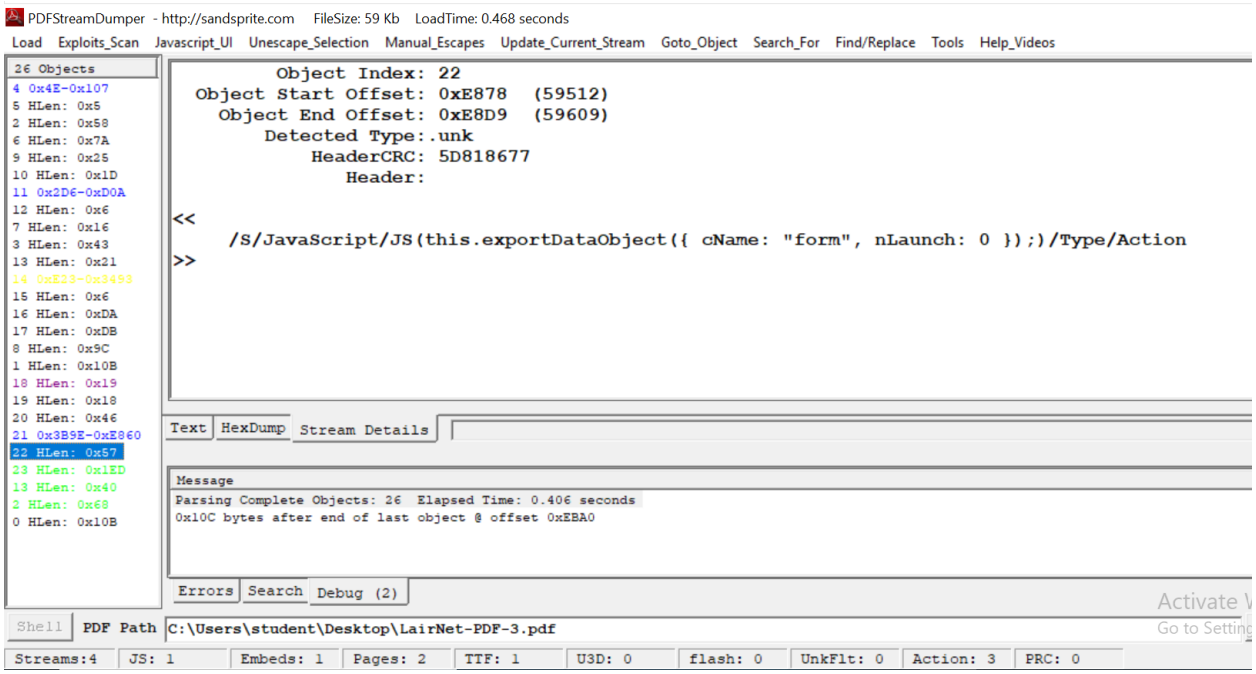
LairNet-PDF-1.pdf – The JavaScript code detected below is performing some operations and it is encrypted content, hence cannot check the further functionality of the code. But it seems to be malicious javascript.

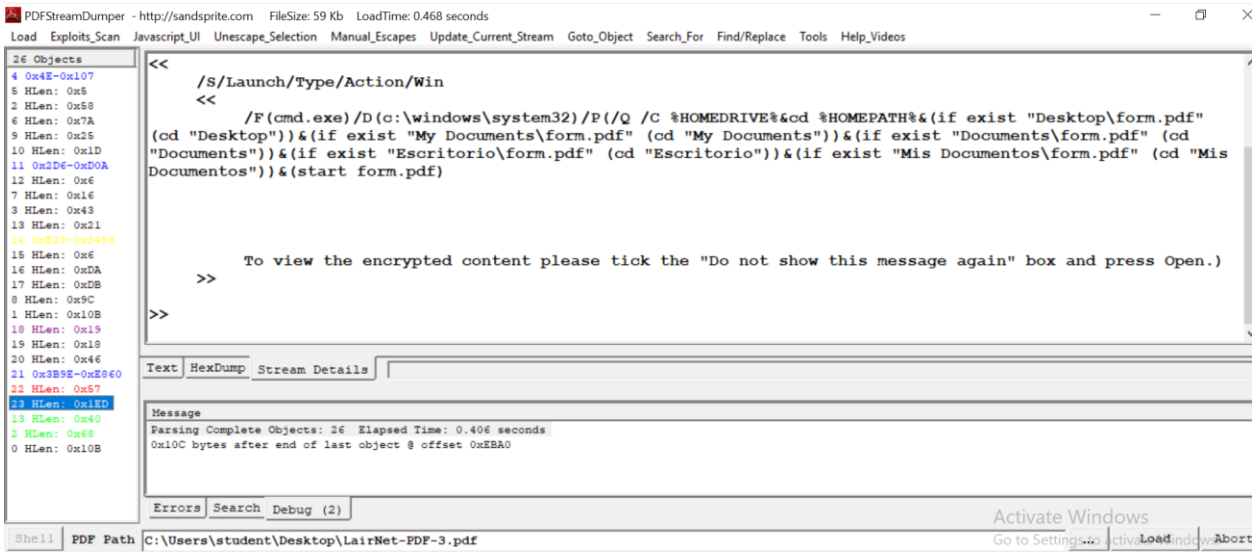


LairNet-PDF-2.pdf - By looking at the below javascript code, it is obfuscated javascript code creating a malicious executable containing some commands which will run on the user's computer once the PDF file is open.

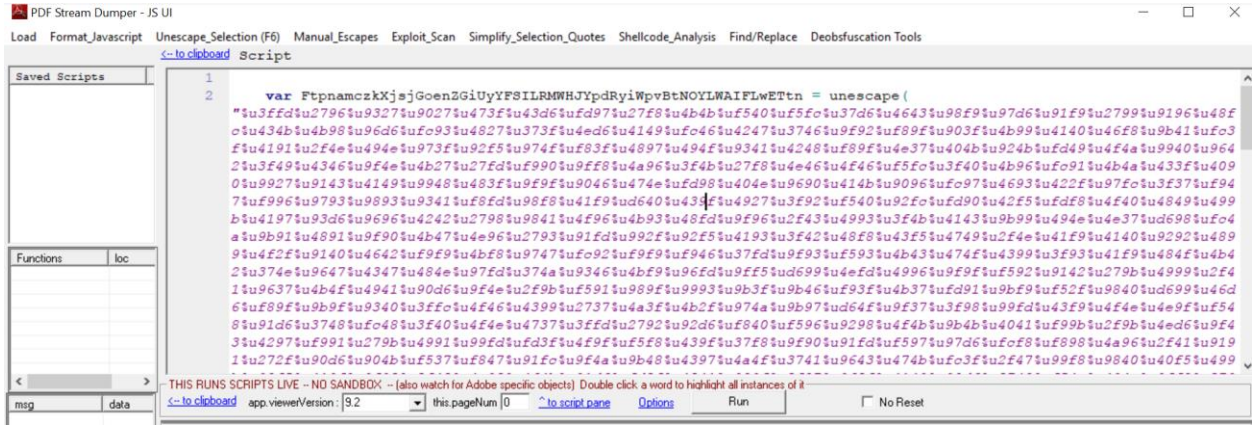


LairNet-PDF-3.pdf – The below code will run some suspicious commands on user's machine once the PDF file is open.

