Public Preview Version

Windows Memory Dump Analysis

Accelerated

Version 4.0

Dmitry Vostokov
Software Diagnostics Services

Prerequisites

Basic Windows troubleshooting

WinDbg Commands

We use these boxes to introduce WinDbg commands used in practice exercises

Training Goals

- Review fundamentals
- Learn how to analyze process dumps
- Learn how to analyze kernel dumps
- Learn how to analyze complete (physical) and active dumps

Training Principles

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

Coverage

- Windows Vista, 7, 8, 10
- Both x86 and x64 platforms
- Process, Kernel, Complete (Physical), and Active memory dumps, Minidumps
- Crashes, Hangs, Memory Leaks, CPU Spikes, Blue Screens (BSOD)

The main set of exercises is focused on Windows 10 x64 platform. All main exercises have their x86 equivalents from older Windows versions for additional practice.

Main Schedule Summary

Day 1

- Analysis Fundamentals (30 minutes)
- Process Memory Dumps (2 hours)

Day 2

- Process Memory Dumps (2 hours)

Windows 10 and 8.1 x64 memory dumps

Day 3

- Kernel Memory Dumps (2 hours)

Day 4

- Complete and Active Memory Dumps (2 hours)

Optional Schedule Summary

Day 1

- Legacy Process Memory Dumps (2 hours)

Day 2

- Legacy Process Memory Dumps (2 hours)

Day 3

- Legacy Kernel Memory Dumps (2 hours)

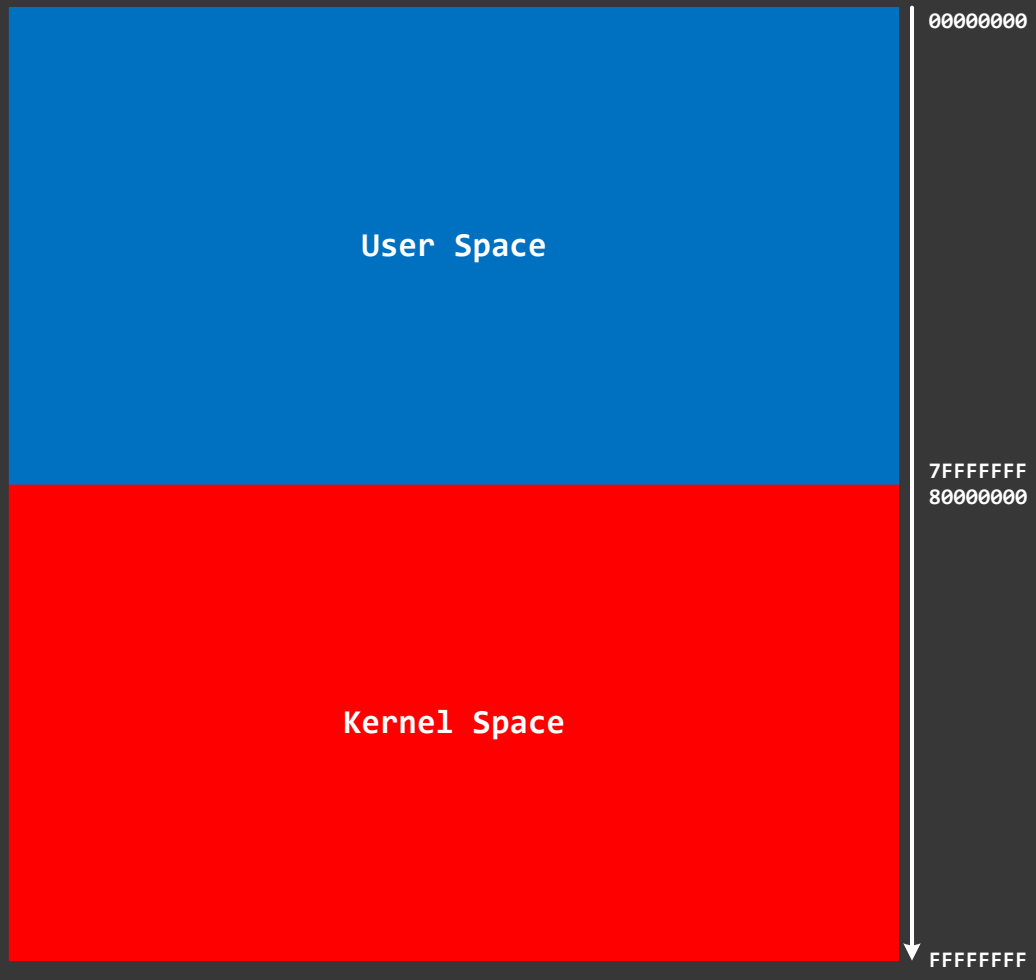
Day 4

- Legacy Complete Memory Dumps (2 hours)

