Windows Memory Dump Analysis

Advanced

with Data Structures

Version 3.0

Dmitry Vostokov
Software Diagnostics Services
Prerequisites

Basic and intermediate level
Windows memory dump analysis
Training Goals

- Use UML for communication
- Learn fundamentals of device drivers
- Learn specialized analysis techniques and commands in the context of x64 complete memory dumps
- Learn how to navigate data structures such as linked lists and arrays
Training Principles

- Talk only about what I can show
- Lots of pictures
- Lots of examples
- Original content and examples
Practice Exercises
Links

- Memory Dumps:

  Not available in preview version

- Exercise Transcripts:

  Not available in preview version
Exercise 0

- **Goal:** Install Debugging Tools for Windows and learn how to set up symbols correctly

- **Patterns:** Incorrect Stack Trace

- \AdvWMDA-Dumps\Exercise-0-Download-Setup-WinDbg.pdf

- \AdvWMDA-Dumps\Exercise-Legacy.0-Download-Setup-WinDbg.pdf
Complete Memory Dumps

Exercises C1-C11
Exercise C1

- **Goal:** Learn how to get stack traces related to sessions, processes and threads; diagnose different thread types; get stack traces from WOW64 processes

- **Patterns:** Stack Trace Collection (unmanaged space); Passive Thread; Coupled Processes (weak); Coupled Processes (strong); Wait Chain (ALPC); Virtualized Process; Truncated Stack Trace

- \AdvWMDA-Dumps\Exercise-C1-Stack-Trace-Collection-64.pdf

- \AdvWMDA-Dumps\Exercise-Legacy.C1-Stack-Trace-Collection-64.pdf
Exercise C2

- **Goal:** Learn how to assemble code and evaluate expressions in WinDbg; recognize byte ordering conventions; search memory for specific values

- **Patterns:** Value References

  - `\AdvWMDA-Dumps\Exercise-C2-Memory-Search-64.pdf`

  - `\AdvWMDA-Dumps\Exercise-Legacy.C2-Memory-Search-64.pdf`
Exercise C3

- **Goal:** Learn how to navigate linked lists
- **Patterns:** Module Variable

- \AdvWMDA-Dumps\Exercise-C3-Linked-Lists-64.pdf
- \AdvWMDA-Dumps\Exercise-Legacy.C3-Linked-Lists-64.pdf
Linked List

DT
+fieldA
+fieldB
+next : DT *
+prev : DT *
+fieldC

Object1 : DT
fieldA
fieldB
next : DT *
prev : DT *
fieldC

Object2 : DT
fieldA
fieldB
next : DT *
prev : DT *
fieldC

Object3 : DT
fieldA
fieldB
next : DT *
prev : DT *
fieldC

© 2017 Software Diagnostics Services
Linked Data Structures

DT
+fieldA
+fieldB
+DTLinks : _LIST_ENTRY
+fieldC

Object1 : DT
fieldA
fieldB
DTLinks : _LIST_ENTRY
fieldC

Object2 : DT
fieldA
fieldB
DTLinks : _LIST_ENTRY
fieldC

DTHead : _LIST_ENTRY
Flink : _LIST_ENTRY *
Blink : _LIST_ENTRY *

+DTLinks : _LIST_ENTRY
Flink : _LIST_ENTRY *
Blink : _LIST_ENTRY *

+DTLinks : _LIST_ENTRY
Flink : _LIST_ENTRY *
Blink : _LIST_ENTRY *

© 2017 Software Diagnostics Services
Exercise C4A

- **Goal:** Learn how to create scripts to extend WinDbg functionality (via built-in scripting)

- **Patterns:** Spiking Thread; Thread Waiting Time; Stack Trace Collection (predicate)

- \AdvWMDA-Dumps\Exercise-C4A-Scripting-64.pdf

- \AdvWMDA-Dumps\Exercise-Legacy.C4-Scripting-64.pdf
Exercise C4B

- **Goal:** Learn how to create scripts to extend WinDbg functionality (via JavaScript scripting)

