



Windows Memory Dump Analysis **Accelerated**

Version 7

Part 1: Process User Space

Dmitry Vostokov
Software Diagnostics Services

Accelerated Windows Memory Dump Analysis, Seventh Edition, Part 1: Process User Space

Published by OpenTask, Republic of Ireland

Copyright © 2026 by OpenTask

Copyright © 2026 by Software Diagnostics Services

Copyright © 2026 by Dmitry Vostokov

Copyright © 2026 by Dublin School of Security

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, transmitted in any form or by any means, or used for training artificial intelligence systems without the prior written permission of the publisher.

Product and company names mentioned in this book may be trademarks of their owners.

OpenTask books and magazines are available through booksellers and distributors worldwide. For further information or comments, send requests to press@opentask.com.

A CIP catalog record for this book is available from the British Library.

ISBN-13: 978-1-912636-18-1 (Paperback)

Revision 7.03 (February 2026)

Contents

About the Author	7
Introduction	8
What's New.....	9
Prerequisites.....	10
Training Goals.....	11
Training Principles	12
Coverage (Part 1)	13
Fundamentals.....	14
Process Space (x64, ARM64).....	15
Application/Process/Module	16
Process Virtual Space (x64, ARM64).....	17
Process Memory Dump (x64, ARM64).....	18
Process Space (x86)	19
Application/Process/Module (x86)	20
Process Memory Dump (x86)	21
Process Virtual Space (WOW64).....	22
Process Memory Dump (WOW64).....	23
Process Threads.....	24
Thread Stack Raw Data	25
Thread Stack Trace.....	26
Thread Stack Trace (no PDB).....	27
Exceptions (Access Violation)	28
Exception (Runtime)	29
Pattern-Oriented Diagnostic Analysis.....	30
Review of x64 Disassembly.....	31
x64 CPU Registers.....	32
x64 Instructions and Registers	33
x64 Stack Addressing.....	34
x64 Memory Cell Sizes.....	35
x64 Memory Load Instructions.....	36
x64 Memory Store Instructions.....	37
x64 Flow Instructions.....	38

x64 Windows API Parameters.....	39
Review of ARM64 Disassembly.....	40
A64 CPU Registers	41
A64 Instructions and Registers.....	42
A64 Stack Addressing	43
A64 Memory Load Instructions	44
A64 Memory Store Instructions	45
A64 Flow Instructions	46
A64 Windows API Parameters	47
Practical Exercises	48
Links.....	49
Exercise 0	50
Process Memory Dumps	57
Types of WinDbg Command	58
Exercise P1.....	59
Exercise P2.....	71
Exercise P3.....	73
Exercise P4.....	85
Exercise P5.....	95
Exercise P6.....	101
Exercise P7.....	112
Exercise P8.....	127
Mechanisms (Invalid Pointer)	141
Mechanisms (Active Thread)	142
Exercise P9.....	143
Deadlock.....	154
Mechanisms (Deadlock)	155
Exercise P10.....	156
Mechanisms (Heap Corruption)	166
Mechanisms (Stack Corruption).....	167
Mechanisms (Stack Overflow).....	168
Exercise P11.....	169
Exercise P12.....	181
Exercise P13.....	198

Exercise P14.....	218
Mechanisms (Memory Leak).....	236
Parameters and Locals	237
Symbol Types	238
Exercise P15.....	239
Exercise P16.....	244
Exercise P17.....	252
Exercise P18.....	264
Exercise P19.....	275
Exercise P20.....	290
Exercise P21.....	293
Exercise P22.....	297
Exercise P23.....	308
Exercise P24.....	316
Exercise P25.....	324
Windows Internals	335
Collection Methods.....	337
Process Dump Generation	338
Pattern Links.....	339
Pattern Classification.....	340
Pattern Case Studies	341
Additional Resources	342
Further Training Courses.....	344
Application Source Code.....	346
AppA.....	346
AppK.....	348
AppL.....	349
AppL2	350
AppM.....	351
AppN	352
AppO.....	354
AppP.....	356
AppR2.....	358
AppS	359

AppQ.....	361
AppT.....	366
ServiceA.....	368
Rusty	371
Selected Q&A.....	372
Added in Version 5.5	372
From Earlier Versions.....	383
Additional Materials	409
Triple Dereference.....	409
Large Heap Allocations.....	412

