Complete
Crash and Hang
Memory Dump Analysis

Fundamentals

Revision 2

Presenter: Dmitry Vostokov
Memory Dump Analysis Services

Prerequisites

To Be Discussed Later

We use these boxes to introduce useful vocabulary to be discussed in later slides

Working knowledge of:

- WinDbg (installation, symbols)
- Basic user process dump analysis
- Basic kernel memory dump analysis

Agenda (Summary)

- Basics
- Patterns
- Exercise
- Guide

Agenda (Basics)

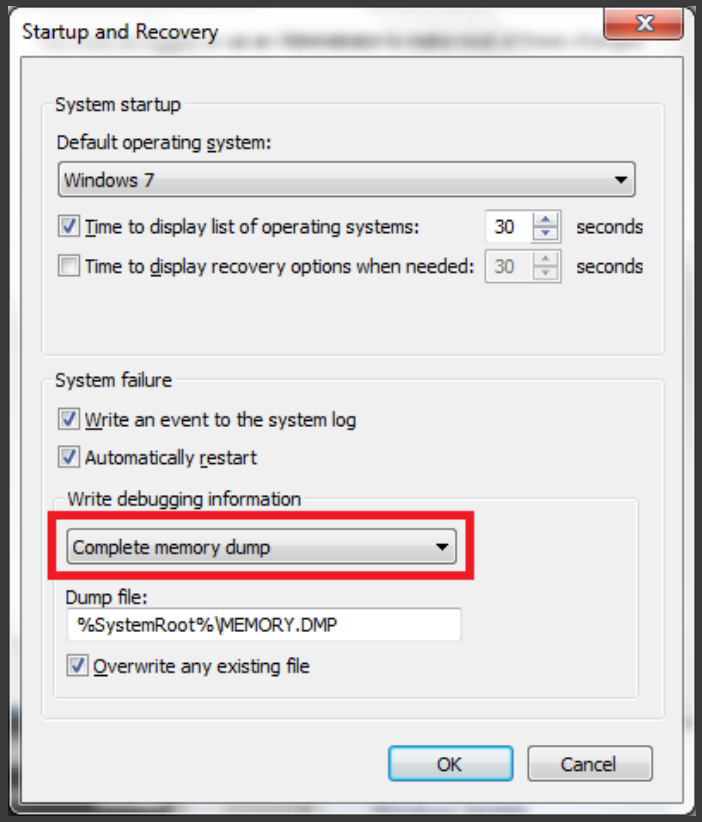
- Dump generation
- Memory spaces
- Major challenges
- Common commands

Dump Generation

To Be Discussed Later

Truncated Dump pattern
Manual Dump pattern

- Control Panel \ System \ Advanced system settings \ Startup and Recovery
- Page file size should be greater than the amount of physical memory by a few MB
- For small system partitions or virtual disk systems: DedicatedDumpFile (KB969028)