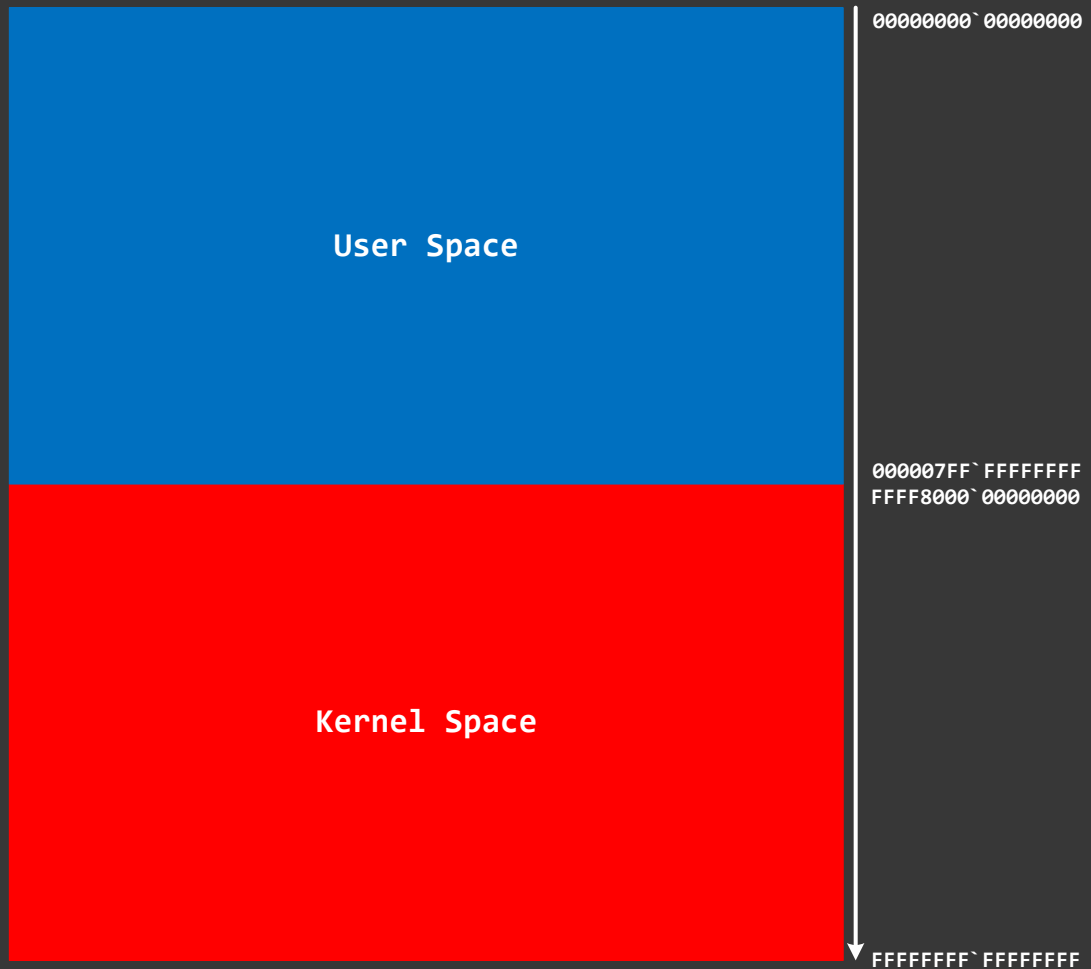
Windows Vista and 7 x86 memory dumps

Part 1: Fundamentals

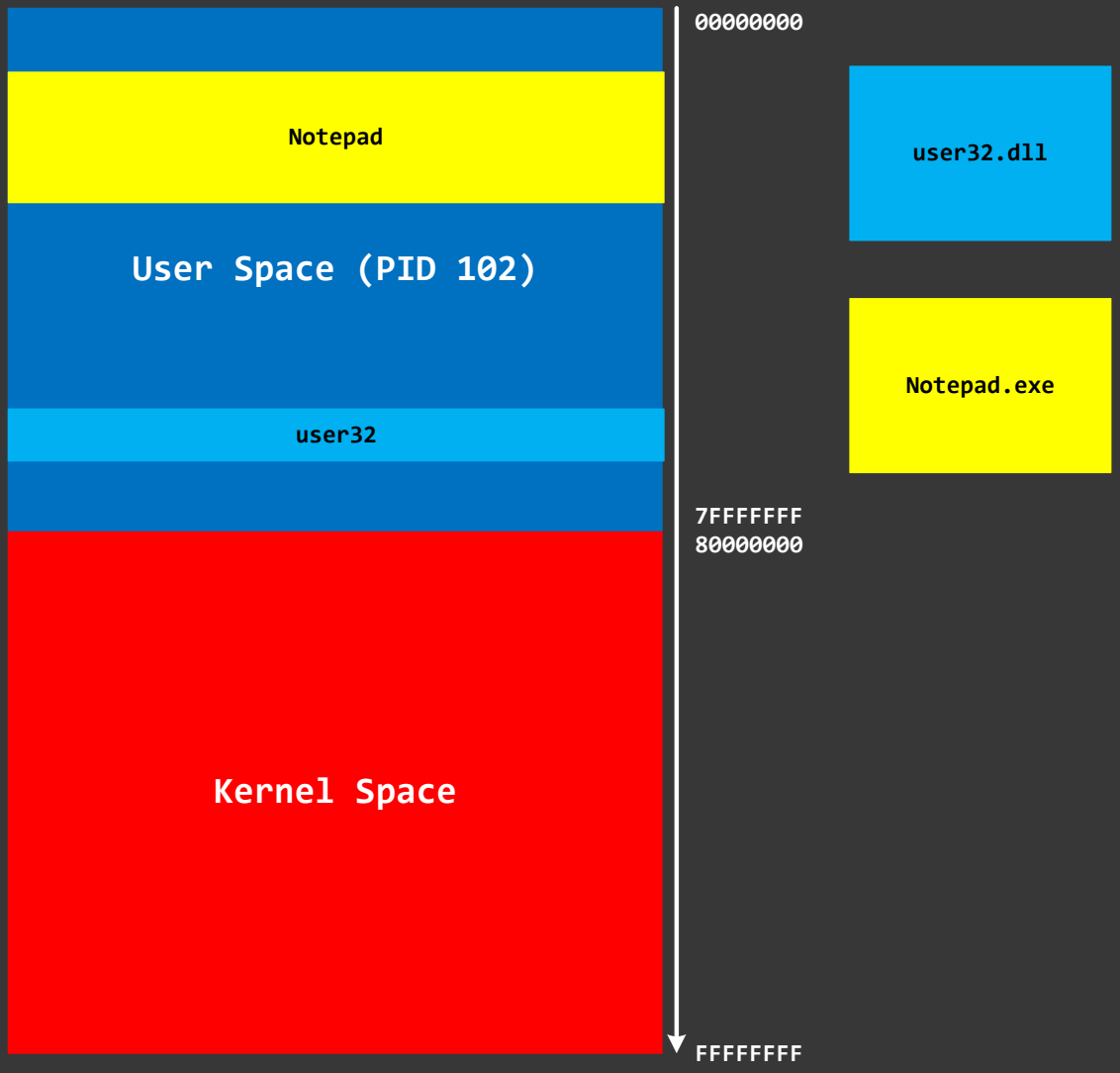
Process Space (x86)



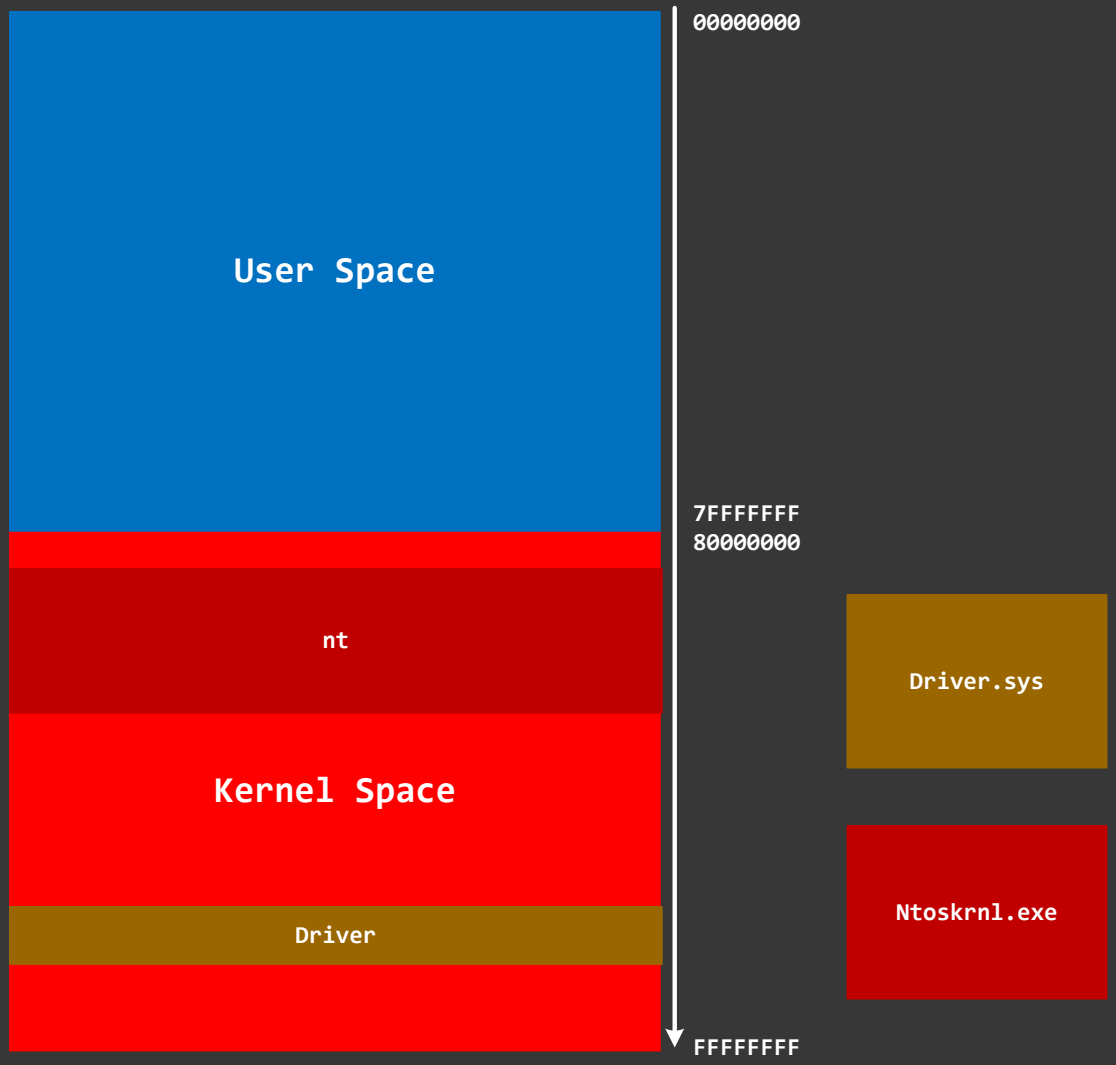
Process Space (x64)



