Sandboxed Memory Inspector

Language: C++
Focus Area: Systems Programming, Windows API, Binary Analysis

Overview

This Sandboxed Memory Inspector is a command-line tool built in C++ for inspecting and analyzing binary files. It allows users to view, interpret, and summarize the internal structure of files at the byte level, similar to how a low-level debugger or hex editor operates.

The project was created to explore Windows API programming and deepen understanding of memory management, file systems, and binary data representation.

Features

- **File Type Detection:** Automatically identifies common file formats using a byte signature.
- **Offset Navigation:** Allows navigating through specific sections of large files with CLI arguments (--offset, --limit).
- **Block Interpretation:** Displays binary blocks in structured hexadecimal and ASCII form for easy analysis.
- **Summary Reporting:** Generates frequency statistics of bytes in a file to highlight data patterns.

- Utilizes Windows API for low-level file operations.
- Written with safe and modular C++, focusing on clarity, reliability, and extendability.
- Implemented multiple analytical features entirely through manual byte-level parsing, without external libraries.
- Designed to serve as a foundation for future reverse engineering tools.

Learning Outcomes

Through this project, I gained hands-on experience with:

- File and memory manipulation using the Windows API.
- Binary data representation and hex interpretation.
- CLI design and argument parsing for real-world developer tools.

Screenshots

```
C:\Coding\Projects\hexViewer>shexviewer.exe portfolio.png
File type detected: PNG
File size: 881254 bytes
Zero bytes: 23849
Printable ASCII bytes: 292255
Top 5 most frequent bytes:
  0x00: 23849 times
  0x02: 16936 times
  0x40: 15605 times
  0x04: 15519 times
  0x08: 15314 times

00000000: 89 50 4E 47 0D 0A 1A 0A 00 00 00 0D 49 48 44 52   .PNG........IHDR
00000010: 00 00 09 FF 00 00 0F D2 08 06 00 00 00 08 3F 58   ..............?X
00000020: 4E 00 00 20 00 49 44 41 54 78 5E EC DD 79 94 B6   N.. .IDATx^..y..
00000030: 79 5D DF F9 5F D5 BD 56 D5 B3 F6 06 0D B6 46 68   y]..._..V......Fh
00000040: 05 45 10 13 23 8A 60 10 30 6A 12 95 91 09 46 A3   .E..#.`.0j....F.
00000050: 22 0A 62 46 27 E7 C4 59 8E 99 39 27 E3 E4 4C 4E   ".bF'..Y..9'..LN
00000060: CE 99 9C E8 C9 8C 59 08 1A 08 6E 91 19 45 A2 86   ......Y...n..E..
00000070: B8 35 B2 C6 2D 46 01 05 64 11 68 9A AD 97 67 AF   .5..-F..d.h...g.
00000080: AA 7B A9 AA F9 BC BF BF BB 1A 50 98 80 F4 D3 3C   .{........P....<
00000090: 77 F7 FB 7E BA BA AA EE FB 5A 7E D7 EB BA 9E 96   w..~.....Z~.....
000000A0: 3F 3E 7E BE 1B E3 1B 1E 7E D4 7C 29 A0 80 02 0A   ?>~.....~.|)....
000000B0: 28 A0 80 02 0A 28 A0 80 02 0A 28 A0 80 02 0A 28   (....(....(....(
000000C0: A0 80 02 0A 28 A0 80 02 0A 28 A0 80 02 0A 28 A0   ....(....(....(.
000000D0: 80 02 0A 28 A0 80 02 0A 28 A0 C0 5A 08 CC EF BA    ...(....(..Z....
000000E0: 63 63 C3 F0 DF 5A DC 2B 17 A9 80 02 0A 28 A0 80   cc...Z.+.....(..
000000F0: 02 0A 28 A0 80 02 0A 28 A0 80 02 0A 28 A0 80 02   ..(....(....(...
```

```
C:\Coding\Projects\hexViewer>hexViewer.exe
Usage: .\hexviewer.exe <file_path> [--offset where_to_start_hex_dump] [--limit how_many_bytes] [--inspect starting_offset]
[--length how_many_bytes_to_inspect] [--type help]
```

```
C:\Coding\Projects\hexViewer>hexViewer.exe aerials.txt --inspect 0x15 --length 22 --type ascii
File size: 240 bytes
Zero bytes: 0
Printable ASCII bytes: 228
Top 5 most frequent bytes:
  0x20: 40 times
  0x65: 26 times
  0x61: 21 times
  0x6E: 15 times
  0x74: 14 times

00000000: 4C 69 66 65 20 69 73 20 61 20 77 61 74 65 72 66   Life is a waterf
00000010: 61 6C 6C 0D 0A 57 65 27 72 65 20 6F 6E 65 20 69   all..We're one i
00000020: 6E 20 74 68 65 20 72 69 76 65 72 0D 0A 41 6E 64   n the river..And
00000030: 20 6F 6E 65 20 61 67 61 69 6E 20 61 66 74 65 72    one again after
00000040: 20 74 68 65 20 66 61 6C 6C 0D 0A 53 77 69 6D 6D    the fall..Swimm
00000050: 69 6E 67 20 74 68 72 6F 75 67 68 20 74 68 65 20   ing through the
00000060: 76 6F 69 64 2C 20 77 65 20 68 65 61 72 20 74 68   void, we hear th
00000070: 65 20 77 6F 72 64 0D 0A 57 65 20 6C 6F 73 65 20   e word..We lose
00000080: 6F 75 72 73 65 6C 76 65 73 2C 20 62 75 74 20 77   ourselves, but w
00000090: 65 20 66 69 6E 64 20 69 74 20 61 6C 6C 0D 0A 27   e find it all..'
000000A0: 43 61 75 73 65 20 77 65 20 61 72 65 20 74 68 65   Cause we are the
000000B0: 20 6F 6E 65 73 20 74 68 61 74 20 77 61 6E 6E 61    ones that wanna
000000C0: 20 70 6C 61 79 0D 0A 41 6C 77 61 79 73 20 77 61    play..Always wa
000000D0: 6E 6E 61 20 67 6F 2C 20 62 75 74 20 79 6F 75 20   nna go, but you
000000E0: 6E 65 76 65 72 20 77 61 6E 6E 61 20 73 74 61 79   never wanna stay
ASCII: We're one in the river
```

```
C:\Coding\Projects\hexViewer>hexViewer.exe aerials.txt --type help
Supported types for --type:
  ascii   : Printable ASCII characters
  uint8   : 8-bit unsigned integer
  uint16  : 16-bit unsigned integer
  uint32  : 32-bit unsigned integer
  int8    : 8-bit signed integer
  int16   : 16-bit signed integer
  int32   : 32-bit signed integer
  float   : 4-byte floating point
  double  : 8-byte floating point
```