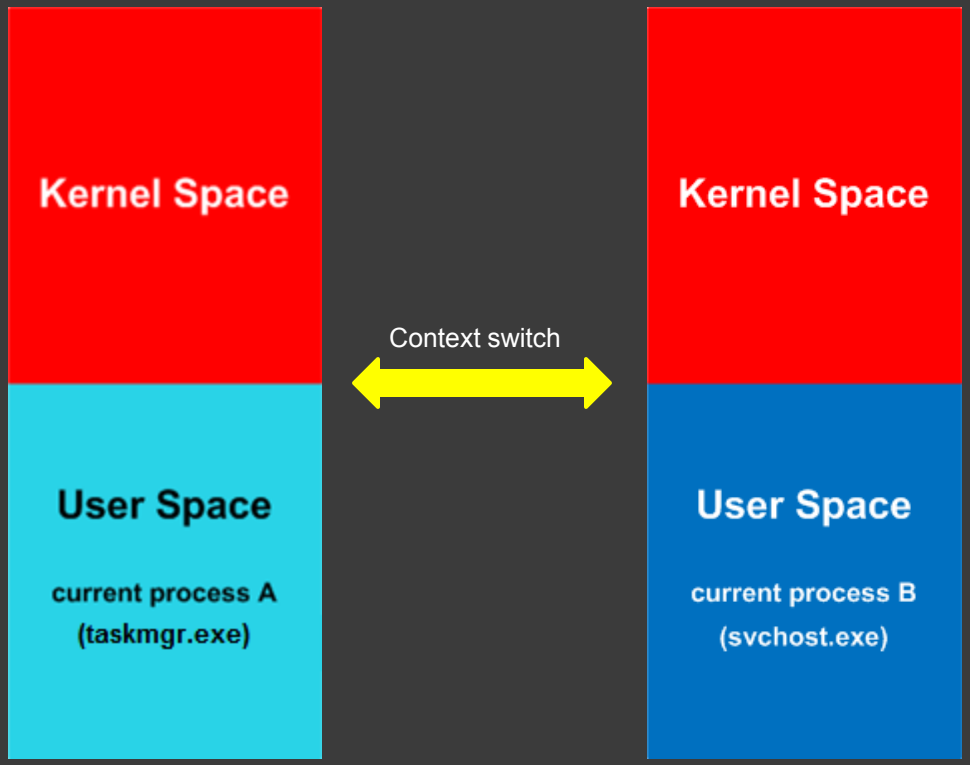


Troubleshooting note:

HKLM \ SYSTEM \ CurrentControlSet \ Control \ CrashControl
CrashDumpEnabled = 1 (DWORD)

Memory Spaces

- Complete memory == Physical memory
- We always see the current process space



To Be Discussed Later

WinDbg command to switch to a different process context:

.process

Major Challenges

To Be Discussed Later

WinDbg extension command to dump all stack traces:

!process 0 ff

- Vast memory space to search
- Multiple processes (user spaces) to examine
- User space view needs to be correct when we examine another thread
- Huge file size (x64)

To Be Discussed Later

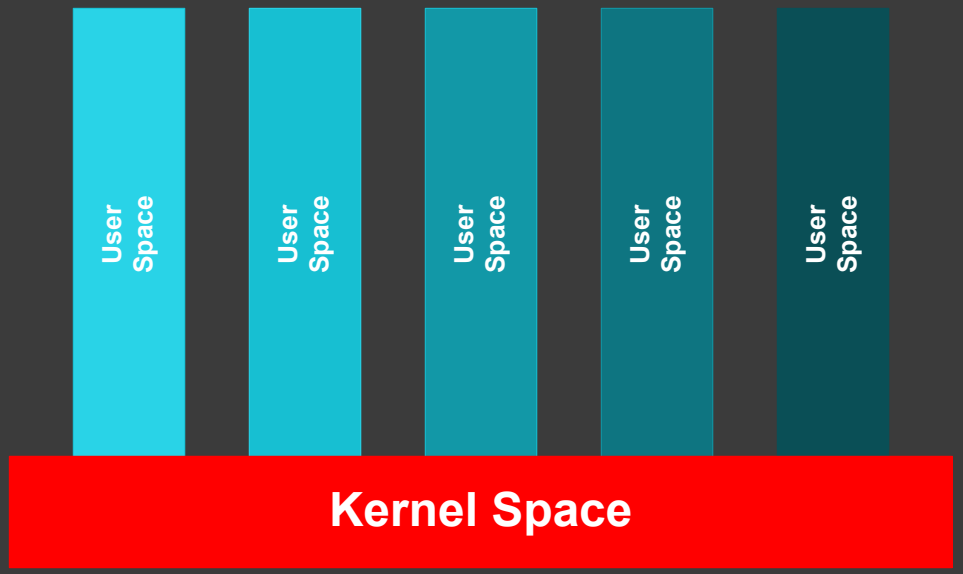
Wait Chain patterns

Fiber Bundles

The name borrowed from mathematics (topology)

Problem: mild freeze of a 64GB memory system

Solution: dump domain specific processes and generate a kernel memory dump



Common Commands

- **.logopen <file>**
  Opens a log file to save all subsequent output
- View commands
  Dump everything or selected processes and threads (context changes automatically)
- Switch commands
  Switch to a specific process or thread for a fine-grain analysis

View Commands

- **!process 0 ff**
  Lists all processes (including times, environment, modules) and their thread stack traces
- **!process 0 1f**
  The same as the previous command but without PEB information (more secure)
- **!process <address> ff or !process <address> 1f**
  The same as the previous commands but only for an individual process
- **!thread <address> ff**
  Shows thread information and stack trace
- **!thread <address> f6**
  The same as the previous command but shows the first 3 parameters for every function