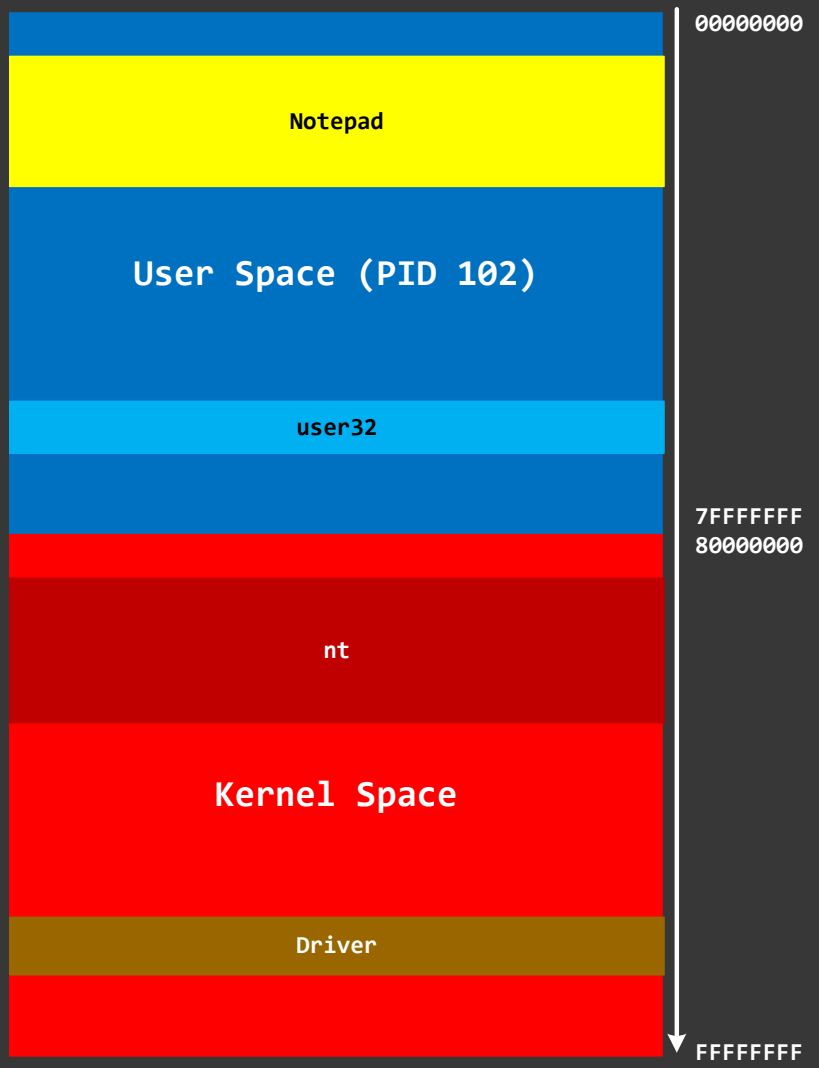
Application/Process/Module



OS Kernel/Driver/Module

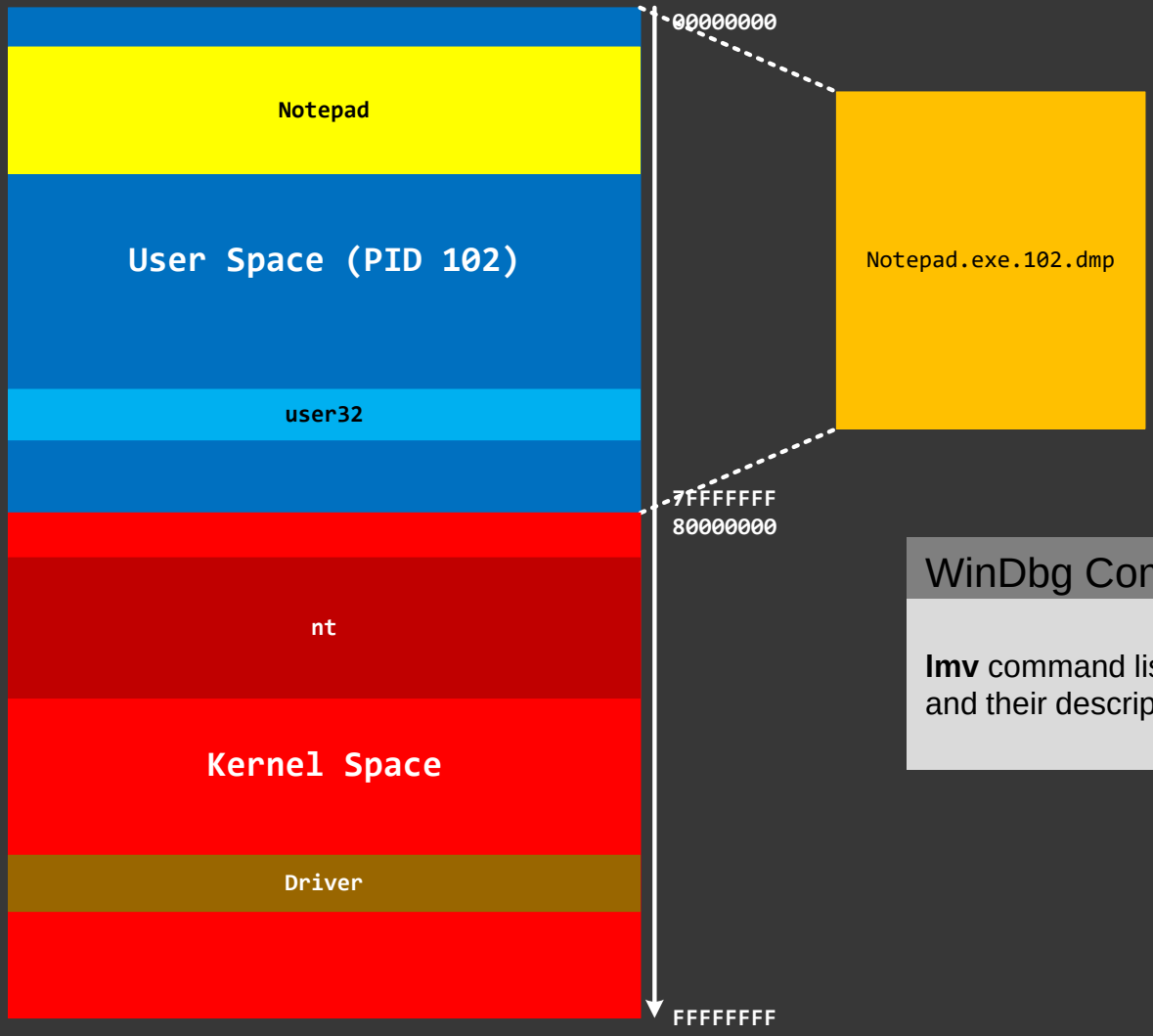


Process Virtual Space



00000000 ... FFFFFFFF

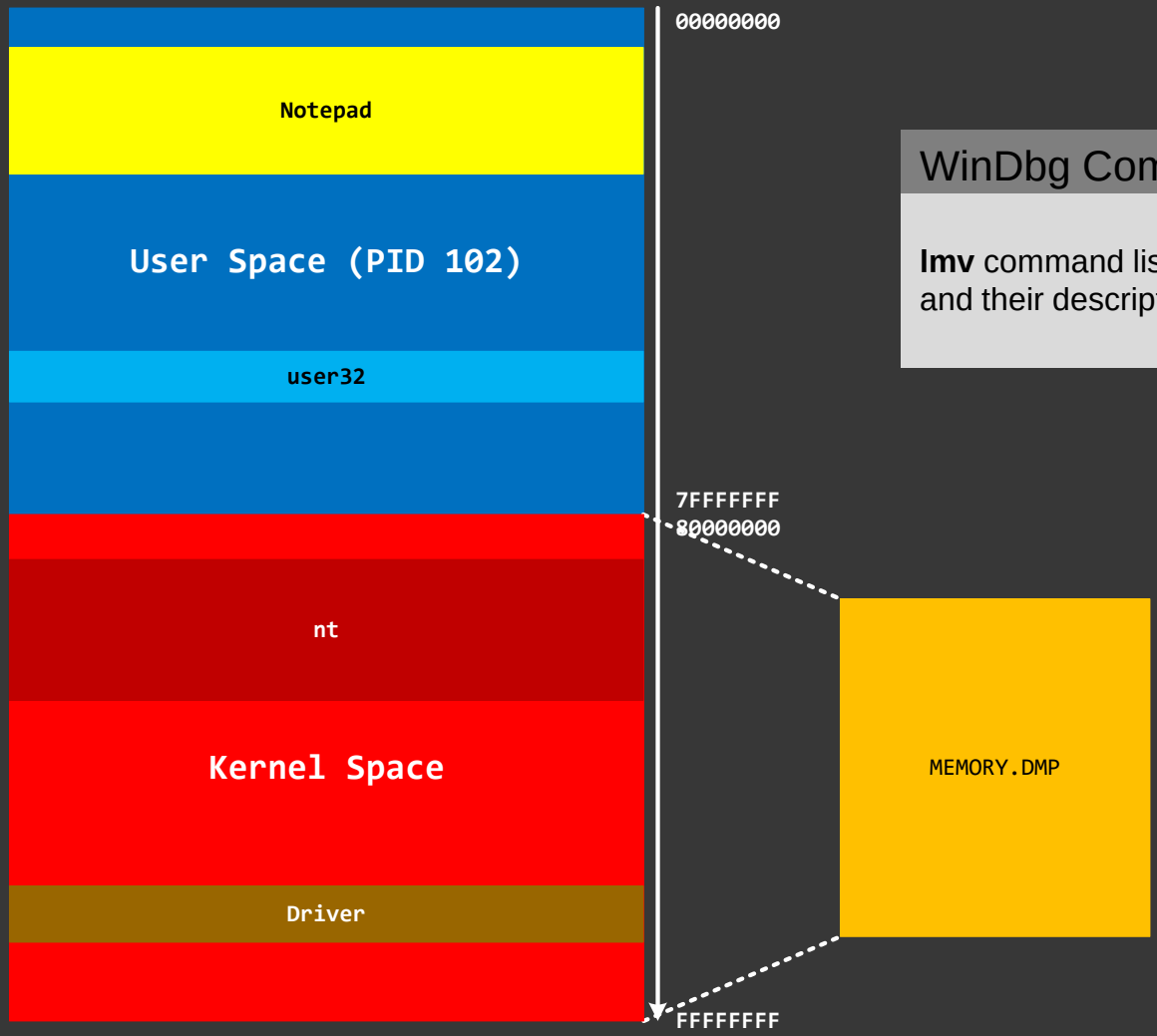
Process Memory Dump



WinDbg Commands

lmv command lists modules and their description

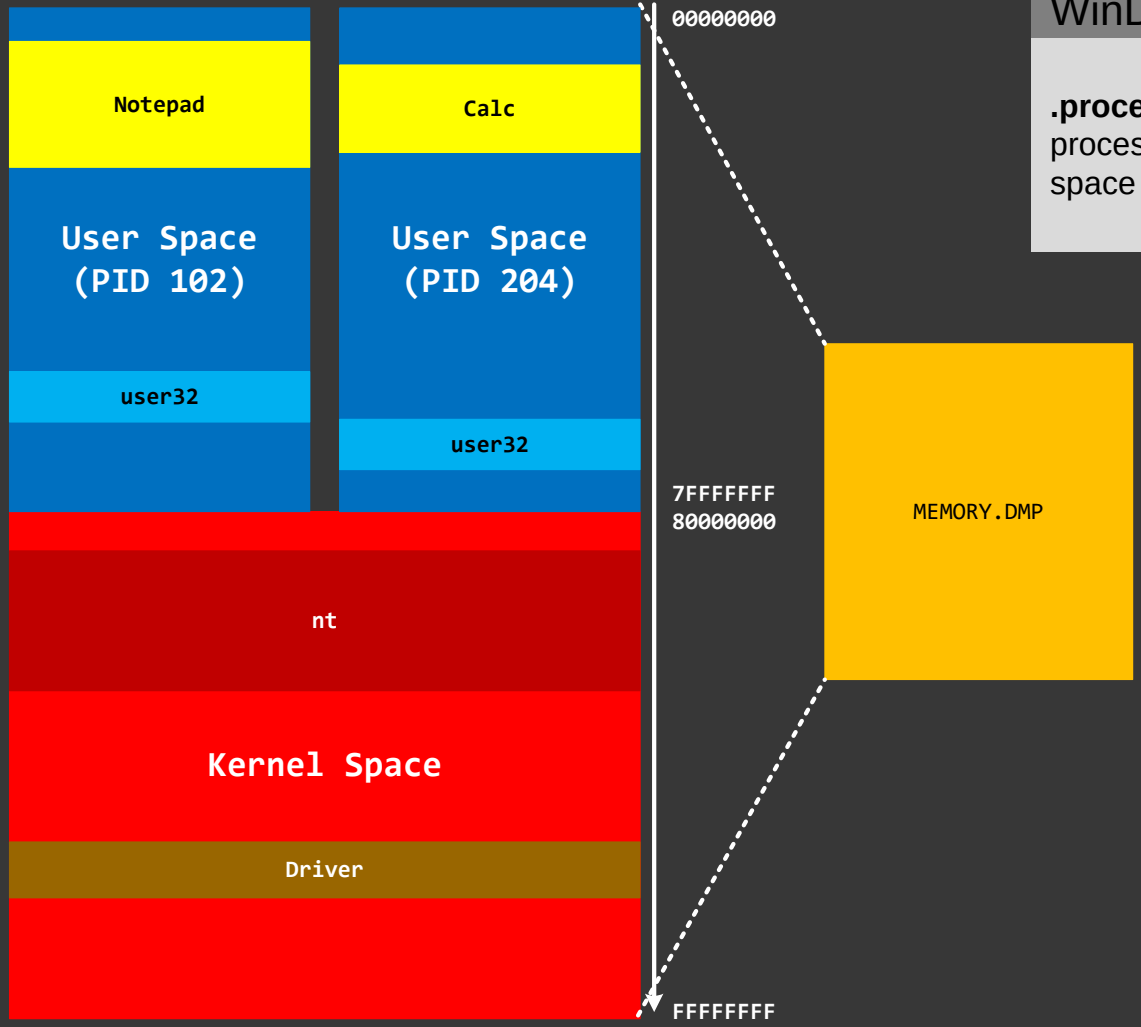
Kernel Memory Dump



WinDbg Commands

lmv command lists modules and their description

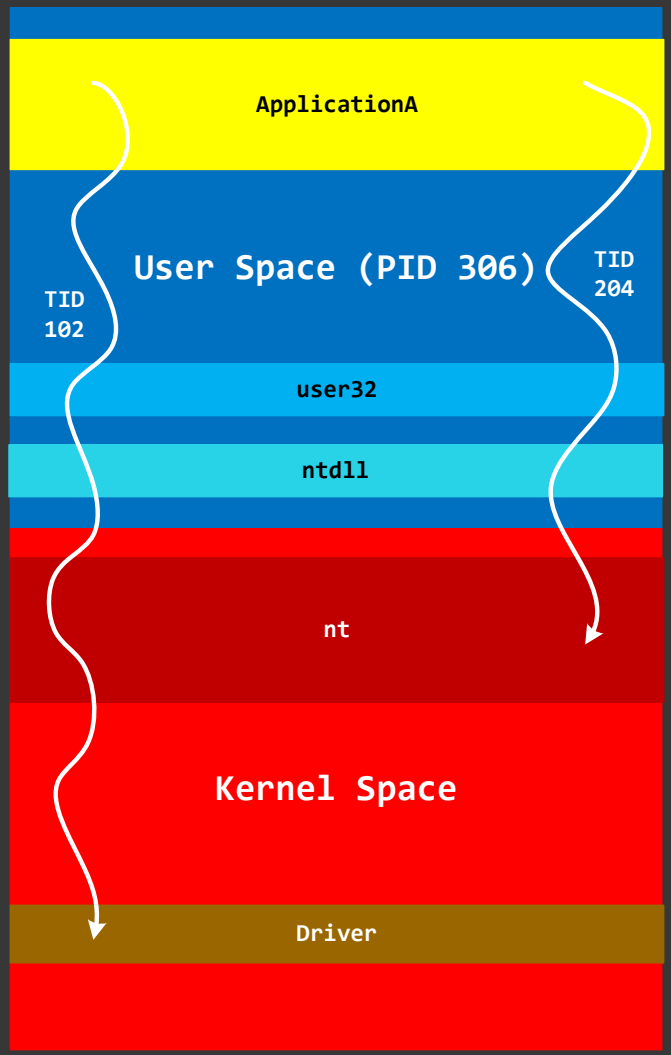
Complete Memory Dump



WinDbg Commands

.process switches between process virtual spaces (kernel space part remains the same)

Process Threads

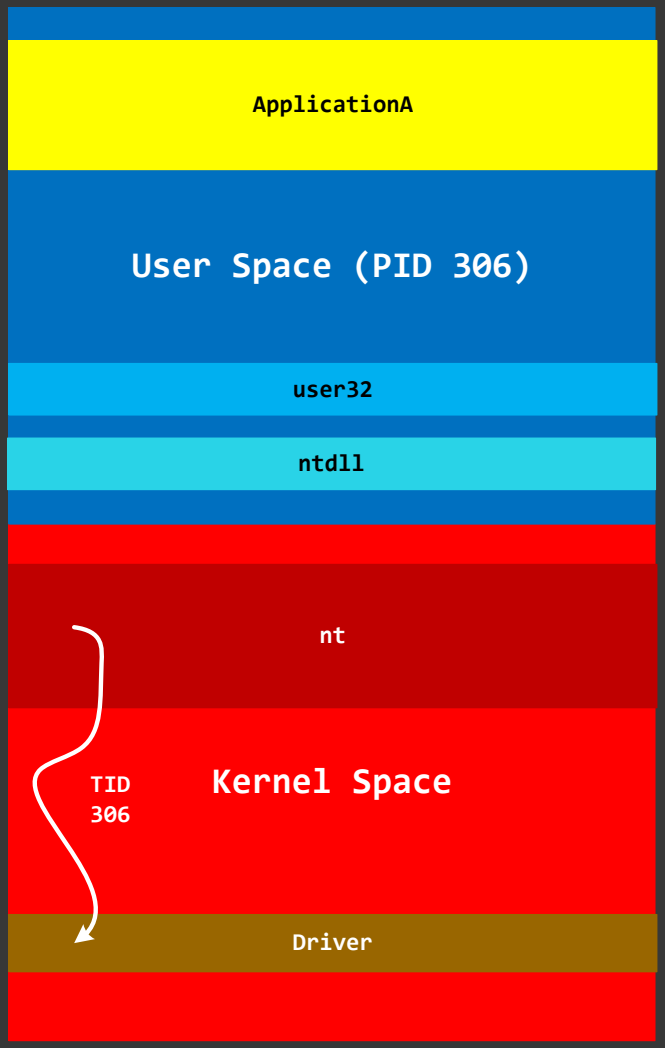


WinDbg Commands