Exercise P1

Exercise P1

- **Goal:** Learn how to see dump file type and version, get a stack trace, check its correctness, perform default analysis, list threads and modules, check module version information, dump module data, and check the process environment
- **Patterns:** Manual Dump (Process); Stack Trace; Not My Version (Software); Environment Hint; Unknown Component; Historical Information
- [\AWMDA-Dumps\Exercise-P1-Analysis-normal-process-dump-wordpad-x64.pdf](#)

Practical Exercises

Goal: Learn how to see dump file type and version, get a stack trace, check its correctness, perform default analysis, list threads and modules, check module version information, dump module data, and check the process environment.

Patterns: Manual Dump (Process); Stack Trace; Not My Version (Software); Environment Hint; Unknown Component; Historical Information.

1. Launch WinDbg.
2. Open \AWMDA-Dumps\Process\x64\wordpad.DMP.
3. We get the dump file loaded:

```
Microsoft (R) Windows Debugger Version 10.0.29482.1003 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.
```

```
Loading Dump File [C:\AWMDA-Dumps\Process\x64\wordpad.DMP]
User Mini Dump File with Full Memory: Only application data is available
```

```
***** Path validation summary *****
Response                               Time (ms)      Location
Deferred                               srv*
Symbol search path is: srv*
Executable search path is:
Windows 10 Version 22000 MP (2 procs) Free x64
Product: WinNt, suite: SingleUserTS Personal
Edition build lab: 22000.1.amd64fre.co_release.210604-1628
Debug session time: Sat Jul 15 18:04:32.000 2023 (UTC + 1:00)
System Uptime: 0 days 0:21:14.009
Process Uptime: 0 days 0:04:51.000
.....
.....
Loading unloaded module list
.....
For analysis of this file, run !analyze -v
win32u!NtUserGetMessage+0x14:
00007ff9`13c31414 c3          ret
```

4. Open a log file to save all future output using the **.logopen** command:

```
0:000> .logopen C:\AWMDA-Dumps\Process\x64\wordpad.log
Opened log file 'C:\AWMDA-Dumps\Process\x64\wordpad.log'
```

5. Type the **k** command to verify the correctness of the stack trace:

```
0:000> k
# Child-SP          RetAddr           Call Site
00 00000037`1567fd48 00007ff9`15a6464e win32u!NtUserGetMessage+0x14
01 00000037`1567fd50 00007ff8`d3150813 user32!GetMessageW+0x2e
02 00000037`1567fdb0 00007ff8`d3150736 mfc42u!CWinThread::PumpMessage+0x23
03 00000037`1567fde0 00007ff8`d314f2bc mfc42u!CWinThread::Run+0x96
```

```

04 00000037`1567fe20 00007ff6`d9eebcfd mfc42u!AfxWinMain+0xbc
05 00000037`1567fe60 00007ff9`14a754e0 wordpad!__wmainCRTStartup+0x1dd
06 00000037`1567ff20 00007ff9`1668485b kernel32!BaseThreadInitThunk+0x10
07 00000037`1567ff50 00000000`00000000 ntdll!RtlUserThreadStart+0x2b