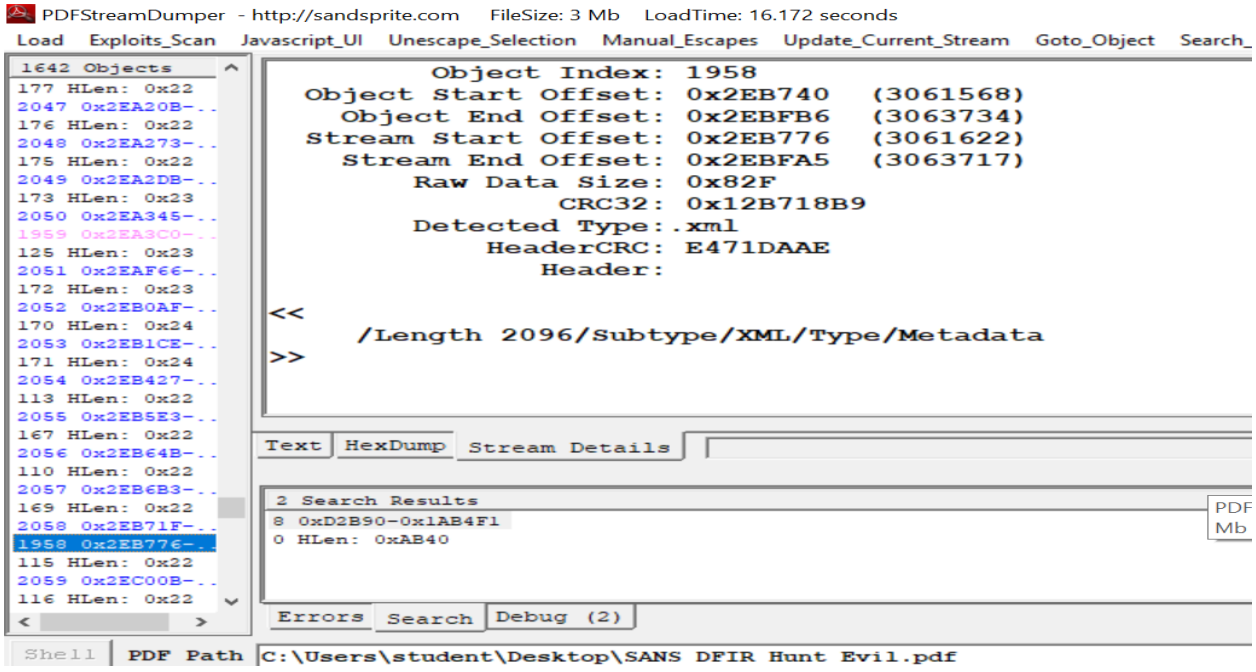


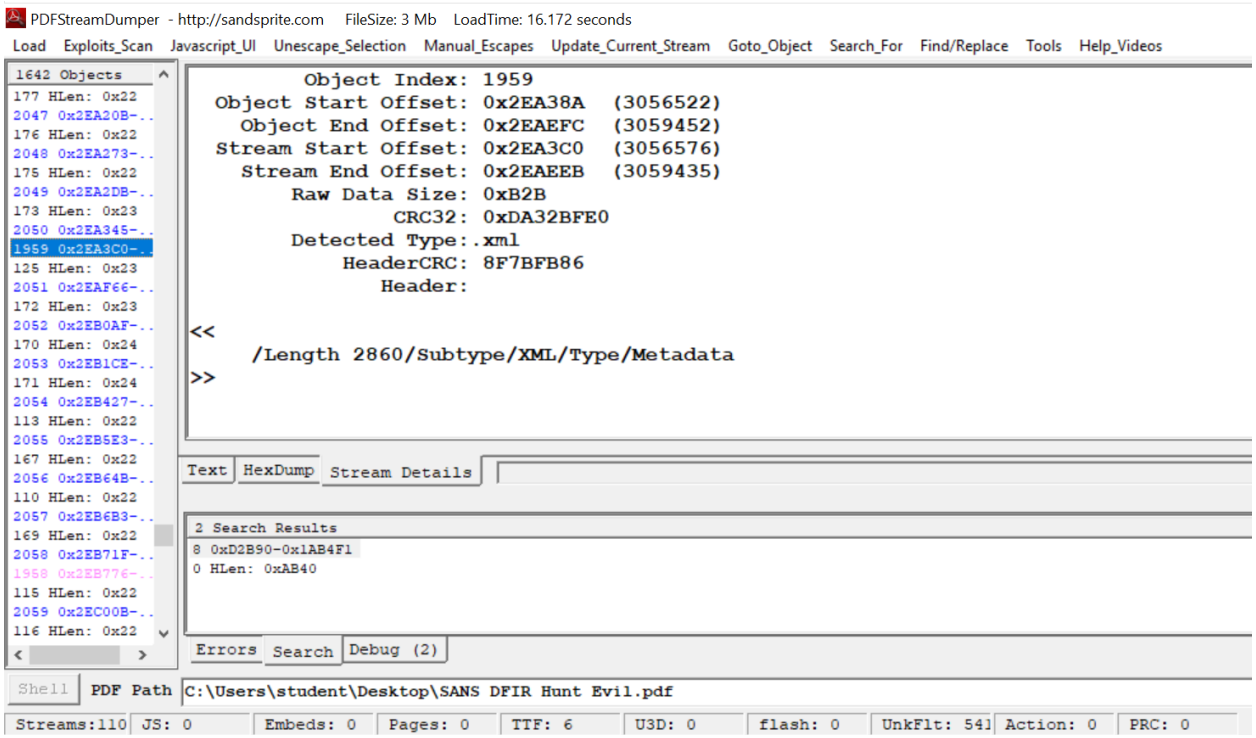


LairNet-PDF-4.pdf – Below is a encrypted javascript code

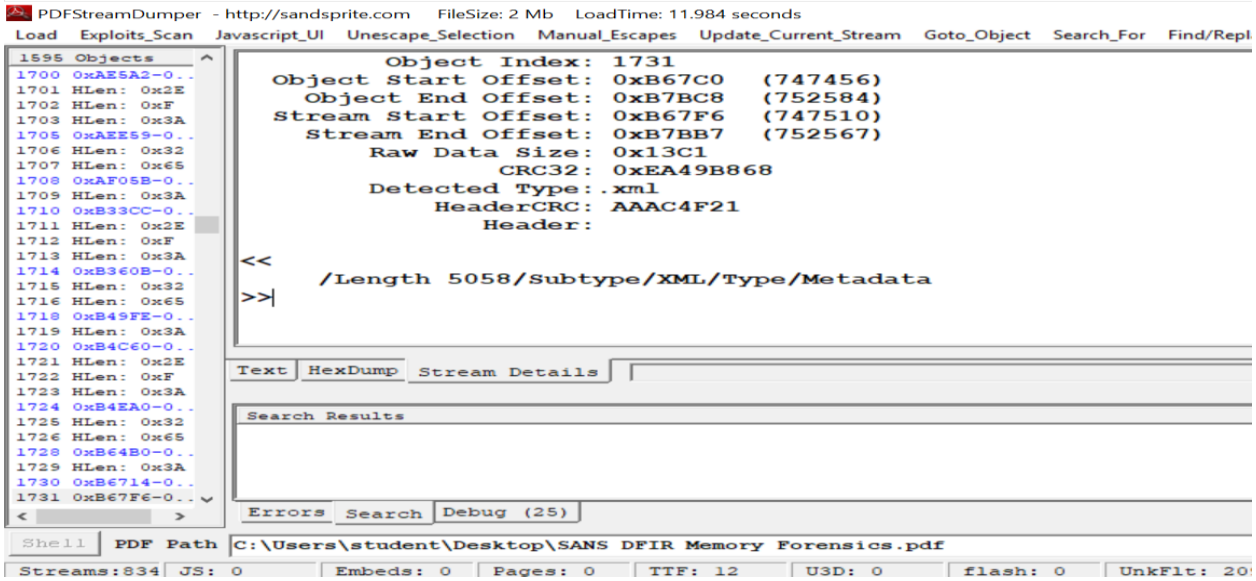


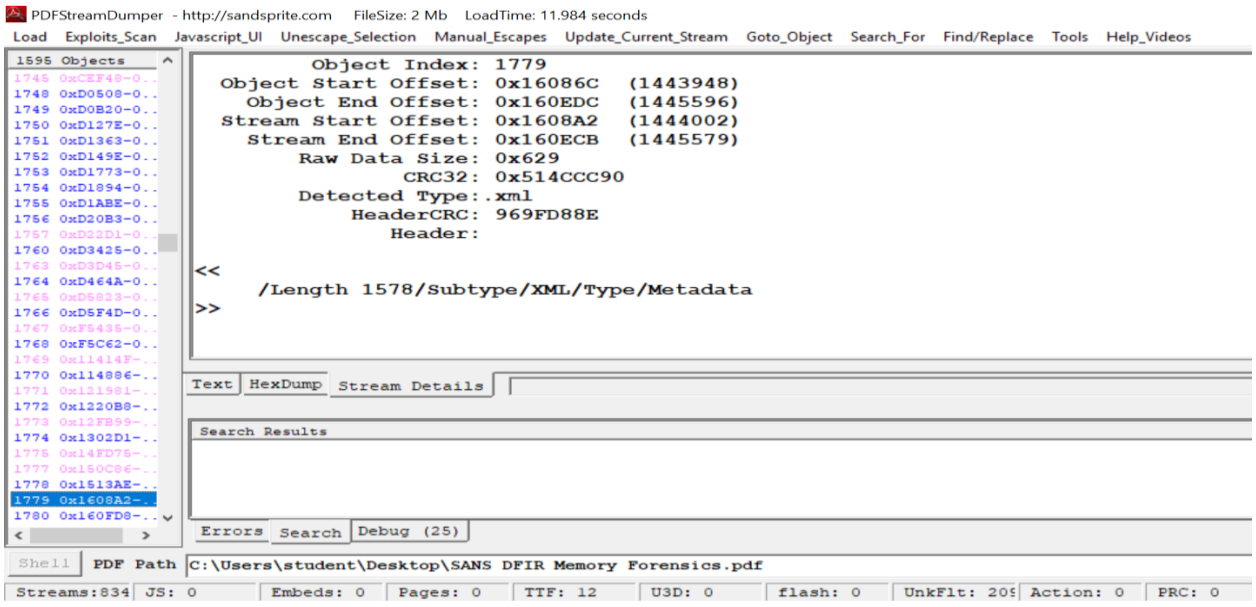
SANS DFIR Hunt Evil.pdf – The below file contain some embedded image which looks suspicious.


PDFStreamDumper - http://sandsprite.com FileSize: 3 Mb LoadTime: 16.172 seconds
Load Exploits_Scan Javascript_UI Unescape_Selection Manual_Escapes Update_Current_Stream Goto_Object Search_
1642 Objects
177 HLen: 0x22
2047 0x2EA20B-..
176 HLen: 0x22
2048 0x2EA273-..
175 HLen: 0x22
2049 0x2EA2DB-..
173 HLen: 0x23
2050 0x2EA345-..
1959 0x2EA3C0-..
125 HLen: 0x23
2051 0x2EAF66-..
172 HLen: 0x23
2052 0x2EB0AF-..
170 HLen: 0x24
2053 0x2EB1CE-..
171 HLen: 0x24
2054 0x2EB427-..
113 HLen: 0x22
2055 0x2EB5E3-..
167 HLen: 0x22
2056 0x2EB64B-..
110 HLen: 0x22
2057 0x2EB6B3-..
169 HLen: 0x22
2058 0x2EB71F-..
1958 0x2EB776-..
115 HLen: 0x22
2059 0x2EC00B-..
116 HLen: 0x22
Object Index: 1958
Object Start Offset: 0x2EB740 (3061568)
Object End Offset: 0x2EBFB6 (3063734)
Stream Start Offset: 0x2EB776 (3061622)
Stream End Offset: 0x2EBFA5 (3063717)
Raw Data Size: 0x82F
CRC32: 0x12B718B9
Detected Type:.xml
HeaderCRC: E471DAAE
Header:
<<
/Length 2096/Subtype/XML/Type/Metadata
>>
Text HexDump Stream Details
2 Search Results
8 0xD2B90-0x1AB4F1
0 HLen: 0xAB40
PDF
Mb
Errors Search Debug (2)
Shell PDF Path C:\Users\student\Desktop\SANS DFIR Hunt Evil.pdf