Switch Commands

To Be Discussed Later

x86 stack trace from WOW64 process:

.thread /w

- **.process /r /p <address>**

  Switches to a specified process. Its context becomes current. Reloads symbol files for user space. Now we can use commands like !cs

```
0: kd> .process /r /p fffffa80044d8b30
Implicit process is now fffffa80`044d8b30
Loading User Symbols
...............................
```

- **.thread <address>**

  Switches to a specified thread. Assumes the current process context

  Now we can use commands like k*

- **.thread /r /p <address>**

  The same as the previous command but makes the thread process context current and reloads symbol files for user space:

```
0: kd> .thread /r /p fffffa80051b7060
Implicit thread is now fffffa80`051b7060
Implicit process is now fffffa80`044d8b30
Loading User Symbols
...............................
```

Agenda (Patterns)

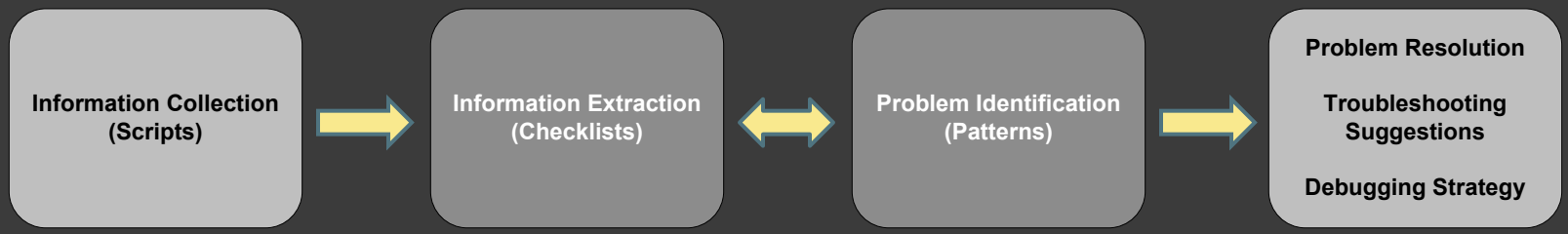
- Pattern-driven analysis
- Pattern classification
- Pattern examples
- Common mistakes

Pattern-driven Analysis

To Be Discussed Later

CARE
Crash Analysis Report Environment

Pattern: a common recurrent identifiable problem together with a set of recommendations and possible solutions to apply in a specific context



Note: we do not discuss BSOD crashes here as most of the time kernel memory dumps are sufficient for analysis

Pattern Classification

- Blocked threads
- Wait chains
- Resource consumption
- Corruption signs
- Special processes

Example: Blocked Thread

```
THREAD fffffa80097f1660  Cid 154c.1570  Teb: 000007fffffd4000 Win32Thread: fffff900c06a83f0 WAIT: (WrUserRequest) UserMode Non-Alertable
    fffffa800447e8f0  SynchronizationEvent
Not impersonating
DeviceMap                 fffff8a001ce6b90
Owning Process            fffffa8004451060       Image:         ApplicationA.exe
Attached Process          N/A            Image:         N/A
Wait Start TickCount      22248          Ticks: 47 (0:00:00:00.733)
Context Switch Count      340                 LargeStack
UserTime                  00:00:00.015
KernelTime                00:00:00.000
Win32 Start Address ApplicationA (0x000000013f2c1210)
Stack Init fffff8800ec25c70 Current fffff8800ec25730
Base fffff8800ec26000 Limit fffff8800ec1d000 Call 0
Priority 11 BasePriority 8 UnusualBoost 0 ForegroundBoost 2 IoPriority 2 PagePriority 5
Child-SP          RetAddr           Call Site
fffff880`0ec25770 fffff800`02ee6f32 nt!KiSwapContext+0x7a
fffff880`0ec258b0 fffff800`02ee974f nt!KiCommitThreadWait+0x1d2
fffff880`0ec25940 fffff960`0013bc97 nt!KeWaitForSingleObject+0x19f
fffff880`0ec259e0 fffff960`0013bd39 win32k!xxxRealSleepThread+0x257
fffff880`0ec25a80 fffff960`0014e7a6 win32k!xxxSleepThread+0x59
fffff880`0ec25ab0 fffff800`02ee0ed3 win32k!NtUserWaitMessage+0x46
fffff880`0ec25ae0 00000000`7709933a nt!KiSystemServiceCopyEnd+0x13 (TrapFrame @ fffff880`0ec25ae0)
00000000`0275f308 00000000`770a4bc4 USER32!ZwUserWaitMessage+0xa
00000000`0275f310 00000000`770a4edd USER32!DialogBox2+0x274
00000000`0275f3a0 00000000`770f2920 USER32!InternalDialogBox+0x135
00000000`0275f400 00000000`770f1c15 USER32!SoftModalMessageBox+0x9b4
00000000`0275f530 00000000`770f146b USER32!MessageBoxWorker+0x31d
00000000`0275f6f0 00000000`770f1362 USER32!MessageBoxTimeoutW+0xb3
>>>  00000000`0275f7c0 00000001`3f2c1089 USER32!MessageBoxW+0x4e
00000000`0275f800 00000000`02135fd0 ApplicationA+0x1089
00000000`0275f808 00000000`00000000 0x2135fd0
```