```

6. Type the **version** command to get the OS version, system, and process uptimes, the dump file timestamp, and its type:

```

0:000> version
Windows 10 Version 22000 MP (2 procs) Free x64
Product: WinNT, suite: SingleUserTS Personal
Edition build lab: 22000.1.amd64fre.co_release.210604-1628
Debug session time: Sat Jul 15 18:04:32.000 2023 (UTC + 1:00)
System Uptime: 0 days 0:21:14.009
Process Uptime: 0 days 0:04:51.000
  Kernel time: 0 days 0:00:01.000
  User time: 0 days 0:00:00.000
Full memory user mini dump: C:\AWMDA-Dumps\Process\x64\wordpad.DMP

Microsoft (R) Windows Debugger Version 10.0.25877.1004 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.

command line: '"C:\Program Files\WindowsApps\Microsoft.Windows.Common-UI_1.2306.12001.0_x-ww_8wekyb3d8bbwe\amd64\EngHost.exe"
npip:pipe=DbgX_485dd9d6637e46dffb0270a9e44d88e,password=0ed044e5d5a4 "C:\Program
Files\WindowsApps\Microsoft.Windows.Common-UI_1.2306.12001.0_x-ww_8wekyb3d8bbwe\amd64" "C:\ProgramData\Dbg" ' Debugger Process 0x21DC
dbgeng: image 10.0.25877.1004,
  [path: C:\Program Files\WindowsApps\Microsoft.Windows.Common-UI_1.2306.12001.0_x-ww_8wekyb3d8bbwe\amd64\dbgeng.dll]
dbghelp: image 10.0.25877.1004,
  [path: C:\Program Files\WindowsApps\Microsoft.Windows.Common-UI_1.2306.12001.0_x-ww_8wekyb3d8bbwe\amd64\dbghelp.dll]
DIA version: 32595
Extension DLL search Path:
[...]
Extension DLL chain:
  DbgEngCoreDMEExt: image 10.0.25877.1004, API 0.0.0,
    [path: C:\Program Files\WindowsApps\Microsoft.Windows.Common-UI_1.2306.12001.0_x-ww_8wekyb3d8bbwe\amd64\winext\DbgEngCoreDMEExt.dll]
  MachOBinComposition: image 10.0.25877.1004, API 0.0.0,
    [path: C:\Program Files\WindowsApps\Microsoft.Windows.Common-UI_1.2306.12001.0_x-ww_8wekyb3d8bbwe\amd64\winext\MachOBinComposition.dll]
  ELFBinComposition: image 10.0.25877.1004, API 0.0.0,
    [path: C:\Program Files\WindowsApps\Microsoft.Windows.Common-UI_1.2306.12001.0_x-ww_8wekyb3d8bbwe\amd64\winext\ELFBinComposition.dll]
  dbghelp: image 10.0.25877.1004, API 10.0.6,
    [path: C:\Program Files\WindowsApps\Microsoft.Windows.Common-UI_1.2306.12001.0_x-ww_8wekyb3d8bbwe\amd64\dbghelp.dll]
  exts: image 10.0.25877.1004, API 1.0.0,
    [path: C:\Program Files\WindowsApps\Microsoft.Windows.Common-UI_1.2306.12001.0_x-ww_8wekyb3d8bbwe\amd64\WINXP\exts.dll]
  uext: image 10.0.25877.1004, API 1.0.0,
    [path: C:\Program Files\WindowsApps\Microsoft.Windows.Common-UI_1.2306.12001.0_x-ww_8wekyb3d8bbwe\amd64\winext\uext.dll]
  ntsdexts: image 10.0.25877.1004, API 1.0.0,
    [path: C:\Program Files\WindowsApps\Microsoft.Windows.Common-UI_1.2306.12001.0_x-ww_8wekyb3d8bbwe\amd64\WINXP\ntsdexts.dll]

```

Note: Debug session time is when the dump was generated. Although the dump is called a “mini dump,” it is a full memory user dump that includes all process memory.

7. Type the default analysis command **!analyze -v**:

Note: This command may take some time initially as symbols may be downloaded from the symbol server:

```

0:000> !analyze -v
*****
*                                     *
*               Exception Analysis   *
*                                     *
*****

KEY_VALUES_STRING: 1

  Key : Analysis.CPU.mSec
  Value: 702

  Key : Analysis.Elapsed.mSec
  Value: 8353

  Key : Analysis.IO.Other.Mb
  Value: 13