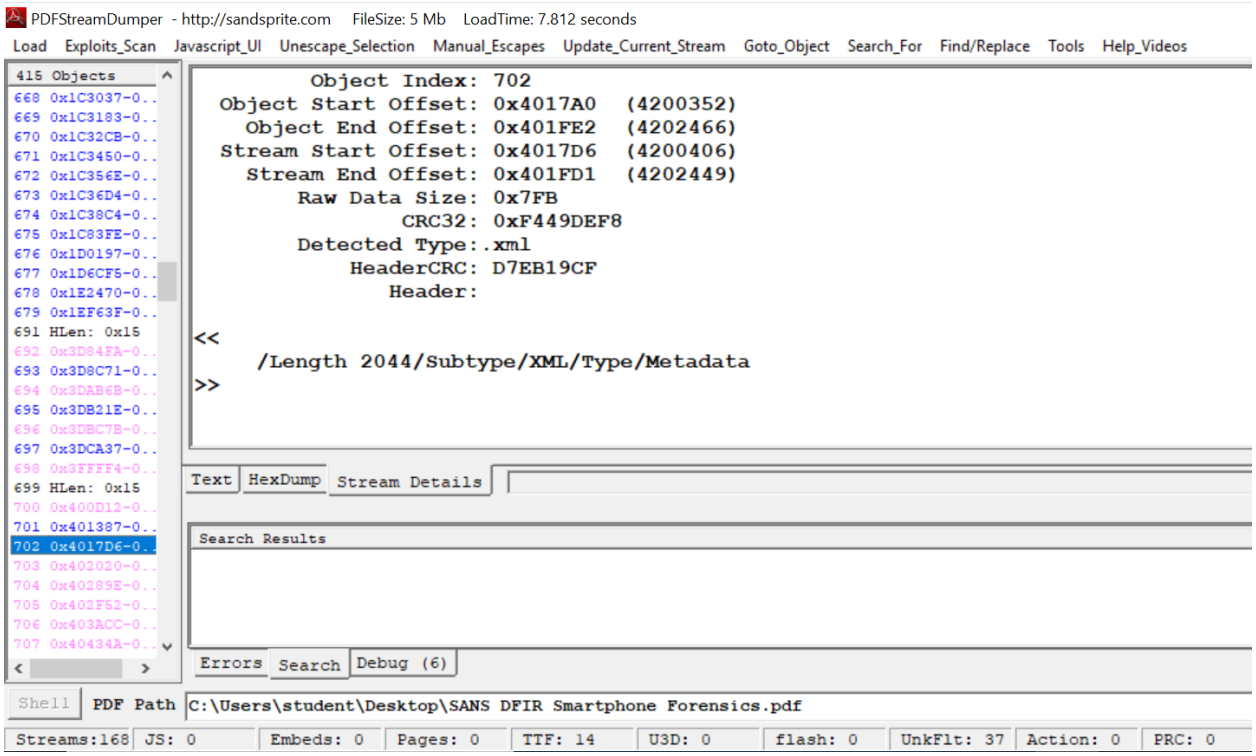

PDFStreamDumper - http://sandsprite.com FileSize: 3 Mb LoadTime: 16.172 seconds
Load Exploits_Scan Javascript_UI Unescape_Selection Manual_Escapes Update_Current_Stream Goto_Object Search_For Find/Replace Tools Help_Videos
1642 Objects
177 HLen: 0x22
2047 0x2EA20B-..
176 HLen: 0x22
2048 0x2EA273-..
175 HLen: 0x22
2049 0x2EA2DB-..
173 HLen: 0x23
2050 0x2EA345-..
1959 0x2EA3C0-..
125 HLen: 0x23
2051 0x2EAF66-..
172 HLen: 0x23
2052 0x2EB0AF-..
170 HLen: 0x24
2053 0x2EB1CE-..
171 HLen: 0x24
2054 0x2EB427-..
113 HLen: 0x22
2055 0x2EB5E3-..
167 HLen: 0x22
2056 0x2EB64B-..
110 HLen: 0x22
2057 0x2EB6B3-..
169 HLen: 0x22
2058 0x2EB71F-..
1958 0x2EB776-..
115 HLen: 0x22
2059 0x2EC00B-..
116 HLen: 0x22
Object Index: 1959
Object Start Offset: 0x2EA38A (3056522)
Object End Offset: 0x2EAEFC (3059452)
Stream Start Offset: 0x2EA3C0 (3056576)
Stream End Offset: 0x2EAEEB (3059435)
Raw Data Size: 0xB2B
CRC32: 0xDA32BFE0
Detected Type:.xml
HeaderCRC: 8F7BFB86
Header:
<<
/Length 2860/Subtype/XML/Type/Metadata
>>
Text HexDump Stream Details
2 Search Results
8 0xD2B90-0x1AB4F1
0 HLen: 0xAB40
Errors Search Debug (2)
Shell PDF Path C:\Users\student\Desktop\SANS DFIR Hunt Evil.pdf
Streams:110 JS: 0 Embeds: 0 Pages: 0 TTF: 6 U3D: 0 flash: 0 UnkFlt: 541 Action: 0 PRC: 0

SANS DFIR Memory Forensics.pdf – The below file contain some embedded XML images on some objects as shown below.