Process dumps:
~<n>s switches between threads

Kernel/Complete dumps:
~<n>s switches between processors
.thread switches between threads

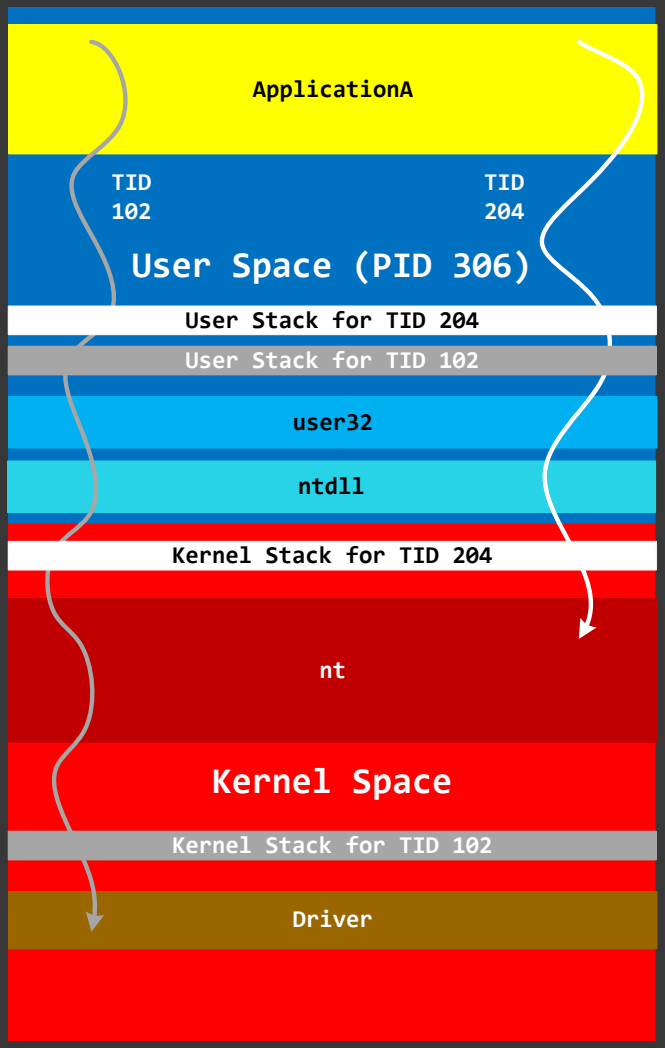
System Threads



WinDbg Commands

Kernel/Complete dumps:
~<n>s switches between processors
.thread switches between threads

Thread Stack Raw Data



WinDbg Commands

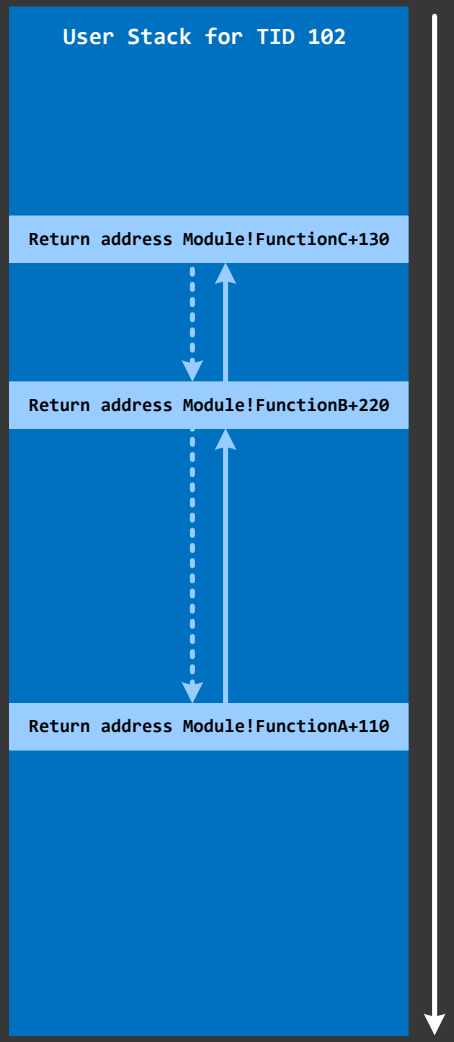
Process dumps:
!teb

Kernel dumps:
!thread

Complete dumps:
!teb for user space
!thread for kernel space

Data:
dc / **dps** / **dpp** / **dpa** / **dpu**

Thread Stack Trace




```
FunctionA()
{
  ...
  FunctionB();
  ...
}

FunctionB()
{
  ...
  FunctionC();
  ...
}

FunctionC()
{
  ...
  FunctionD();
  ...
}
```

