Proof-of-Concept Report:-

Tool Name:

PDBReSym & PDFStreamDumper

Description:

PDBReSym is a CLI-based symbolication tool written in Rust that resolves memory addresses from logs (like STrace) into human-readable function names using PDB files.

PDFStreamDumper is a Windows GUI tool for analyzing suspicious PDF files, allowing inspection of embedded scripts, shellcode, and exploit signatures.

What Is This Tool About?

PDBReSym helps reverse engineers and analysts understand crash logs and binary behavior by resolving raw memory addresses.

PDFStreamDumper enables forensic analysts to dissect malicious PDFs and extract embedded threats like JavaScript and shellcode.

Key Characteristics / Features:

PDBReSym:

CLI-based and lightweight

Downloads PDBs from Microsoft Symbol Server

Resolves symbols from logs

Caches PDBs locally

No dependency on Microsoft DIA SDK

Works with STrace logs

Modular architecture

PDFStreamDumper:

GUI-based PDF object viewer

JavaScript interpreter

Shellcode emulator (LibEmu)

Exploit signature scanner

Supports multiple decoding filters

Flash ActionScript decompiler

Save raw and decoded streams

Types / Modules Available:

PDBReSym:

Log Parser

Symbol Resolver

PDB Cache Manager

PDFStreamDumper:

Object Explorer

JavaScript Analyzer

Shellcode Emulator

Exploit Scanner

Stream Decoder

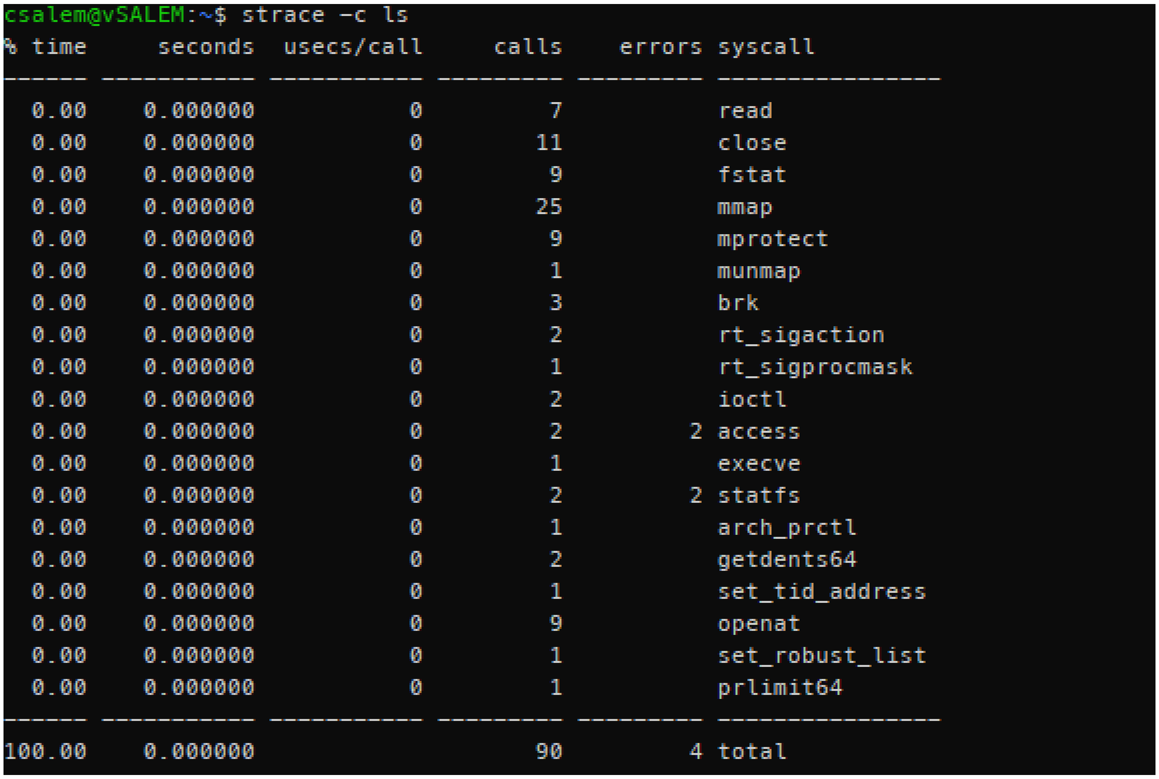
How Will This Tool Help?