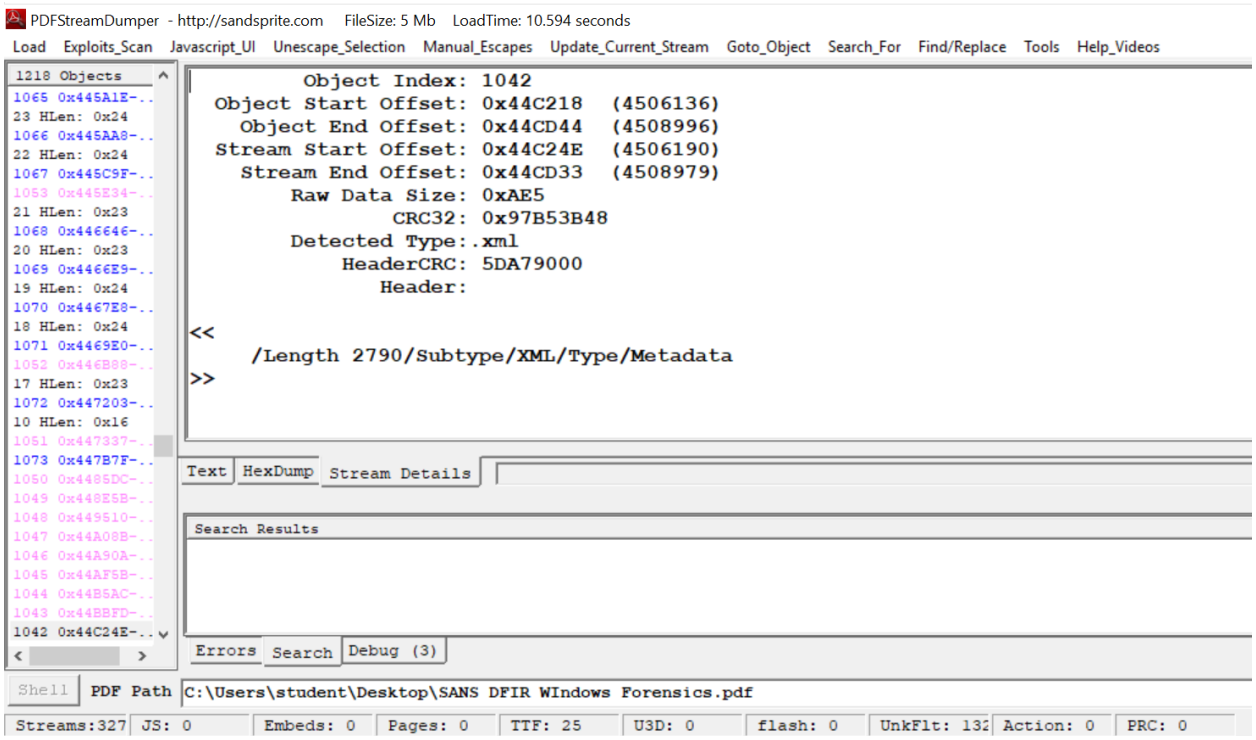




SANS DFIR Smartphone Forensics.pdf – The file contains some XML content of embedded image as shown below on various object

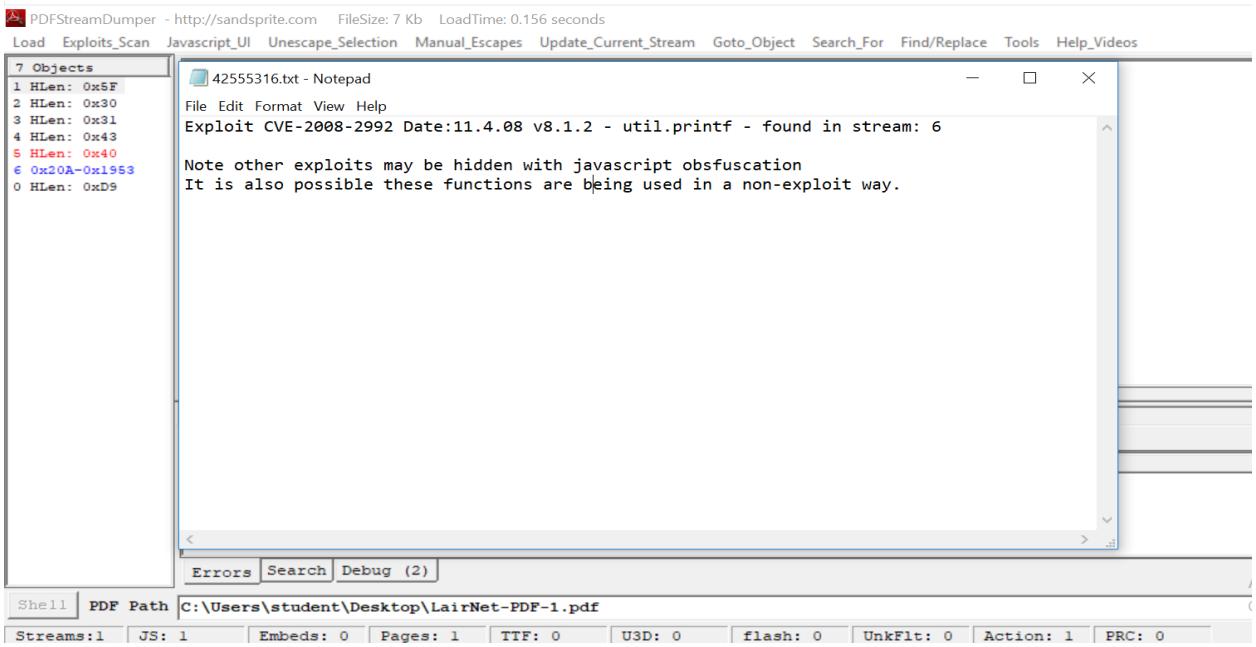


SANS DFIR Windows Forensics.pdf – The below file contains some suspicious XML content of the embedded file.