Module!FunctionA

Resumes from address Module!FunctionA+110

Saves return address Module!FunctionA+110

Module!FunctionB

Resumes from address Module!FunctionB+220

Saves return address Module!FunctionB+220

Module!FunctionC

Resumes from address Module!FunctionC+130

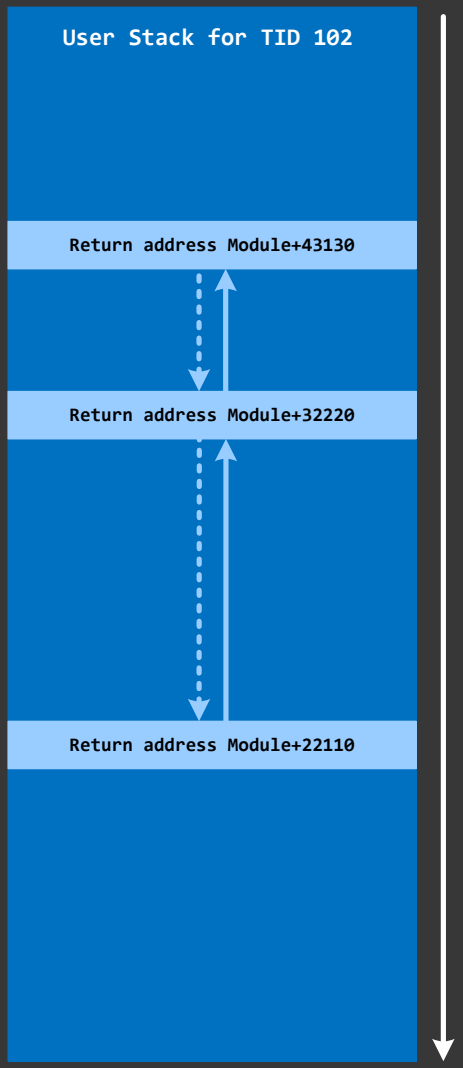
Saves return address Module!FunctionC+130

Module!FunctionD

WinDbg Commands

```
0:000> k
Module!FunctionD
Module!FunctionC+130
Module!FunctionB+220
Module!FunctionA+110
```

Thread Stack Trace (no PDB)

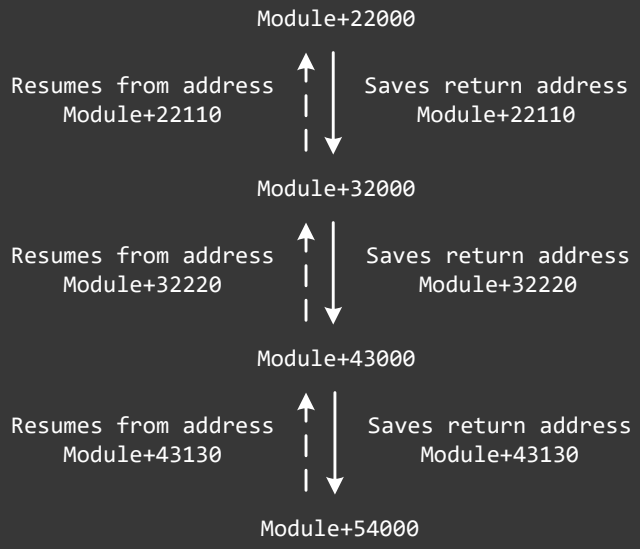



```
FunctionA()
{
  ...
  FunctionB();
  ...
}
FunctionB()
{
  ...
  FunctionC();
  ...
}
FunctionC()
{
  ...
  FunctionD();
  ...
}
```

Symbol file Module.pdb

FunctionA 22000 - 23000
FunctionB 32000 - 33000
FunctionC 43000 – 44000
FunctionD 54000 - 55000

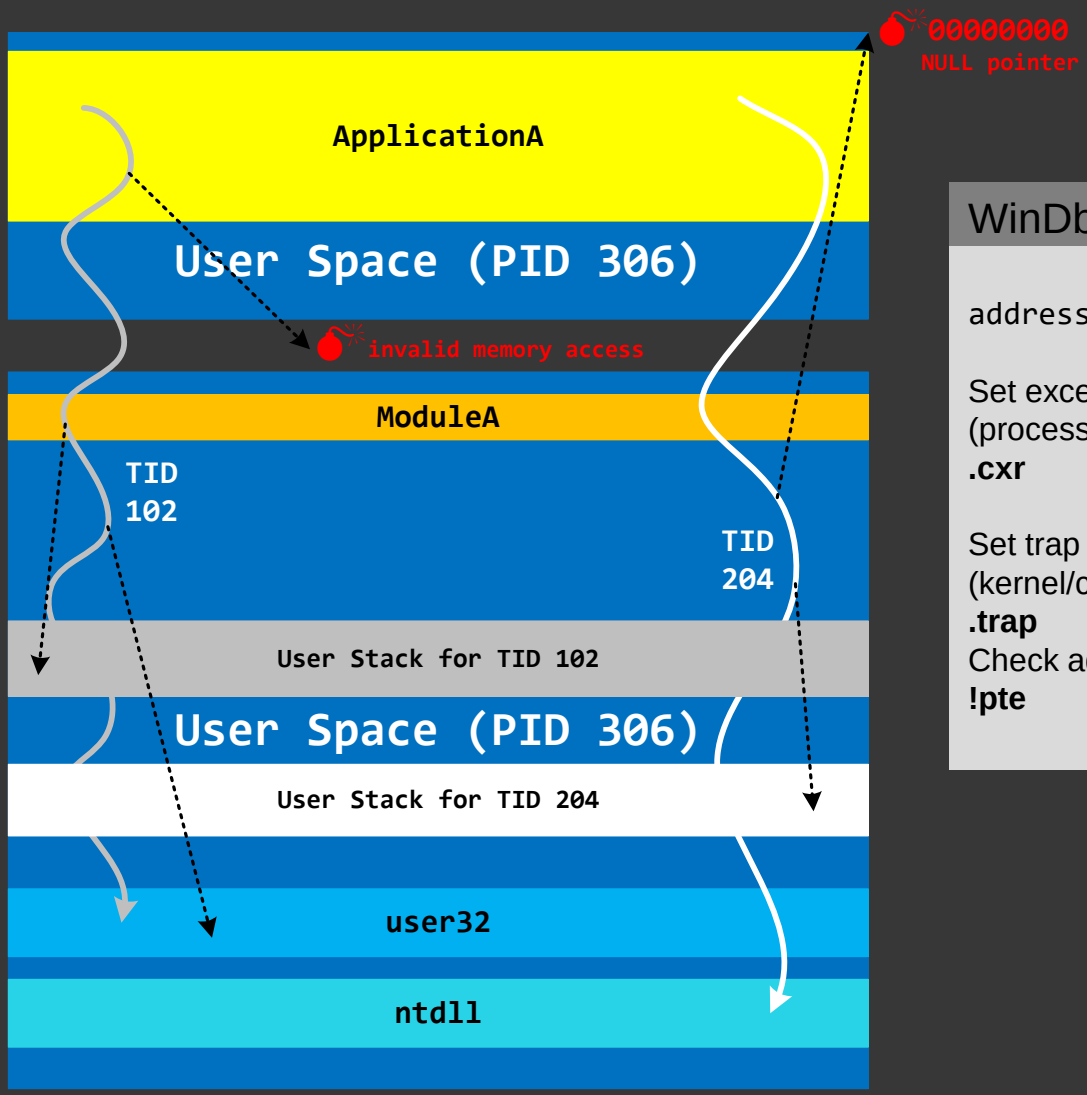
No symbols for Module



WinDbg Commands

```
0:000> k
Module+0
Module+43130
Module+32220
Module+22110
```

Exceptions (Access Violation)