PDBReSym helps in reverse engineering and debugging by converting raw logs into readable symbols.

PDFStreamDumper aids in malware analysis by exposing hidden threats in PDF files.

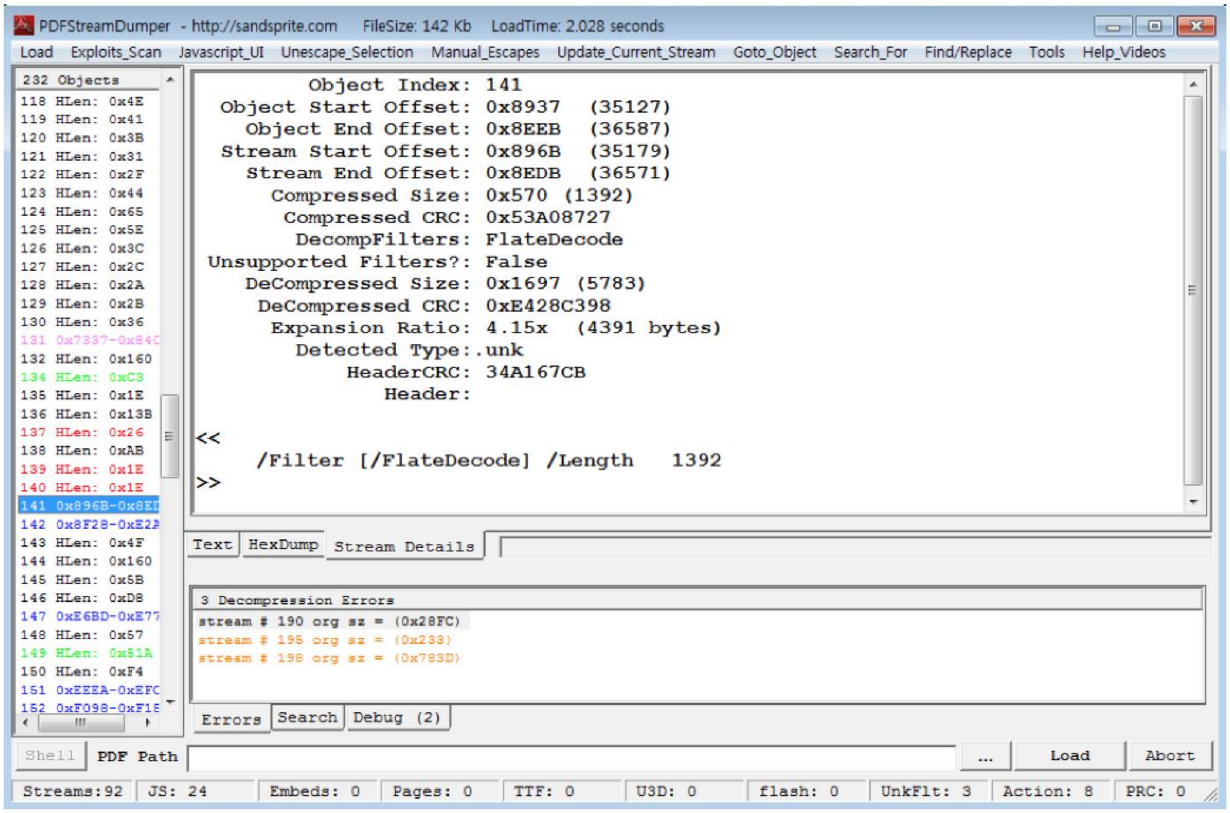
Proof of Concept (POC) Images:

PDBReSym resolving symbols from STrace logs



```
csalem@vSALEM:~$ strace -c ls
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
  0.00    0.000000           0         7           read
  0.00    0.000000           0        11           close
  0.00    0.000000           0         9           fstat
  0.00    0.000000           0        25           mmap
  0.00    0.000000           0         9           mprotect
  0.00    0.000000           0         1           munmap
  0.00    0.000000           0         3           brk
  0.00    0.000000           0         2           rt_sigaction
  0.00    0.000000           0         1           rt_sigprocmask
  0.00    0.000000           0         2           ioctl
  0.00    0.000000           0         2         2 access
  0.00    0.000000           0         1           execve
  0.00    0.000000           0         2         2 statfs
  0.00    0.000000           0         1           arch_prctl
  0.00    0.000000           0         2           getdents64
  0.00    0.000000           0         1           set_tid_address
  0.00    0.000000           0         9           openat
  0.00    0.000000           0         1           set_robust_list
  0.00    0.000000           0         1           prlimit64
------ ----------- ----------- --------- --------- ----------------
100.00    0.000000                    90         4 total
```