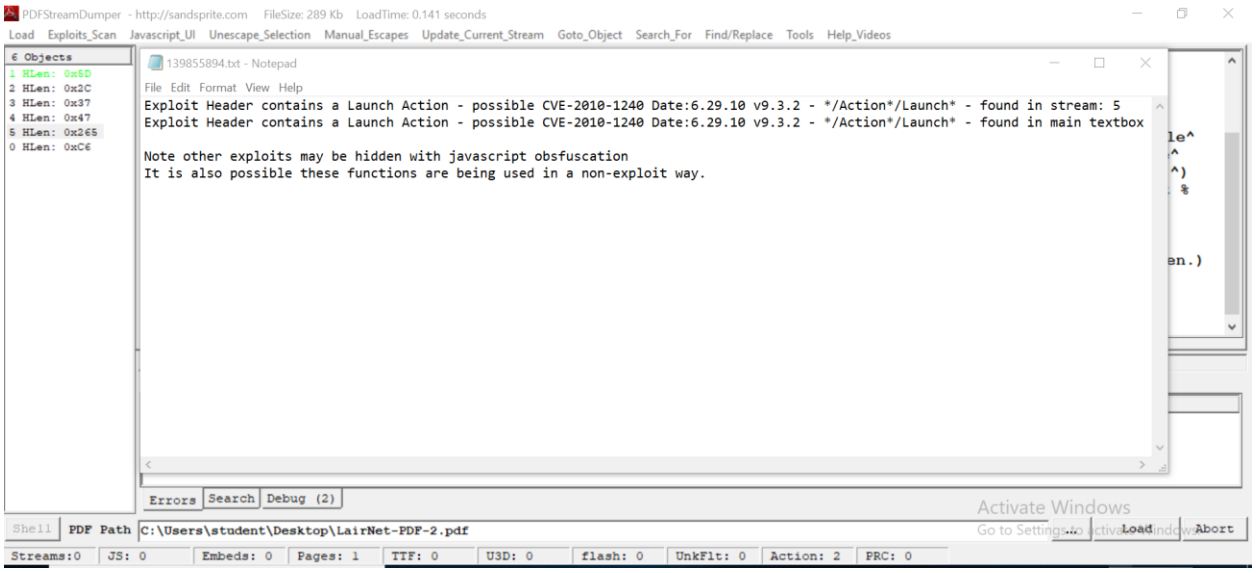


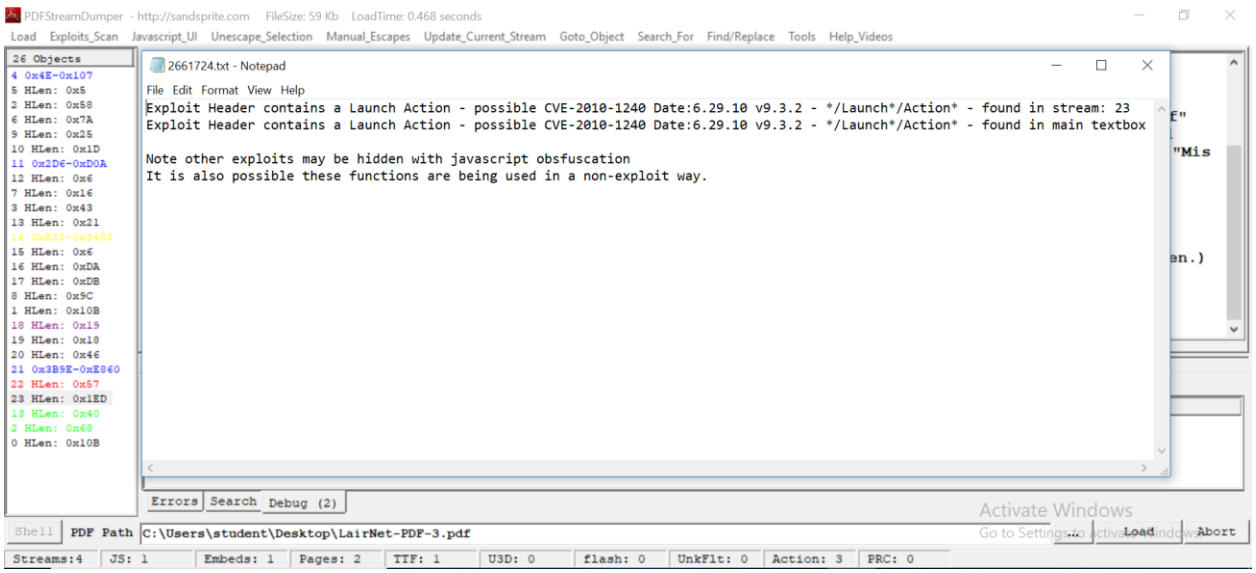
5. Use the "Exploits_Scan" option from the menu to scan the loaded PDF document to see if it contains any exploits.