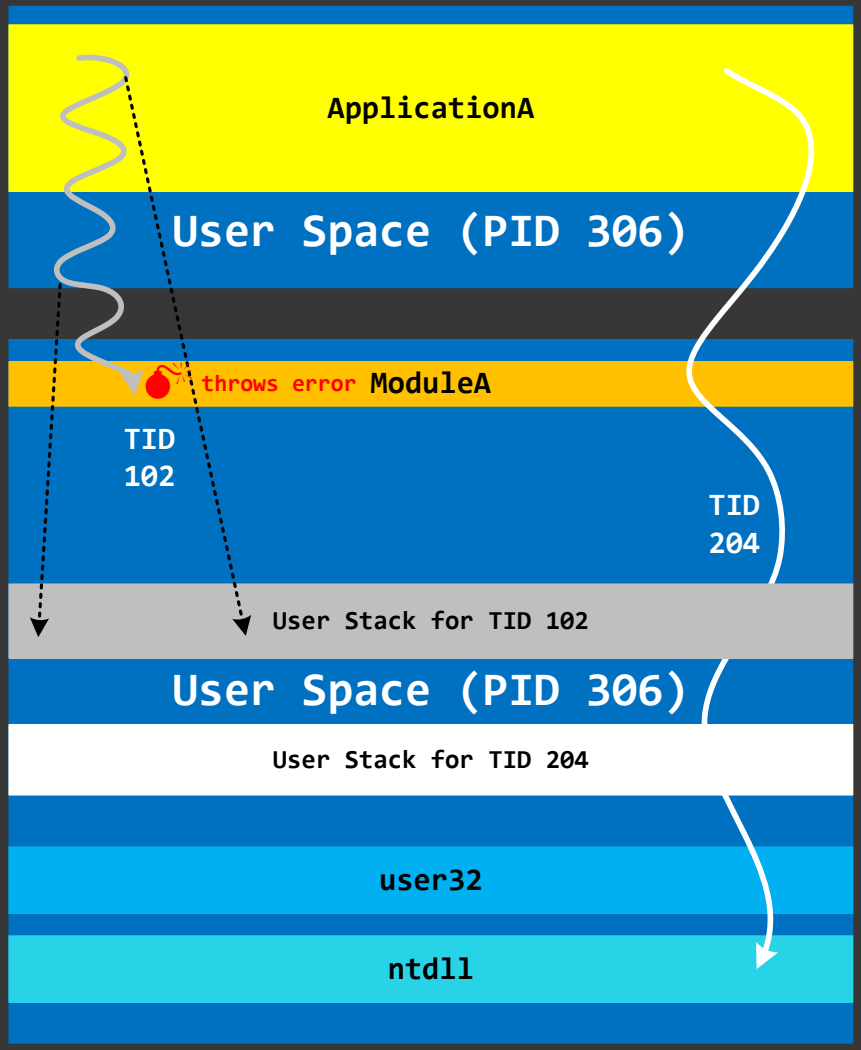
WinDbg Commands

```
address=????????
```

Set exception context (process dump):
.cxr

Set trap context (kernel/complete dump):
.trap
Check address:
!pte

Exceptions (Runtime)



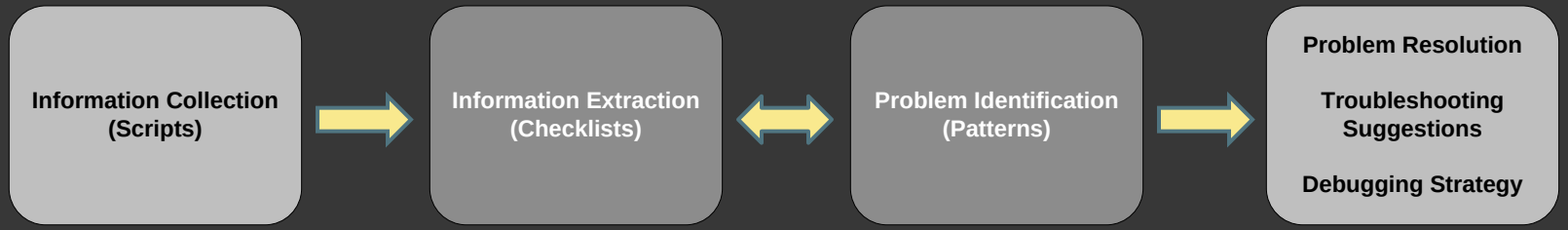
Pattern-Oriented Diagnostic Analysis

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

Diagnostic Problem: a set of indicators (symptoms, signs) describing a problem.

Diagnostic Analysis Pattern: a common recurrent analysis technique and method of diagnostic pattern identification in a specific context.

Diagnostics Pattern Language: common names of diagnostic and diagnostic analysis patterns. The same language for any operating system: Windows, Mac OS X, Linux, ...



Checklist: http://www.dumpanalysis.org/windows-memory-analysis-checklist

Patterns: http://www.dumpanalysis.org/blog/index.php/crash-dump-analysis-patterns/

Part 2: Practice Exercises

Links

- Memory Dumps:

NOT IN THE PUBLIC PREVIEW VERSION

- Exercise Transcripts:

NOT IN THE PUBLIC PREVIEW VERSION

Exercise 0

- **Goal:** Install Debugging Tools for Windows and learn how to set up symbols correctly
- **Patterns:** Incorrect Stack Trace
- \AWMDA-Dumps\Exercise-0-Download-Setup-WinDbg.pdf
- \AWMDA-Dumps\Exercise-Legacy.0-Download-Setup-WinDbg.pdf

Process Memory Dumps

Exercises P1 – P17

Exercise P1

- **Goal:** Learn how to see dump file type and version, get a stack trace, check its correctness, perform default analysis, list modules, check their version information, check process environment
- **Patterns:** Manual Dump; Stack Trace; Not My Version; Environment Hint
- \AWMDA-Dumps\Exercise-P1-Analysis-normal-process-dump-notepad-32.pdf
- \AWMDA-Dumps\Exercise-Legacy.P1-Analysis-normal-process-dump-notepad-32.pdf

Exercise P2

- **Goal:** Learn how to list stack traces, check their correctness, perform default analysis, list modules, check their version information, check process environment; dump module data
- **Patterns:** Manual Dump; Stack Trace; Not My Version; Environment Hint; Unknown Component
- \AWMDA-Dumps\Exercise-P2-Analysis-normal-process-dump-notepad-64.pdf
- \AWMDA-Dumps\Exercise-Legacy.P2-Analysis-normal-process-dump-notepad-64.pdf

Exercise P3

- **Goal:** Learn how to list stack traces, check their correctness, perform default analysis, list modules, check their version information, check thread age and CPU consumption
- **Patterns:** Stack Trace Collection
- \AWMDA-Dumps\Exercise-P3-Analysis-normal-process-dump-MicrosoftEdge-64.pdf
- \AWMDA-Dumps\Exercise-Legacy.P3-Analysis-normal-process-dump-iexplore-32.pdf

Exercise P4

- **Goal:** Learn to recognize exceptions in process memory dumps and get their context
- **Patterns:** Exception Thread; Multiple Exceptions; NULL Pointer
- \AWMDA-Dumps\Exercise-P4-Analysis-process-dump-ApplicationK-64-no-symbols.pdf
- \AWMDA-Dumps\Exercise-Legacy.P4-Analysis-process-dump-ApplicationK-32-no-symbols.pdf

Exercise P5

- **Goal:** Learn how to load application symbols, recognize exceptions in process memory dumps and get their context
- **Patterns:** Exception Thread; Multiple Exceptions; NULL Pointer
- \AWMDA-Dumps\Exercise-P5-Analysis-process-dump-ApplicationK-64-with-symbols.pdf
- \AWMDA-Dumps\Exercise-Legacy.P5-Analysis-process-dump-ApplicationK-32-with-symbols.pdf

Exercise P6

- **Goal:** Learn how to recognize heap corruption
- **Patterns:** Exception Thread; Dynamic Memory Corruption
- \AWMDA-Dumps\Exercise-P6-Analysis-process-dump-ApplicationL-32.pdf
- \AWMDA-Dumps\Exercise-Legacy.P6-Analysis-process-dump-ApplicationL-32.pdf

Exercise P7

- **Goal:** Learn how to recognize heap corruption and check error and status codes
- **Patterns:** Exception Thread; Dynamic Memory Corruption
- \AWMDA-Dumps\Exercise-P7-Analysis-process-dump-ApplicationL-64.pdf
- \AWMDA-Dumps\Exercise-Legacy.P7-Analysis-process-dump-ApplicationL-64.pdf

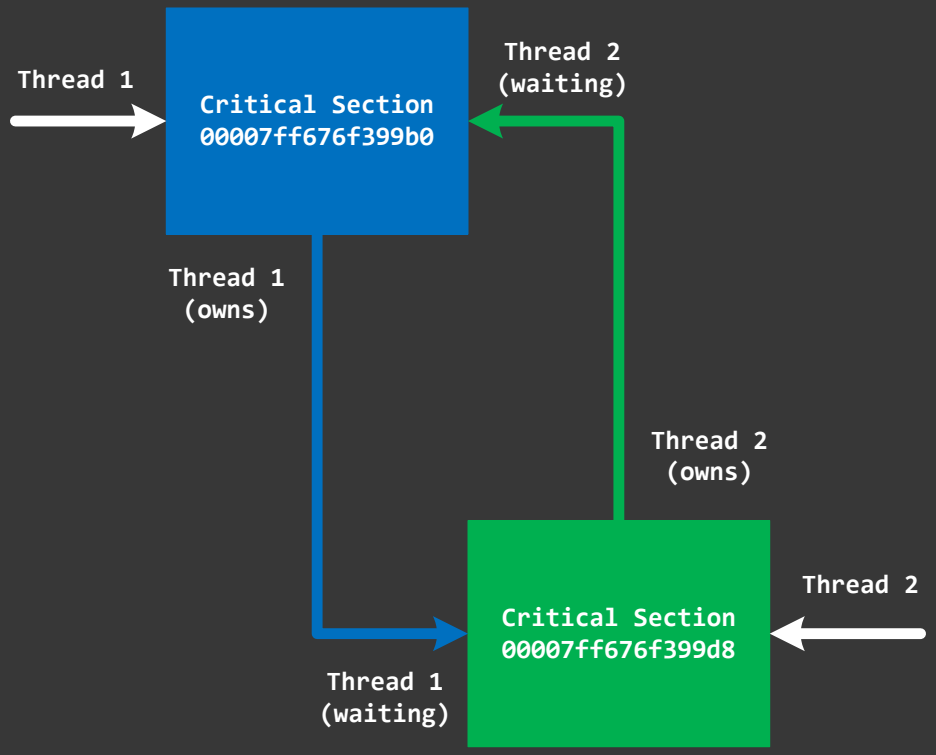
Exercise P8

- **Goal:** Learn how to recognize CPU spikes, invalid pointers and disassemble code
- **Patterns:** Exception Thread; Wild Code; CPU Spike; Multiple Exceptions; NULL Code Pointer; Invalid Pointer; Truncated Stack Trace; Stored Exception
- \AWMDA-Dumps\Exercise-P8-Analysis-process-dump-ApplicationM-64.pdf
- \AWMDA-Dumps\Exercise-Legacy.P8-Analysis-process-dump-ApplicationM-32.pdf