To Be Discussed Later

Complete Dump Analysis Exercise

Example: Wait Chain

To Be Discussed Later

Complete Dump Analysis Exercise

```
 THREAD fffffa8004514060  Cid 07f4.1470  Teb: 000007fffffae000 Win32Thread: 0000000000000000 WAIT: (UserRequest) UserMode Non-Alertable
>>>         fffffa80044c53c0  Mutant - owning thread fffffa8004569750
        Not impersonating
        DeviceMap                 fffff8a001ce6b90
        Owning Process            fffffa8004546060       Image:         ApplicationC.exe
        Attached Process          N/A            Image:         N/A
        Wait Start TickCount      14474          Ticks: 7821 (0:00:02:02.008)
        Context Switch Count      2
        UserTime                  00:00:00.000
        KernelTime                00:00:00.000
        Win32 Start Address ApplicationC (0x000000013f6c12a0)
        Stack Init fffff8800d196c70 Current fffff8800d1967c0
        Base fffff8800d197000 Limit fffff8800d191000 Call 0
        Priority 11 BasePriority 8 UnusualBoost 0 ForegroundBoost 2 IoPriority 2 PagePriority 5
        Child-SP          RetAddr           Call Site
        fffff880`0d196800 fffff800`02ee6f32 nt!KiSwapContext+0x7a
        fffff880`0d196940 fffff800`02ee974f nt!KiCommitThreadWait+0x1d2
        fffff880`0d1969d0 fffff800`031d844e nt!KeWaitForSingleObject+0x19f
        fffff880`0d196a70 fffff800`02ee0ed3 nt!NtWaitForSingleObject+0xde
        fffff880`0d196ae0 00000000`772f135a nt!KiSystemServiceCopyEnd+0x13 (TrapFrame @ fffff880`0d196ae0)
        00000000`0107f708 000007fe`fd9e10dc ntdll!NtWaitForSingleObject+0xa
        00000000`0107f710 00000001`3f6c112e KERNELBASE!WaitForSingleObjectEx+0x79
        00000000`0107f7b0 00000000`00586570 ApplicationC+0x112e
        00000000`0107f7b8 00000000`00000000 0x586570
```

Example: Consumption

To Be Discussed Later

Complete Dump Analysis Exercise

```
1: kd> !process 0 0
**** NT ACTIVE PROCESS DUMP ****
PROCESS fffffa80042d5400
    SessionId: none  Cid: 0004    Peb: 00000000  ParentCid: 0000
    DirBase: 00187000  ObjectTable: fffff8a0000017e0  HandleCount: 785.
    Image: System

PROCESS fffffa8006fa8750
    SessionId: none  Cid: 0144    Peb: 7fffffda000  ParentCid: 0004
    DirBase: 107226000  ObjectTable: fffff8a0002dcf90  HandleCount:  32.
    Image: smss.exe

PROCESS fffffa80083cdb30
    SessionId: 0  Cid: 01ec    Peb: 7fffffdc000  ParentCid: 01a8
    DirBase: 918c1000  ObjectTable: fffff8a0013b98a0  HandleCount: 679.
    Image: csrss.exe


[...]

PROCESS fffffa800442ab30
    SessionId: 1  Cid: 1418    Peb: 7fffffde000  ParentCid: 0840
>>> DirBase: 11e2c5000  ObjectTable: fffff8a004cc3b50  HandleCount: 20014.
    Image: ApplicationE.exe

[...]
```