```

Practical Exercises

```
Key : Analysis.IO.Read.Mb
Value: 0

Key : Analysis.IO.Write.Mb
Value: 27

Key : Analysis.Init.CPU.mSec
Value: 171

Key : Analysis.Init.Elapsed.mSec
Value: 330539

Key : Analysis.Memory.CommitPeak.Mb
Value: 145

Key : Failure.Bucket
Value: BREAKPOINT_80000003_win32u.dll!NtUserGetMessage

Key : Failure.Hash
Value: {3112b5eb-303b-e877-0655-90bdfa336126}

Key : Timeline.OS.Boot.DeltaSec
Value: 1274

Key : Timeline.Process.Start.DeltaSec
Value: 291

Key : WER.OS.Branch
Value: co_release

Key : WER.OS.Version
Value: 10.0.22000.1

Key : WER.Process.Version
Value: 10.0.22000.1

FILE_IN_CAB: wordpad.DMP

NTGLOBALFLAG: 400

APPLICATION_VERIFIER_FLAGS: 0

EXCEPTION_RECORD: (.exr -1)
ExceptionAddress: 0000000000000000
ExceptionCode: 80000003 (Break instruction exception)
ExceptionFlags: 00000000
NumberParameters: 0

FAULTING_THREAD: 000002d8

PROCESS_NAME: wordpad.exe

ERROR_CODE: (NTSTATUS) 0x80000003 - {EXCEPTION} Breakpoint A breakpoint has been reached.

EXCEPTION_CODE_STR: 80000003

STACK_TEXT:
00000037`1567fd48 00007ff9`15a646ae : 00007ff8`d328a460 00007ff8`d314b71b 00007ff8`d328a460 00007ff8`d314ba4b : win32u!NtUserGetMessage+0x14
00000037`1567fd50 00007ff8`d3150813 : 00007ff6`d9fddfd0 00000000`00000000 00000000`ffffffff 00000000`00000000 : user32!GetMessageW+0x2e
00000037`1567fdb0 00007ff8`d3150736 : 00000000`00000002 00000000`00000001 00000000`00000000 00000000`00000000 : mfc42u!CWinThread::PumpMessage+0x23
00000037`1567fde0 00007ff8`d314f2bc : 00000000`00000001 00007ff6`d9ee0000 00000000`00000000 00000164`9e3d6bec : mfc42u!CWinThread::Run+0x96
00000037`1567fe20 00007ff6`d9eebcfd : 00000000`00000001 00000000`00000000 00000000`00000000 00000000`0000001f : mfc42u!AfxWinMain+0xbc
00000037`1567fe60 00007ff9`14a754e0 : 00000000`00000000 00000000`00000000 00000000`00000000 00000000`00000000 : wordpad!__wmainCRTStartup+0x1dd
00000037`1567ff20 00007ff9`1668485b : 00000000`00000000 00000000`00000000 00000000`00000000 00000000`00000000 : kernel32!BaseThreadInitThunk+0x10
00000037`1567ff50 00000000`00000000 : 00000000`00000000 00000000`00000000 00000000`00000000 00000000`0000002b : ntdll!RtlUserThreadStart+0x2b

STACK_COMMAND: ~0s; .ecxr ; kb

SYMBOL_NAME: win32u!NtUserGetMessage+14

MODULE_NAME: win32u

IMAGE_NAME: win32u.dll

FAILURE_BUCKET_ID: BREAKPOINT_80000003_win32u.dll!NtUserGetMessage

OS_VERSION: 10.0.22000.1

BUILDLAB_STR: co_release

OSPLATFORM_TYPE: x64

OSNAME: Windows 10

IMAGE_VERSION: 10.0.22000.434

FAILURE_ID_HASH: {3112b5eb-303b-e877-0655-90bdfa336126}

Followup: MachineOwner
-----
```

Note: “Break instruction exception” can be the sign of a **Manual Dump** pattern, but often WinDbg is not able to figure out an exception that may be on another thread or hidden. **STACK_COMMAND** shows the sequence of commands that WinDbg executed to get **STACK_TEXT**.

8. Now we check how many threads there are by using the ~ command:

```
0:000> ~
. 0 Id: 118c.2d8 Suspend: 0 Teb: 00000037`15400000 Unfrozen
  1 Id: 118c.1064 Suspend: 0 Teb: 00000037`15408000 Unfrozen
  2 Id: 118c.2108 Suspend: 0 Teb: 00000037`1540a000 Unfrozen
  3 Id: 118c.50c Suspend: 0 Teb: 00000037`1540c000 Unfrozen
```

Note: 118c is Process ID (PID), and 2d8 is Thread ID (TID). 118c.2d8 is called the Client ID (CID).