Exercise P9

- **Goal:** Learn how to recognize critical section waits and deadlocks, dump raw stack data and see hidden exceptions
- **Patterns:** Wait Chain; Deadlock; Hidden Exception
- \AWMDA-Dumps\Exercise-P9-Analysis-process-dump-ApplicationN-64.pdf
- \AWMDA-Dumps\Exercise-Legacy.P9-Analysis-process-dump-ApplicationN-64.pdf

Deadlock



Exercise P10

- **Goal:** Learn how to recognize application heap problems, buffer and stack overflow patterns and analyze raw stack data
- **Patterns:** Double Free; Local Buffer Overflow; Stack Overflow
- \AWMDA-Dumps\Exercise-P10-Analysis-process-dump-ApplicationO-64.pdf
- \AWMDA-Dumps\Exercise-Legacy.P10-Analysis-process-dump-ApplicationO-64.pdf

Exercise P11

- **Goal:** Learn how to analyze various patterns, raw stacks and execution residue
- **Patterns:** Divide by Zero; C++ Exception; Multiple Exceptions; Execution Residue
- \AWMDA-Dumps\Exercise-P11-Analysis-process-dump-ApplicationP-64.pdf
- \AWMDA-Dumps\Exercise-Legacy.P11-Analysis-process-dump-ApplicationP-32.pdf

Exercise P12

- **Goal:** Learn how to load the correct .NET WinDbg extension and analyze managed space
- **Patterns:** CLR Thread; Version-Specific Extension; Managed Code Exception; Managed Stack Trace
- \AWMDA-Dumps\Exercise-P12-Analysis-process-dump-ApplicationR-32.pdf

Exercise P13

- **Goal:** Learn how to analyze 32-process saved as a 64-bit process memory dump
- **Patterns:** Virtualized Process; Message Box; Execution Residue
- \AWMDA-Dumps\Exercise-P13-Analysis-process-dump-ApplicationA-64.pdf
- \AWMDA-Dumps\Exercise-Legacy.P13-Analysis-process-dump-ApplicationA-32.pdf

Exercise P14

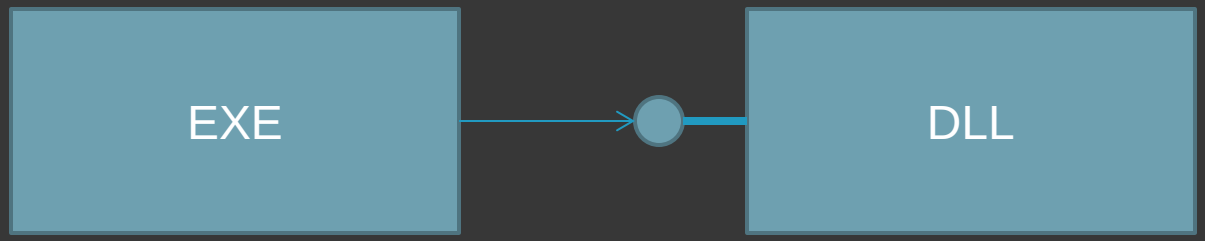
- **Goal:** Learn how to analyze process memory leaks
- **Patterns:** Spiking Thread; Thread Age; Memory Leak (process heap)
- \AWMDA-Dumps\Exercise-P14-Analysis-process-dump-ApplicationS-64.pdf
- \AWMDA-Dumps\Exercise-Legacy.P14-Analysis-process-dump-ApplicationS-32.pdf

Parameters and Locals

Debugging TV Frames episode 0x18

Symbol Types

- Exported and imported names



- Function and variable names

- Data types

Exercise P15

- **Goal:** Learn how to navigate function parameters in cases of reduced symbolic information in 32-bit process memory dumps
- **Patterns:** Reduced Symbolic Information
- \AWMDA-Dumps\Exercise-P15-Analysis-process-dump-notepad-32.pdf
- \AWMDA-Dumps\Exercise-Legacy.P15-Analysis-process-dump-notepad-32.pdf

Exercise P16

- **Goal:** Learn how to navigate function parameters in x64 process memory dumps
- **Patterns:** False Function Parameters, Injected Symbols
- \AWMDA-Dumps\Exercise-P16-Analysis-process-dump-notepad-64.pdf
- \AWMDA-Dumps\Exercise-Legacy.P16-Analysis-process-dump-notepad-64.pdf

Exercise P17

- **Goal:** Learn how to navigate object wait chains in 32-bit memory dumps saved with ProcDump
- **Patterns:** Wait Chain, Execution Residue, Deadlock
- \AWMDA-Dumps\Exercise-P17-Analysis-process-dump-ApplicationQ-32.pdf
- \AWMDA-Dumps\Exercise-Legacy.P17-Analysis-process-dump-ApplicationQ-32.pdf

Pattern Links

Spiking Thread
C++ Exception
Divide by Zero
Heap Corruption
Execution Residue
Invalid Pointer
Manual Dump
Managed Stack Trace
Not My Version
NULL Code Pointer
Stack Trace Collection
Environment Hint
Unknown Component
Virtualized Process
Version-Specific Extension
False Function Parameters
Reduced Symbolic Information
Stored Exception
CLR Thread
Critical Section Deadlock
Double Free
Exception Stack Trace
Hidden Exception
Local Buffer Overflow
Managed Code Exception
Multiple Exceptions
NULL Data Pointer
Stack Trace
Stack Overflow
Wild Code
Wait Chain
Message Box
Memory Leak
Injected Symbols
Truncated Stack Trace

Kernel Memory Dumps

Exercises K1 – K5

Exercise K1

- **Goal:** Learn how to get various information related to hardware, system, sessions, processes, threads and modules
- **Patterns:** NULL Pointer; False Effective Address; Invalid Pointer; Virtualized System; Stack Trace Collection
- \AWMDA-Dumps\Exercise-K1-Analysis-normal-kernel-dump-64.pdf
- \AWMDA-Dumps\Exercise-Legacy.K1-Analysis-normal-kernel-dump-32.pdf

Exercise K2

- **Goal:** Learn how to check and compare kernel pool usage
- **Patterns:** Manual Dump; Insufficient Memory (kernel pool)
- \AWMDA-Dumps\Exercise-K2-Analysis-kernel-dump-leak-64.pdf
- \AWMDA-Dumps\Exercise-Legacy.K2-Analysis-kernel-dump-leak-32.pdf

Exercise K3

- **Goal:** Learn how to recognize pool corruption and check pool data
- **Patterns:** Dynamic Memory Corruption (kernel pool); Regular Data; Execution Residue
- \AWMDA-Dumps\Exercise-K3-Analysis-kernel-dump-pool-corruption-64.pdf
- \AWMDA-Dumps\Exercise-Legacy.K3-Analysis-kernel-dump-pool-corruption-32.pdf

Exercise K4

- **Goal:** Learn how to check memory access violations, hooked or invalid code, and kernel raw stack
- **Patterns:** Invalid Pointer; Hooked Functions (kernel space); Execution Residue; Coincidental Symbolic Information; Past Stack Trace; Rough Stack Trace; Effect Component
- \AWMDA-Dumps\Exercise-K4-Analysis-kernel-dump-code-corruption-64.pdf
- \AWMDA-Dumps\Exercise-Legacy.K4-Analysis-kernel-dump-code-corruption-32.pdf

Exercise K5

- **Goal:** Learn how to check I/O requests
- **Patterns:** Blocking File; One-Thread Process
- \AWMDA-Dumps\Exercise-K5-Analysis-kernel-dump-hang-io-64.pdf
- \AWMDA-Dumps\Exercise-Legacy.K5-Analysis-kernel-dump-hang-io-32.pdf

Pattern Links

Manual Dump
Virtualized System
Insufficient Memory
Execution Residue
Hooked Functions
Blocking File
Past Stack Trace
Effect Component
One-Thread Process
Invalid Pointer
Stack Trace Collection
Dynamic Memory Corruption
Null Pointer
Coincidental Symbolic Information
Regular Data
Rough Stack Trace
False Effective Address

Additional Pattern Links

ERESOURCE patterns and case studies

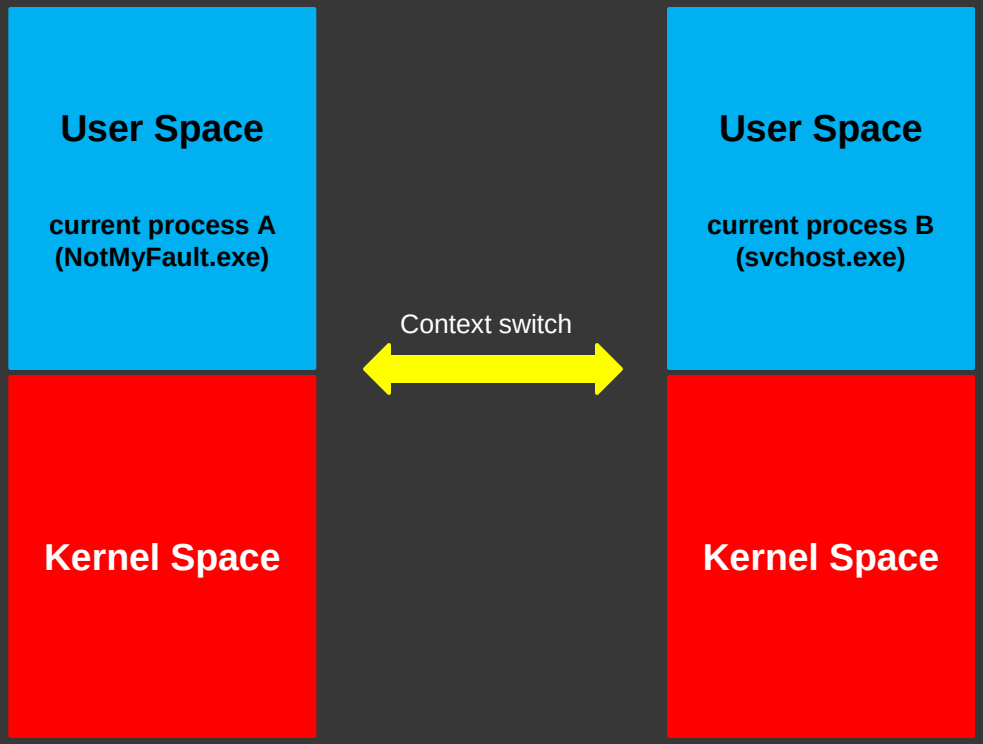
Wait Chain (Executive Resources) pattern is now reprinted in this course from Memory Dump Analysis Anthology, Volume 2, pages 147 – 150