Example: Corruption

To Be Discussed Later

Complete Dump Analysis Exercise

```
THREAD fffffa80044e2720  Cid 06fc.15c0  Teb: 000007fffffae000 Win32Thread: 0000000000000000 WAIT: (UserRequest) UserMode Alertable
          fffffa8009a5bb30  ProcessObject
      Not impersonating
      DeviceMap                 fffff8a001ce6b90
      Owning Process            fffffa8004364060       Image:         ApplicationD.exe
      [...]
      Win32 Start Address ApplicationD (0x000000013f061318)
      Stack Init fffff8800ec10c70 Current fffff8800ec107c0
      Base fffff8800ec11000 Limit fffff8800ec0b000 Call 0
      Priority 11 BasePriority 8 UnusualBoost 0 ForegroundBoost 2 IoPriority 2 PagePriority 5
      Child-SP          RetAddr           Call Site
      fffff880`0ec10800 fffff800`02ee6f32 nt!KiSwapContext+0x7a
      fffff880`0ec10940 fffff800`02ee974f nt!KiCommitThreadWait+0x1d2
      fffff880`0ec109d0 fffff800`031d844e nt!KeWaitForSingleObject+0x19f
      fffff880`0ec10a70 fffff800`02ee0ed3 nt!NtWaitForSingleObject+0xde
      fffff880`0ec10ae0 00000000`772f135a nt!KiSystemServiceCopyEnd+0x13 (TrapFrame @ fffff880`0ec10ae0)
      00000000`0071e7f8 00000000`77363062 ntdll!NtWaitForSingleObject+0xa
      00000000`0071e800 00000000`773632a5 ntdll!RtlReportExceptionEx+0x1d2
      00000000`0071e8f0 00000000`7736330a ntdll!RtlReportException+0xb5
      00000000`0071e970 00000000`77364145 ntdll!RtlpTerminateFailureFilter+0x1a
      00000000`0071e9a0 00000000`772b85a8 ntdll!RtlReportCriticalFailure+0x96
      [...]
      00000000`0071ea70 00000000`772b97a8 ntdll!RtlDispatchException+0x45a
      00000000`0071f150 00000000`773640f2 ntdll!RtlRaiseException+0x22f
      00000000`0071fb00 00000000`77364736 ntdll!RtlReportCriticalFailure+0x62
      00000000`0071fbd0 00000000`77365942 ntdll!RtlpReportHeapFailure+0x26
      00000000`0071fc00 00000000`773675f4 ntdll!RtlpHeapHandleError+0x12
      00000000`0071fc30 00000000`7730dc8f ntdll!RtlpLogHeapFailure+0xa4
      00000000`0071fc60 00000000`771a301a ntdll! ?? ::FNODOBFM::`string'+0x10c54
>>>   00000000`0071fce0 00000001`3f061274 kernel32!HeapFree+0xa
      00000000`0071fd10 00000001`3f0610c3 ApplicationD+0x1274
      [...]
```

Example: Special Process

```
1: kd> !vm

[...]

        0458 svchost.exe       1922 (      7688 Kb)
        0bfc iexplore.exe      1891 (      7564 Kb)
        0bf4 msseces.exe       1863 (      7452 Kb)
        0fc8 NisSrv.exe        1845 (      7380 Kb)
        0584 StageRemoteSer    1776 (      7104 Kb)
        05dc spoolsv.exe       1729 (      6916 Kb)
        11e8 mscorsvw.exe      1620 (      6480 Kb)
        0c6c vmware-authd.e    1593 (      6372 Kb)
        02d8 services.exe      1560 (      6240 Kb)
>>>     1134 WerFault.exe      1558 (      6232 Kb)
        0c1c StageRemote.ex    1518 (      6072 Kb)
        135c taskmgr.exe       1513 (      6052 Kb)
        0ea4 WmiPrvSE.exe      1411 (      5644 Kb)
        0350 svchost.exe       1283 (      5132 Kb)
        02e0 lsass.exe         1218 (      4872 Kb)
        03b8 svchost.exe       1214 (      4856 Kb)
        09b0 SftService.exe    1171 (      4684 Kb)
        13fc daemonu.exe       1117 (      4468 Kb)
        0b5c igfxpers.exe      1080 (      4320 Kb)
        080c DTLite.exe        1051 (      4204 Kb)
        1194 DTShellHlp.exe    1008 (      4032 Kb)
        04c8 taskhost.exe       975 (      3900 Kb)
        0b34 STService.exe      933 (      3732 Kb)
        0270 csrss.exe          855 (      3420 Kb)
        029c winlogon.exe       845 (      3380 Kb)
[...]
```