- **Patterns:** Spiking Thread; Thread Waiting Time; Stack Trace Collection (predicate)

- `\AdvWMDA-Dumps\Exercise-C4B-Scripting-64.pdf`
Exercise C5

- **Goal:** Learn how to inspect registry

- **Patterns:** Self-Diagnosis

- \AdvWMDA-Dumps\Exercise-C5-Registry-64.pdf

- \AdvWMDA-Dumps\Exercise-Legacy.C5-Registry-64.pdf
Exercise C6

- **Goal:** Learn how to inspect module (including system/kernel) variables and check them with extension command output

- **Patterns:** Module Variable

  - \AdvWMDA-Dumps\Exercise-C6-ModuleVariables-64.pdf

  - \AdvWMDA-Dumps\Exercise-Legacy.C6-ModuleVariables-64.pdf
Exercise C7

- **Goal:** Learn how to inspect various system (kernel) objects
- **Patterns:** System Object

- `\AdvWMDA-Dumps\Exercise-C7-SystemObjects-64.pdf`
- `\AdvWMDA-Dumps\Exercise-Legacy.C7-SystemObjects-64.pdf`
Exercise C8

- **Goal:** Learn how to inspect network protocols and adapters
- **Patterns:** Disconnected Network Adapter

- \AdvWMDA-Dumps\Exercise-C8-Network-64.pdf
- \AdvWMDA-Dumps\Exercise-Legacy.C8-Network-64.pdf
Exercise C9

- **Goal:** Learn how to inspect IRP, file, device and driver objects; dump arrays

- **Patterns:** Stack Trace (I/O requests); Stack Trace (I/O devices)

- \AdvWMDA-Dumps\Exercise-C9-Device-Drivers-64.pdf

- \AdvWMDA-Dumps\Exercise-Legacy.C9-Device-Drivers-64.pdf
Device Driver

- A pluggable component for a device or several devices
- Creates device objects and symbolic links to them
- Provides entry points for I/O operations including IOCTL interface (I/O Control - used for any purpose)
- Implemented as a C structure with data and pointers to functions

Diagram:
- Driver
  - Dispatch
  - \Driver\ <Name>
  - \FileSystem\ <Name>
  - Driver(Init, Unload)
Device Driver Example

3: kd> !drvobj \Driver\Beep 3
Driver object (ffffe000ea9309c0) is for:
    \Driver\Beep
Driver Extension List: (id, addr)

Device Object list:
    fffff000eac2c990

DriverEntry:         fffff801c02e6000     Beep!GsDriverEntry
DriverStartIo:       fffff801c02e16c0     Beep!BeepStartIo
DriverUnload:        fffff801c02e1760     Beep!BeepUnload
AddDevice:           00000000

Dispatch routines:
[00] IRP_MJ_CREATE                      fffff801c02e1430     Beep!BeepOpen
[01] IRP_MJ_CREATE_NAMED_PIPE           fffff8014875ad94     nt!IopInvalidDeviceRequest
[02] IRP_MJ_CLOSE                       fffff801c02e1150     Beep!BeepClose
[03] IRP_MJ_READ                        fffff8014875ad94     nt!IopInvalidDeviceRequest
[04] IRP_MJ_WRITE                       fffff8014875ad94     nt!IopInvalidDeviceRequest
[05] IRP_MJ_QUERY_INFORMATION           fffff8014875ad94     nt!IopInvalidDeviceRequest
[...]  
[0c] IRP_MJ_DIRECTORY_CONTROL           fffff8014875ad94     nt!IopInvalidDeviceRequest
[0d] IRP_MJ_FILE_SYSTEM_CONTROL         fffff8014875ad94     nt!IopInvalidDeviceRequest
[0e] IRP_MJ_DEVICE_CONTROL              fffff801c02e1200     Beep!BeepDeviceControl
[0f] IRP_MJ_INTERNAL_DEVICE_CONTROL     fffff8014875ad94     nt!IopInvalidDeviceRequest
[10] IRP_MJ_SHUTDOWN                    fffff8014875ad94     nt!IopInvalidDeviceRequest
[...]  
[1b] IRP_MJ_PNP                         fffff8014875ad94     nt!IopInvalidDeviceRequest
Devices