Complete Memory Dumps

Exercises C1 – C4

Memory Spaces

- Complete memory == Physical memory
- We always see the current process space
- Kernel space is the same for any process



WinDbg Commands

switching to a different process context:

.process /r /p

Major Challenges

- Multiple processes (user spaces) to examine
- User space view needs to be correct when we examine another thread

WinDbg Commands
dump all stack traces: **!process 0 3f**

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 3f**
  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> 3f or !process <address> 1f**
  The same as the previous commands but only for an individual process

- **!thread <address> 1f**
  Shows thread information and stack trace

- **!thread <address> 16**
  The same as the previous command but shows the first 3 parameters for every function

Switch Commands

- **.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
...............................
```

Exercise C1

- **Goal:** Learn how to get various information related to processes, threads and modules
- **Patterns:** Stack Trace Collection
- \AWMDA-Dumps\Exercise-C1-Analysis-normal-complete-dump-64.pdf
- AWMDA-Dumps\Exercise-Legacy.C1-Analysis-normal-complete-dump-32.pdf

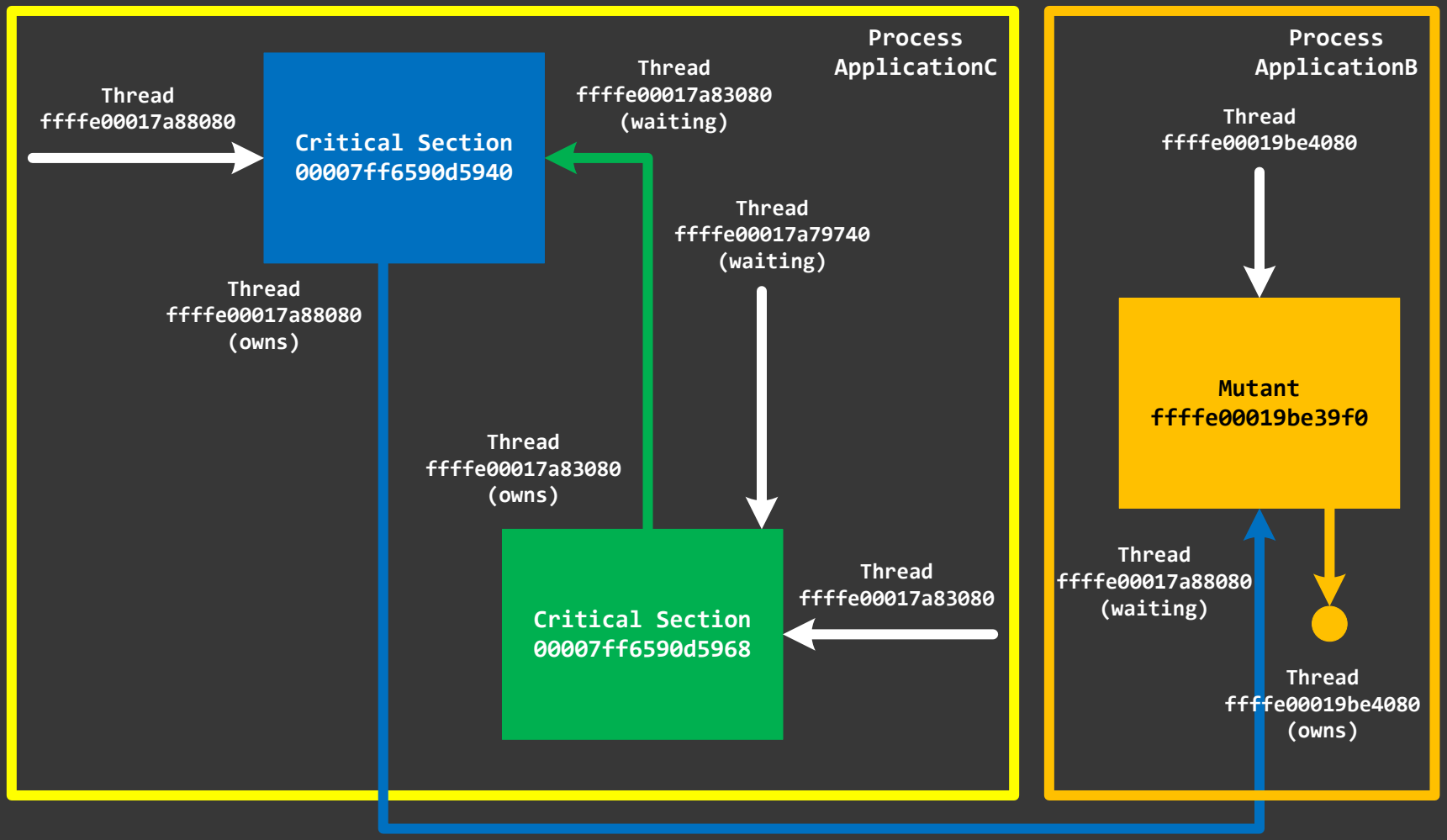
Exercise C2

- **Goal:** Learn how to recognize various abnormal software behavior patterns
- **Patterns:** Special Process; Handle Leak; Spiking Thread; Paged Out Data; Zombie Processes; Wait Chain; Dialog Box; Suspended Thread
- \AWMDA-Dumps\Exercise-C2-Analysis-problem-complete-dump-64.pdf
- \AWMDA-Dumps\Exercise-Legacy.C2-Analysis-problem-complete-dump-32.pdf

Exercise C3

- **Goal:** Learn how to recognize various abnormal software behavior patterns
- **Patterns:** Stack Trace Collection; Message Box; Wait Chain; Exception Thread
- \AWMDA-Dumps\Exercise-C3-Analysis-problem-complete-dump-64.pdf

Wait Chain



Exercise C4

- **Goal:** Learn how to recognize various abnormal software behavior patterns in x64 memory dumps
- **Patterns:** Virtualized Process; Message Box; Frozen Process; Wait Chain (ALPC)
- \AWMDA-Dumps\Exercise-C4-Analysis-problem-complete-dump-64.pdf

Active Memory Dump

Exercise A1

Exercise A1

- **Goal:** Get familiar with active memory dumps introduced in Windows 10
- **Patterns:** Stack Trace Collection; Execution Residue; Rough Stack Trace; Dual Stack Trace
- \AWMDA-Dumps\Exercise-A1-Analysis-problem-active-dump-64.pdf

Pattern Links

Special Process
Spiking Thread
Message Box
Exception Stack Trace
Frozen Process
Zombie Processes
Dialog Box
Execution Residue
Dual Stack Trace
Handle Leak
Stack Trace Collection
Wait Chain (critical sections)
Virtualized Process
Wait Chain (LPC/ALPC)
Paged Out Data
Suspended Thread
Rough Stack Trace

Also another pattern is present in Legacy.C2 memory dump (not shown in the exercise transcript):

Wait Chain (window messaging)

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 listing both x86 and x64 stack traces

Kernel Minidumps

Memory Dump Analysis Anthology, Volume 1
pages 43 – 67

Now reprinted in this course

Pattern Classification

Space/Mode
Hooksware
DLL Link Patterns
Contention Patterns
Stack Trace Patterns
Exception Patterns
Module Patterns
Thread Patterns
Dynamic Memory Corruption Patterns
.NET / CLR / Managed Space Patterns
Falsity and Coincidence Patterns
Memory dump type
Wait Chain Patterns
Insufficient Memory Patterns
Stack Overflow Patterns
Symbol Patterns
Meta-Memory Dump Patterns
Optimization Patterns
Process Patterns
Deadlock and Livelock Patterns
Executive Resource Patterns
RPC, LPC and ALPC Patterns

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

Additional Resources

- WinDbg Help / WinDbg.org (quick links)
- DumpAnalysis.org / PatternDiagnostics.org
- Debugging.TV / YouTube.com/DebuggingTV
- Windows Internals, 6th ed.
- Practical Foundations of Windows Debugging, Disassembling, Reversing
- Advanced Windows Debugging
- Inside Windows Debugging
- Windows Debugging Notebook: Essential User Space WinDbg Commands
- Memory Dump Analysis Anthology

Further Training Courses

- Practical Foundations of Windows Debugging, Disassembling, Reversing
- Advanced Windows Memory Dump Analysis with Data Structures, 2nd edition
- Accelerated .NET Memory Dump Analysis, 2nd edition
- Accelerated Windows Malware Analysis with Memory Dumps
- Accelerated Disassembly, Reconstruction and Reversing
- Accelerated Windows Debugging[3]

Q&A

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

Thank you for attendance!