To Be Discussed Later

Complete Dump Analysis Exercise

Common Mistakes

- Not switching to the appropriate context
- Not looking at full stack traces
- Not looking at all stack traces
- Not using checklists
- Not looking past the first found evidence
- Not comparing to the reference debugger output

Note: Listing both x86 and x64 stack traces
http://www.dumpanalysis.org/blog/index.php/2010/02/09/complete-stack-traces-from-x64-system/

Agenda (Exercise)

- Run processes that model abnormal behavior
- Generate a complete memory dump
- Analyze the memory dump

Note: Due to security concerns I'm not making a complete memory dump downloadable. You can generate your own complete memory dump after downloading and running model applications

Exercise: Run Processes

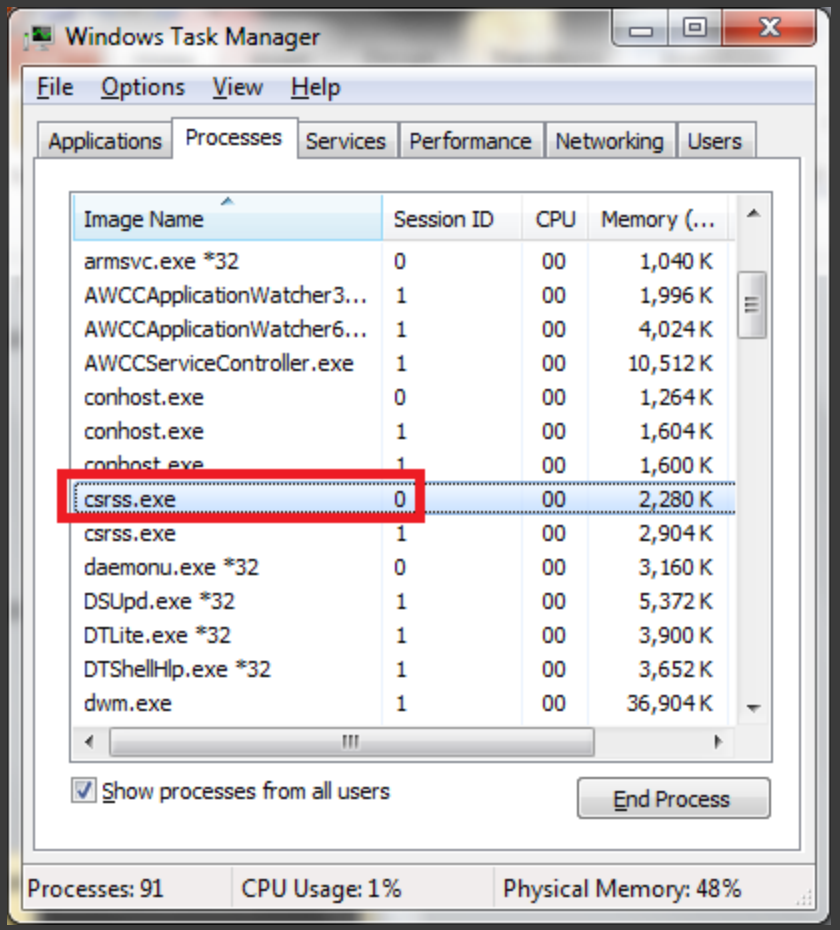
These processes model specific patterns:

ApplicationA , ApplicationB, ApplicationC, ApplicationD, ApplicationE
For demonstration I run x64 versions plus x86 version of ApplicationA

Note: Run applications in alphabetical order

Can be downloaded from this location:
www.DumpAnalysis.com/Training/FreeWebinars/CMDA-Examples.zip
There are x86 and x64 versions

Exercise: Force A Dump



The system is x64 Windows 7

Note: Wait at least 10 seconds after running model applications to have them properly initialize their dependencies

Exercise: Dump Analysis

Now I switch to a WinDbg session...