9. Now we dump a stack trace using the kc command (only modules and symbols):

```
0:000> kc
# Call Site
00 win32u!NtUserGetMessage
01 user32!GetMessageW
02 mfc42u!CWinThread::PumpMessage
03 mfc42u!CWinThread::Run
04 mfc42u!AfxWinMain
05 wordpad!__wmainCRTStartup
06 kernel32!BaseThreadInitThunk
07 ntdll!RtlUserThreadStart
```

10. Now we dump the stack trace of the current thread using the k command (with symbols, return addresses, and function offsets):

```
0:000> k
# Child-SP          RetAddr           Call Site
00 00000037`1567fd48 00007ff9`15a6464e win32u!NtUserGetMessage+0x14
01 00000037`1567fd50 00007ff8`d3150813 user32!GetMessageW+0x2e
02 00000037`1567fdb0 00007ff8`d3150736 mfc42u!CWinThread::PumpMessage+0x23
03 00000037`1567fde0 00007ff8`d314f2bc mfc42u!CWinThread::Run+0x96
04 00000037`1567fe20 00007ff6`d9eebcfd mfc42u!AfxWinMain+0xbc
05 00000037`1567fe60 00007ff9`14a754e0 wordpad!__wmainCRTStartup+0x1dd
06 00000037`1567ff20 00007ff9`1668485b kernel32!BaseThreadInitThunk+0x10
07 00000037`1567ff50 00000000`00000000 ntdll!RtlUserThreadStart+0x2b
```

Hint: How to check that the stack trace is correct. Use the **ub** (unassemble **backward**) command to check if there is a *call* instruction. We check that the *GetMessageW* function was called from the *CWinThread::PumpMessage* function:

```
0:000> k
# Child-SP          RetAddr           Call Site
00 00000037`1567fd48 00007ff9`15a6464e win32u!NtUserGetMessage+0x14
01 00000037`1567fd50 00007ff8`d3150813 user32!GetMessageW+0x2e
02 00000037`1567fdb0 00007ff8`d3150736 mfc42u!CWinThread::PumpMessage+0x23
03 00000037`1567fde0 00007ff8`d314f2bc mfc42u!CWinThread::Run+0x96
04 00000037`1567fe20 00007ff6`d9eebcfd mfc42u!AfxWinMain+0xbc
```

Practical Exercises

```

05 00000037`1567fe60 00007ff9`14a754e0 wordpad!__wmainCRTStartup+0x1dd
06 00000037`1567ff20 00007ff9`1668485b kernel32!BaseThreadInitThunk+0x10
07 00000037`1567ff50 00000000`00000000 ntdll!RtlUserThreadStart+0x2b

0:000> ub 00007ff8`d3150813
mfc42u!CWinThread::PumpMessage+0x9:
00007ff8`d31507f9 20488d and byte ptr [rax-73h],cl
00007ff8`d31507fc 59 pop rcx
00007ff8`d31507fd 68488bf948 push 48F98B48h
00007ff8`d3150802 8bcb mov ecx,ebx
00007ff8`d3150804 4533c9 xor r9d,r9d
00007ff8`d3150807 4533c0 xor r8d,r8d
00007ff8`d315080a 33d2 xor edx,edx
00007ff8`d315080c 48ff152dc00f00 call qword ptr [mfc42u!_imp_GetMessageW (00007ff8`d324c840)]

0:000> dps 00007ff8`d324c840 L1
00007ff8`d324c840 00007ff9`15a64620 user32!GetMessageW

```

Then we check that the *NtUserGetMessage* function was called from the *GetMessageW* function:

```

0:000> k
# Child-SP RetAddr Call Site
00 00000037`1567fd48 00007ff9`15a6464e win32u!NtUserGetMessage+0x14
01 00000037`1567fd50 00007ff8`d3150813 user32!GetMessageW+0x2e
02 00000037`1567fdb0 00007ff8`d3150736 mfc42u!CWinThread::PumpMessage+0x23
03 00000037`1567fde0 00007ff8`d314f2bc mfc42u!CWinThread::Run+0x96
04 00000037`1567fe20 00007ff6`d9eebcfd mfc42u!AfxWinMain+0xbc
05 00000037`1567fe60 00007ff9`14a754e0 wordpad!__wmainCRTStartup+0x1dd
06 00000037`1567ff20 00007ff9`1668485b kernel32!BaseThreadInitThunk+0x10
07 00000037`1567ff50 00000000`00000000 ntdll!RtlUserThreadStart+0x2b

0:000> ub 00007ff9`15a6464e
user32!GetMessageW+0x9:
00007ff9`15a64629 488bd9 mov rbx,rcx
00007ff9`15a6462c 458bc8 mov r9d,r8d
00007ff9`15a6462f 440bc8 or r9d,eax
00007ff9`15a64632 41f7c10000feff test r9d,0FFFFFF000h
00007ff9`15a64639 0f85c70b0200 jne user32!GetMessageW+0x20be6 (00007ff9`15a85206)
00007ff9`15a6463f 448bc8 mov r9d,eax
00007ff9`15a64642 48897c2460 mov qword ptr [rsp+60h],rdi
00007ff9`15a64647 48ff158a260700 call qword ptr [user32!_imp_NtUserGetMessage (00007ff9`15ad6cd8)]

0:000> dps 00007ff9`15ad6cd8 L1
00007ff9`15ad6cd8 00007ff9`13c31400 win32u!NtUserGetMessage