- Represents physical or logical device (\Device\MousePad)
- Target of an I/O operation
- Name: \Device\<Name> or \FileSystem\<Name>
- Implemented as a C structure

![Diagram of device and driver relationships]

© 2017 Software Diagnostics Services
I/O Manager

- Provides an interface between drivers and OS
- Defines a detailed framework and specification for device drivers
- Provides support functions to drivers
- Packet-driven architecture: each I/O operation is described by IRP (I/O Request Packet) structure
Big Picture

IRP * = IoAllocateIrpo(…)
IoCallDriver(DEVICE_OBJECT *, IRP *)

ntkrnlmp.exe
IRP
Driver.sys

Kernel Mode/Space

Device.sys

User Mode/Space

ntdll.dll
kernel32.dll
Application.exe

© 2017 Software Diagnostics Services
Exercise C10

- **Goal:** Learn how to inspect storage device queues and file system filter stack traces

- **Patterns:** Disk Packet Buildup; Stack Trace (file system filters)

- \\AdvWMDA-Dumps\Exercise-C10-Storage-File-System-Filters-64.pdf
Exercise C11

- **Goal**: Learn how to manually analyze raw stack to mine for missing information

- **Patterns**: Main Thread; Wait Chain (window messaging); Execution Residue; Hidden Parameter; Value References; Data Correlation

- \AdvWMDA-Dumps\Exercise-C11-Window-Messaging-64.pdf
Pattern Links

Stack Trace Collection (unmanaged space)
Coupled Processes (weak)
Value References
Spiking Thread
Stack Trace Collection (predicate)
Invalid Pointer
Disconnected Network Adapter
Stack Trace Collection (I/O requests)
Wait Chain (window messaging)
Hidden Parameter
Wait Chain (ALPC)
Incorrect Stack Trace
Stack Trace (I/O requests)
Stack Trace (file system filters)

Passive Thread
Virtualized Process
Module Variable
Thread Waiting Time
Self-Diagnosis
System Object
Historical Information
Main Thread
Execution Residue
Data Correlation
Coupled Processes (strong)
Truncated Stack Trace
Stack Trace (I/O devices)
Disk Packet Buildup

© 2017 Software Diagnostics Services
Pattern Case Studies

70 multiple pattern case studies:

http://www.dumpanalysis.org/blog/index.php/pattern-cooperation/

Pattern Interaction chapters in Memory Dump Analysis Anthology

Hunting for a Driver
Resources

- WinDbg Help / WinDbg.org (quick links) / DumpAnalysis.org
- Debugging.TV / YouTube.com/DebuggingTV
- UML Distilled
- Windows NT Device Driver Development (OSR)
- Developing Windows NT Device Drivers: A Programmer’s Handbook
- Programming the Microsoft Windows Driver Model, 2nd ed.
- Windows Internals, 6th ed.
- Memory Dump Analysis Anthology (Volumes 1 – 10)
Going Further

More basic:

- Accelerated Windows Memory Dump Analysis, 4th edition

Special topics:

- Practical Foundations of Windows Debugging, Disassembling, Reversing
- Accelerated Windows Malware Analysis with Memory Dumps
- Accelerated Disassembly, Reconstruction and Reversing
- Accelerated .NET Memory Dump Analysis, 2nd edition
- Accelerated Windows Debugging³
Q&A

Please send your feedback using the contact form on PatternDiagnostics.com
Thank you for attendance!