Agenda (Guide)

- Patterns related to complete memory dumps
- Pattern cooperation case studies from complete memory dumps
- Pattern Map

Pattern Examples

Some patterns that are relevant to complete memory dumps:

Incorrect Symbolic Information
Semantic Split
Paged Out Data
Wait Chain (thread objects)
Wait Chain (LPC/ALPC)
Last Error Collection
Suspended Thread
Coupled Processes (strong)
Truncated Dump
Spiking Thread
Deadlock (critical sections)
Problem Vocabulary
Semantic Structures
Virtualized System

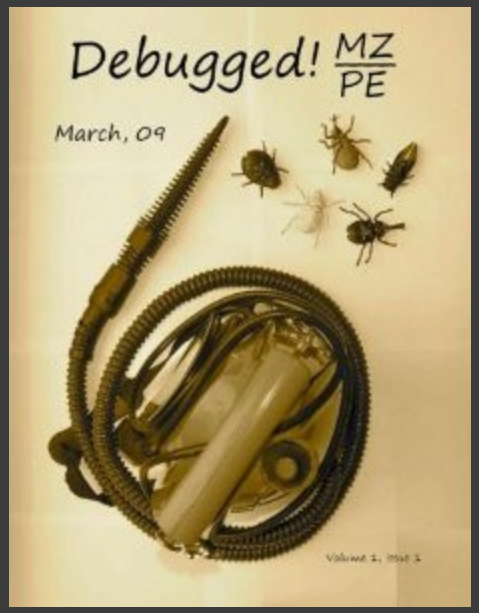
No System Dumps
Message Box
Inconsistent Dump
Wait Chain (critical sections)
Wait Chain (process objects)
Special Process
Historical Information
Stack Trace Collection
Insufficient Memory (handle leak)
Main Thread
Suspended Thread
Pleiades
Dual Stack Trace

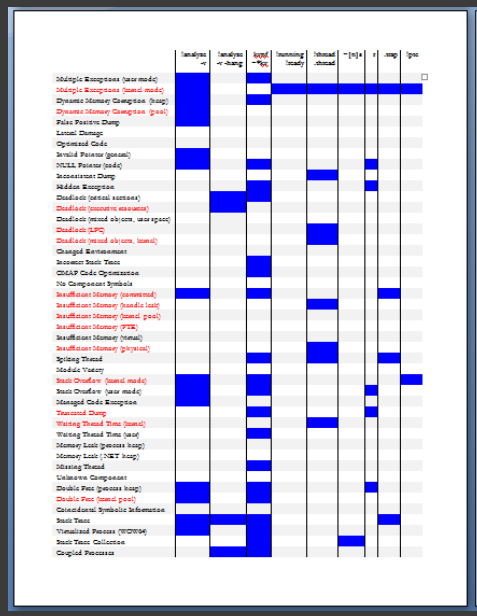
Case Studies

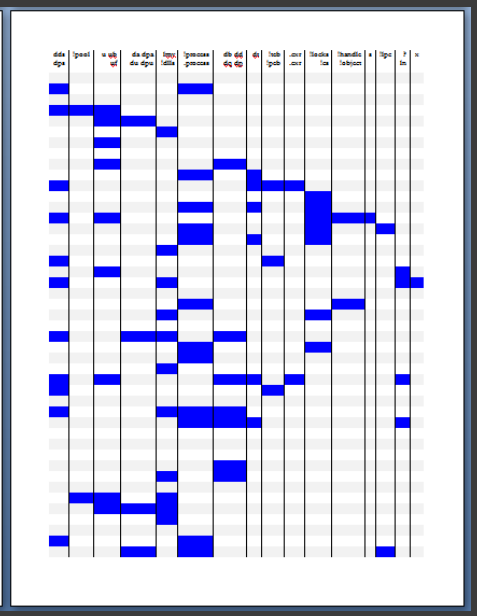
17 pattern interaction case studies using complete memory dumps:

http://www.dumpanalysis.org/blog/index.php/category/complete-memory-dump-analysis/