    1. LairNet-PDF-1.pdf

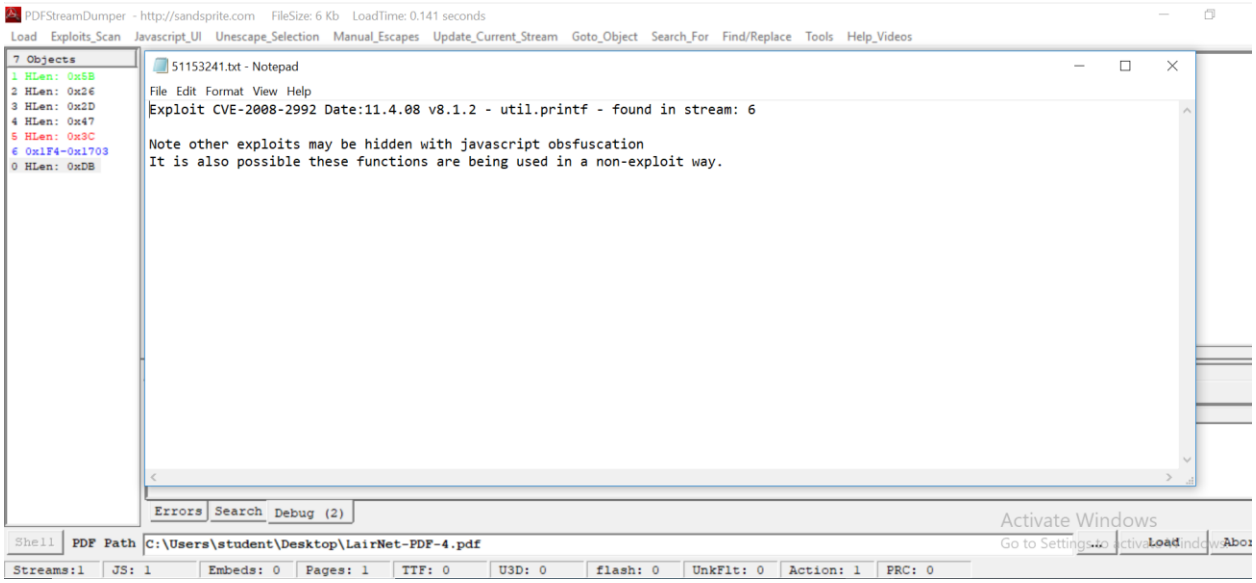


    2. LairNet-PDF-2.pdf –

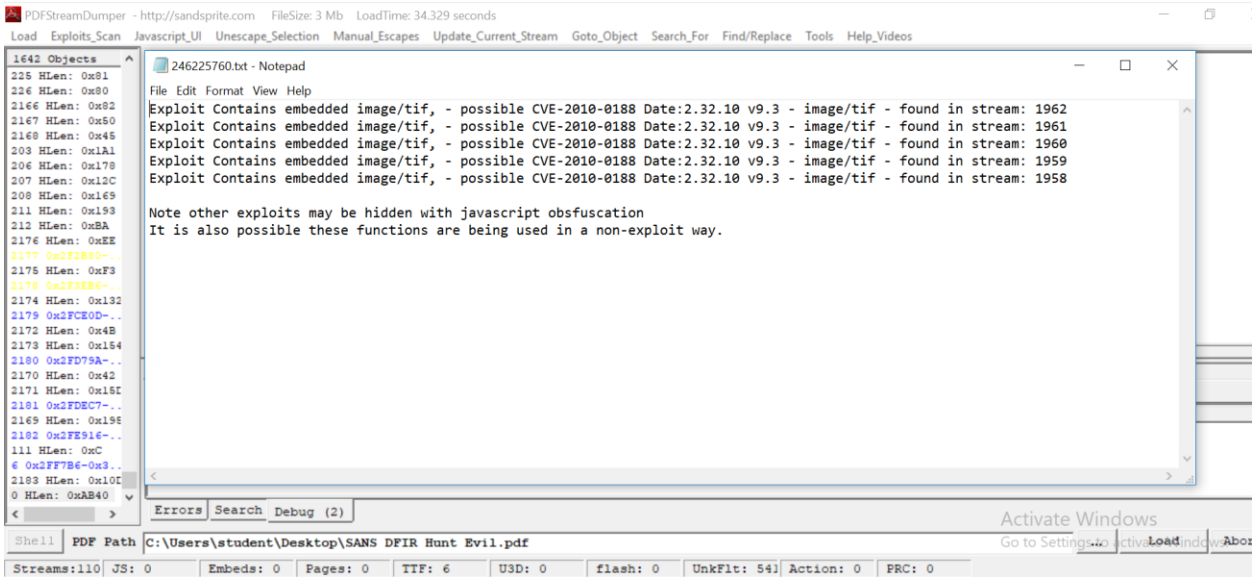




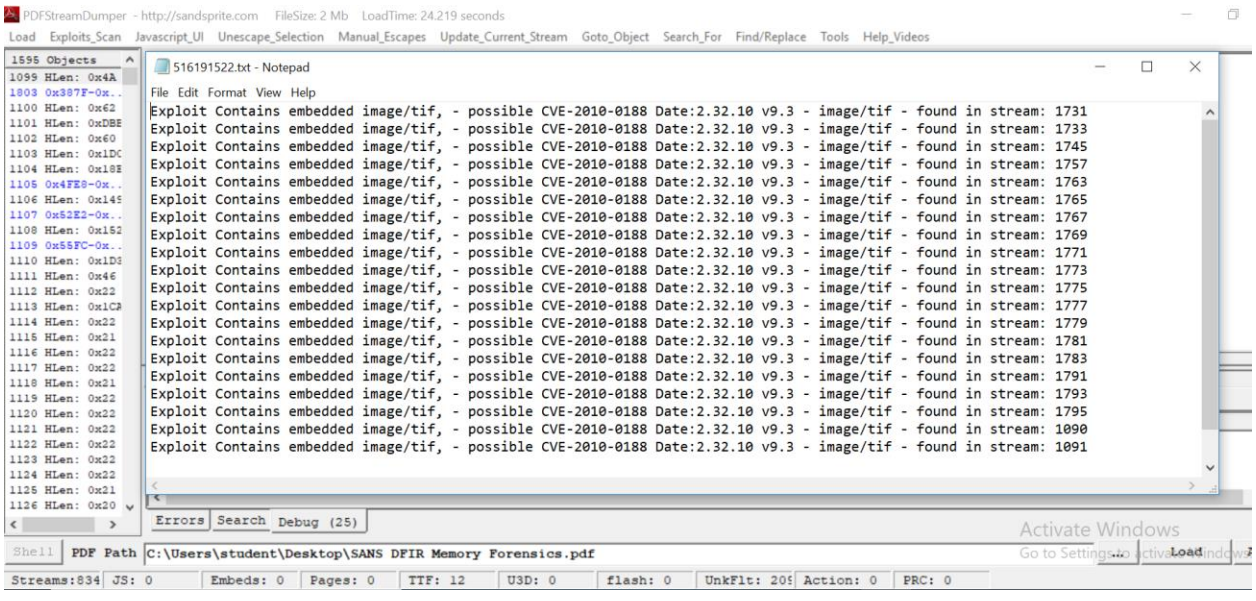
LairNet-PDF-4.pdf -

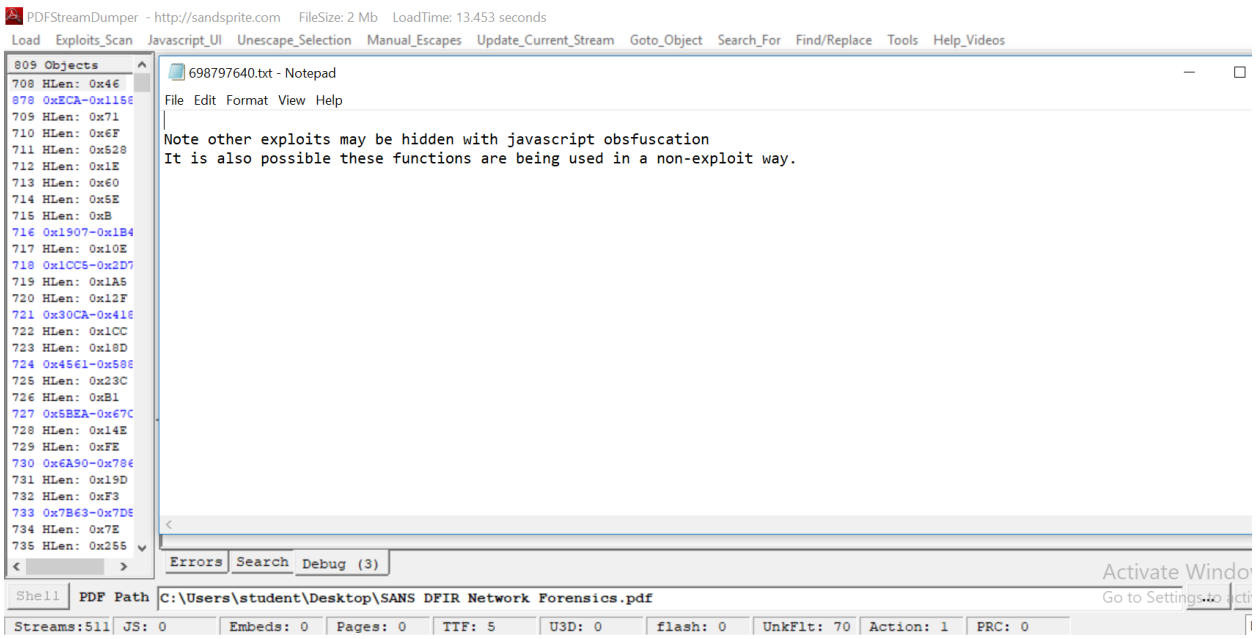


- SANS DFIR Hunt Evil.pdf

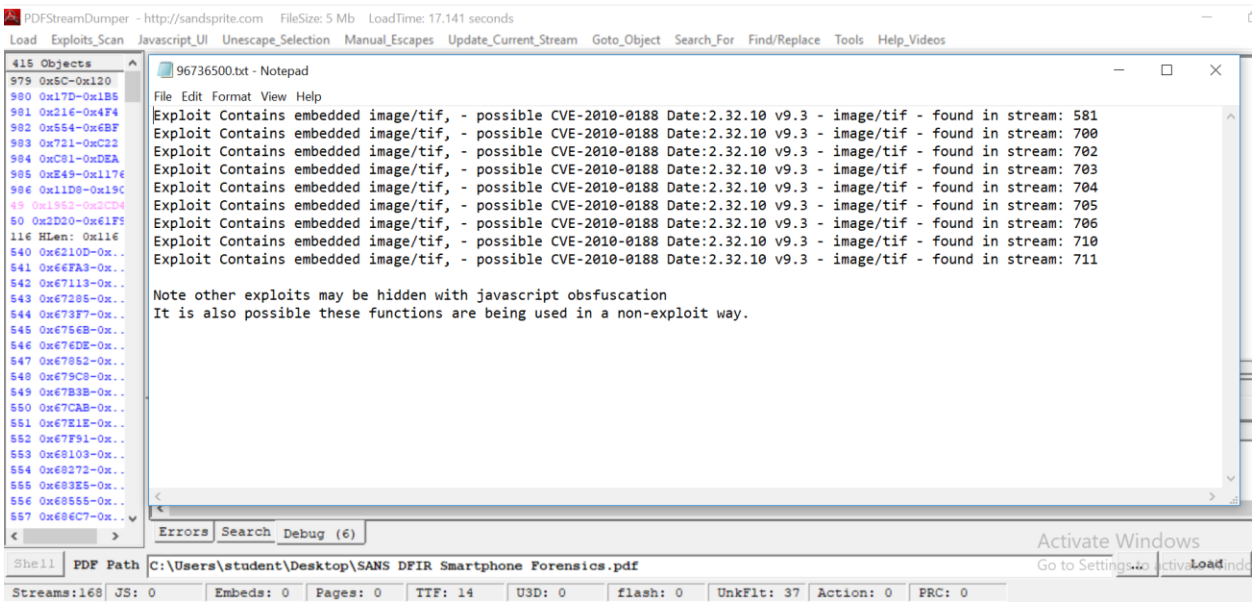


- SANS DFIR Memory Forensics.pdf

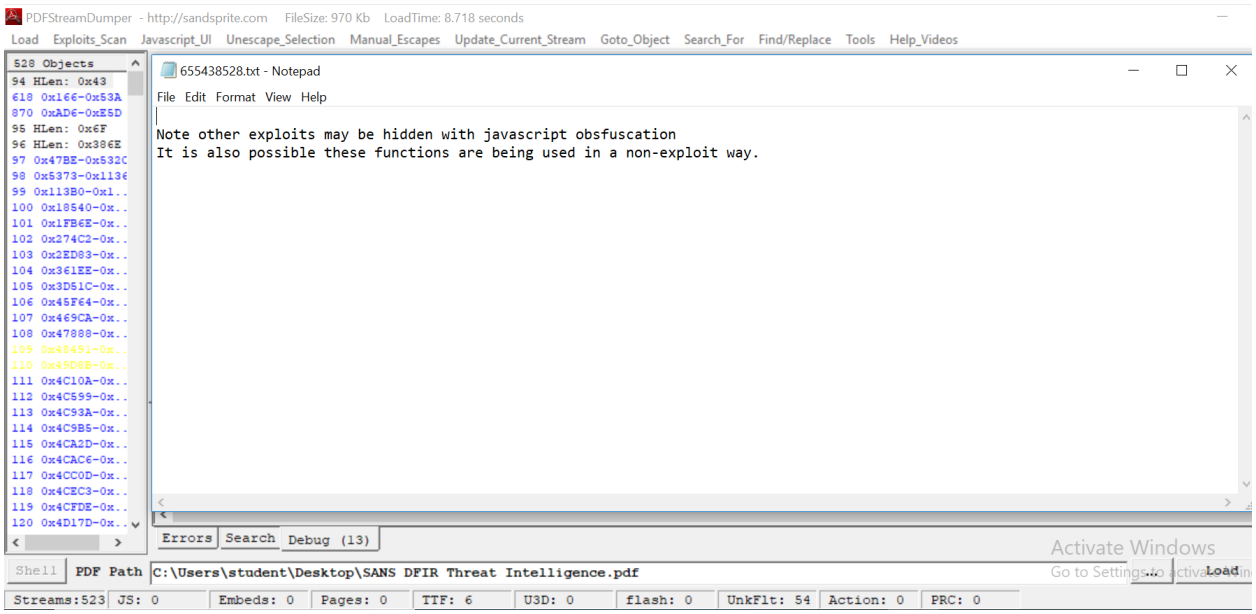




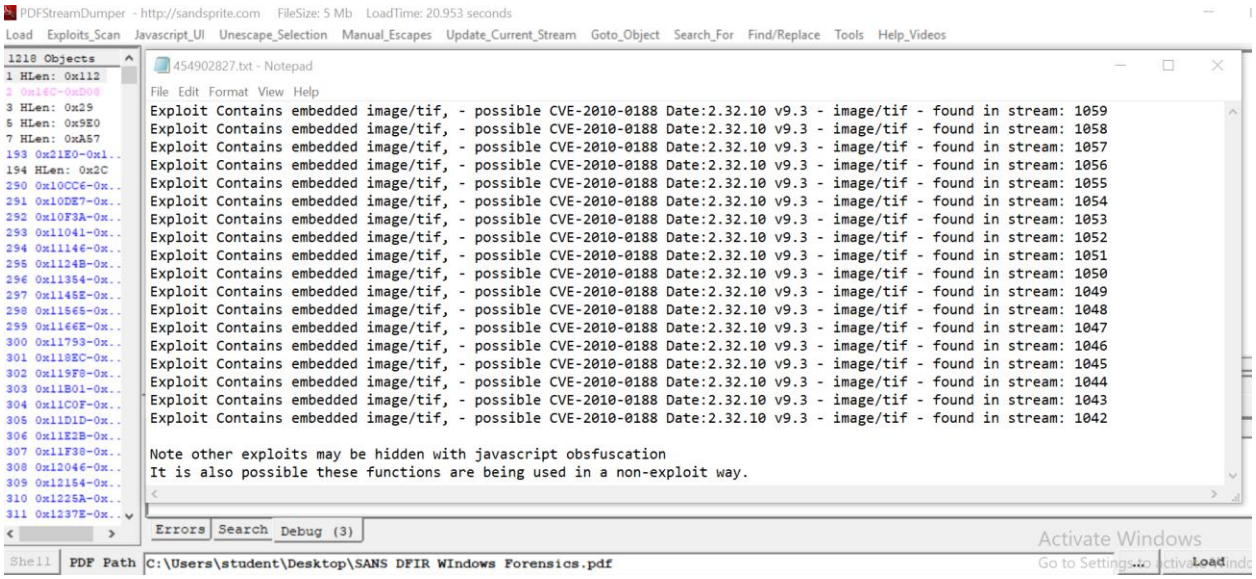
- SANS DFIR Smartphone Forensics.pdf



- SANS DFIR Threat Itelligence.pdf



- SANS DFIR Windows Forensics.pdf



Citations –

1. Ibeakanma, C. (2023, January 18). What Is DLL Hijacking and How to Protect Yourself. MakeUseOf. Retrieved from https://www.makeuseof.com/what-is-dll-hijacking/
2. Microsoft. (2010, June 27). Trojan:Win32/Swrort.A. Microsoft Threats | Malware Encyclopedia. Retrieved from https://www.microsoft.com/en-us/wdsi/threats/malware-encyclopedia-description?Name=Trojan%3AWin32/Swrort.A
3. Ravindran, U. (2021, October 17). PDF injection in simple words. Medium. Retrieved from https://medium.com/@urshilaravindran/pdf-injection-in-simple-words-8c399f92593c
4. Balaji, P. (2020, June 5). PEStudio: Initial Malware Assessment Made Simple. SOC Investigation. Retrieved from https://www.socinvestigation.com/pestudio-initial-malware-assessment-made-simple/
5. Van Impe, K. (2015, June 24). Signature-Based Detection With YARA. Intelligence & Analytics. Retrieved from https://securityintelligence.com/signature-based-detection-with-yara/