PDFStreamDumper showing PDF object tree



```
232 Objects
118 HLen: 0x4E
119 HLen: 0x41
120 HLen: 0x3B
121 HLen: 0x31
122 HLen: 0x2F
123 HLen: 0x44
124 HLen: 0x65
125 HLen: 0x5E
126 HLen: 0x3C
127 HLen: 0x2C
128 HLen: 0x2A
129 HLen: 0x2B
130 HLen: 0x36
131 0x7337-0x84C
132 HLen: 0x160
134 HLen: 0xC3
135 HLen: 0x1E
136 HLen: 0x13B
137 HLen: 0x26
138 HLen: 0xAB
139 HLen: 0x1E
140 HLen: 0x1E
141 0x896B-0x8ED
142 0x8F28-0xE2A
143 HLen: 0x4F
144 HLen: 0x160
145 HLen: 0x5B
146 HLen: 0xD8
147 0xE6BD-0xE77
148 HLen: 0x57
149 HLen: 0x51A
150 HLen: 0xF4
151 0xEEEA-0xEFC
152 0xF098-0xF15
```

```
          Object Index: 141
   Object Start Offset: 0x8937  (35127)
     Object End Offset: 0x8EEB  (36587)
   Stream Start Offset: 0x896B  (35179)
     Stream End Offset: 0x8EDB  (36571)
       Compressed Size: 0x570 (1392)
        Compressed CRC: 0x53A08727
         DecompFilters: FlateDecode
Unsupported Filters?: False
     DeCompressed Size: 0x1697 (5783)
      DeCompressed CRC: 0xE428C398
       Expansion Ratio: 4.15x   (4391 bytes)
         Detected Type:.unk
             HeaderCRC: 34A167CB
                Header:

<<
     /Filter [/FlateDecode] /Length   1392
>>
```

```
3 Decompression Errors
stream # 190 org sz = (0x28FC)
stream # 195 org sz = (0x233)
stream # 198 org sz = (0x783D)
```

```
Streams:92  JS: 24  Embeds: 0  Pages: 0  TTF: 0  U3D: 0  flash: 0  UnkFlt: 3  Action: 8  PRC: 0
```

15-Liner Summary:

PDBReSym resolves memory addresses using PDBs

CLI-based and Rust-powered

Works with STrace logs

No Microsoft SDK dependency

Caches PDBs for offline use

PDFStreamDumper analyzes malicious PDFs

GUI-based with deep inspection tools

Supports JavaScript and shellcode analysis

Detects known exploits and CVEs

Useful in phishing and malware investigations

Supports forensic workflows

Ideal for reverse engineers and analysts

Open-source and portable

Lightweight and fast

Great for red/blue team operations

Time to Use / Best Case Scenarios:

PDBReSym

During reverse engineering

While debugging crash logs

Offline symbol resolution

PDFStreamDumper:

When analyzing phishing PDFs

During incident response

For malware triage and training

When to Use During Investigation:

Reverse engineering malware

Tracing method calls in logs

PDF-based phishing analysis

Shellcode inspection

Exploit detection in documents

Best Person to Use This Tool & Required Skills:

Best Users:

Reverse Engineers

Malware Analysts

Forensic Investigators

Required Skills:

Familiarity with CLI and scripting

Understanding of PE/PDB formats

PDF structure and exploit behavior

Debugging and profiling experience

Flaws / Suggestions to Improve:

PDBReSym:

No GUI interface

Limited to symbolication only

Requires technical setup

PDFStreamDumper:

Windows-only

May miss zero-day exploits

No sandboxing or AV integration

Good About the Tool:

PDBReSym:

Fast and efficient

Works offline

No SDK dependency

PDFStreamDumper:

Deep PDF inspection

Visual and interactive

Great for forensic workflows