WinDbg Command Map



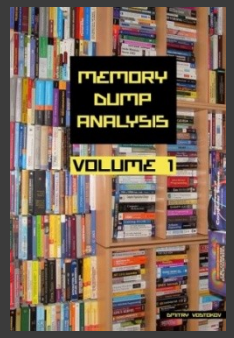


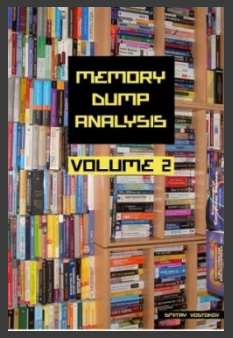


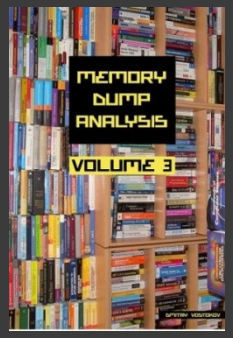
[Pattern <-> WinDbg command]

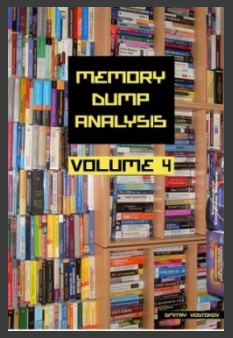
Resources

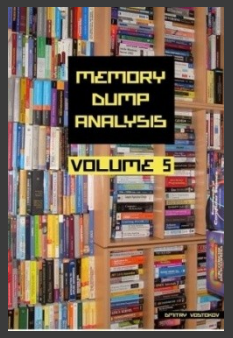
- WinDbg.org
- DumpAnalysis.org
- Memory Dump Analysis Anthology
- Accelerated Windows Memory Dump Analysis

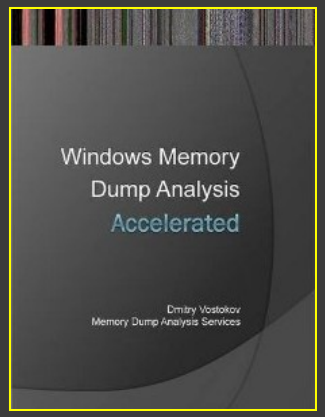


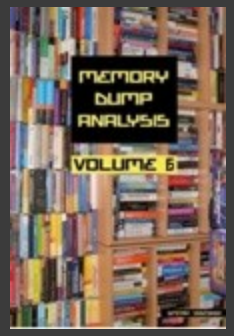








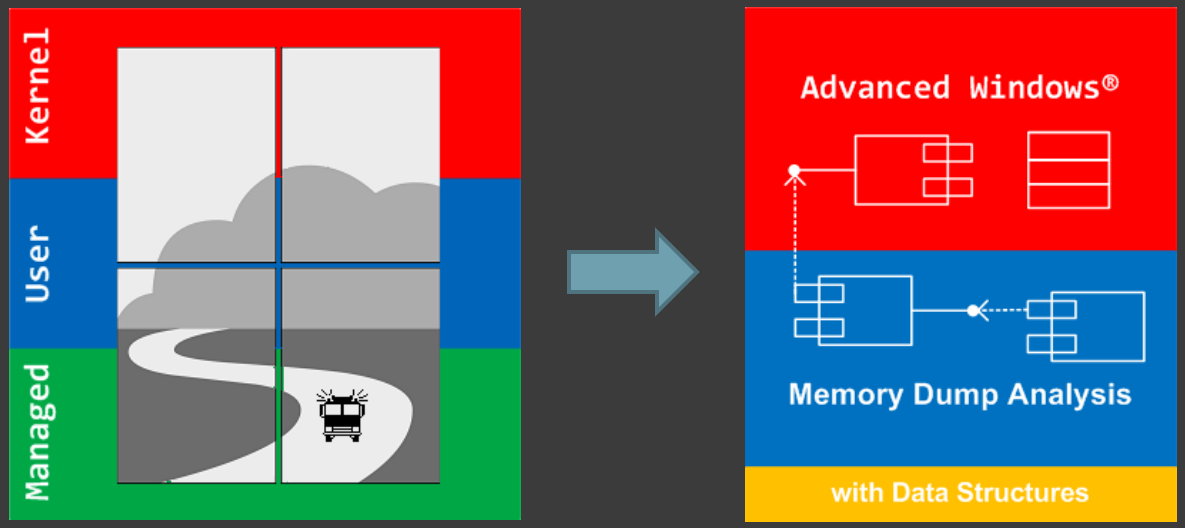




Forthcoming, end of 2011

Training

- Accelerated Windows Memory Dump Analysis
- Advanced Windows Memory Dump Analysis with Data Structures



20% discount if taking both or buying both course books

Q&A

Please send your feedback using the contact form on DumpAnalysis.com

Thank you for attendance!