```

Note: Remember that the functions call each other from bottom to top. The topmost function in the stack trace is the last one called. **ExceptionAddress** may point to the last one. We will come to this in the real exception process dumps later.

11. Now we check the list of loaded modules using the **lm** command:

```

0:000> lm
start end module_name
00007ff6`d9ee0000 00007ff6`da1cd000 wordpad (pdb symbols) C:\WinDbg.Docker.AwMDA6\mss\wordpad.pdb\B193BA11D609CB39E8D086A748A191651\wordpad.pdb
00007ff8`bafe0000 00007ff8`bb39c000 UIRibbon (deferred)
00007ff8`d3140000 00007ff8`d32b4000 mfc42u (pdb symbols) C:\WinDbg.Docker.AwMDA6\mss\mfc42u.pdb\C250069F808FD6D342ADAE9524B0F1EE1\mfc42u.pdb
00007ff8`e0c80000 00007ff8`e0eb5000 opcservices (deferred)
00007ff8`e0ec0000 00007ff8`e112e000 msxml3 (deferred)
00007ff8`f4cb0000 00007ff8`f4d11000 AcGenral (deferred)
00007ff8`f5320000 00007ff8`f5596000 xpsservices (deferred)
00007ff8`f55a0000 00007ff8`f567b000 MXDWDVR (deferred)

```

```

00007ff8`f5680000 00007ff8`f5a3e000 PrintConfig (deferred)
00007ff8`f6620000 00007ff8`f662d000 atlthunk (deferred)
00007ff8`f6630000 00007ff8`f67e3000 GdiPlus (deferred)
00007ff8`f7470000 00007ff8`f77d6000 msftedit (deferred)
00007ff8`f7d70000 00007ff8`f7d96000 globinpthost (deferred)
00007ff8`fa3c0000 00007ff8`fa41d000 dataexchange (deferred)
00007ff8`fa420000 00007ff8`fa489000 oleacc (deferred)
00007ff8`fb530000 00007ff8`fb59a000 ninput (deferred)
00007ff8`fb8d0000 00007ff8`fb8f3000 fontsub (deferred)
00007ff8`fcd00000 00007ff8`fd072000 msxml6 (deferred)
00007ff8`ff650000 00007ff8`ff6eb000 winspool (deferred)
00007ff9`00880000 00007ff9`00b25000 comctl32 (deferred)
00007ff9`01170000 00007ff9`01246000 jscript (deferred)
00007ff9`01840000 00007ff9`0189b000 XpsPushLayer (deferred)
00007ff9`01df0000 00007ff9`01e6b000 MpOAV (deferred)
00007ff9`04070000 00007ff9`0408d000 mpr (deferred)
00007ff9`07360000 00007ff9`0740e000 TextShaping (deferred)
00007ff9`074e0000 00007ff9`0760d000 textinputframework (deferred)
00007ff9`08120000 00007ff9`0812a000 version (deferred)
00007ff9`08130000 00007ff9`08163000 winmm (deferred)
00007ff9`08360000 00007ff9`08522000 Windows_Globalization (deferred)
00007ff9`08530000 00007ff9`0878f000 DWrite (deferred)
00007ff9`08790000 00007ff9`08fa6000 OneCoreUAPCommonProxyStub (deferred)
00007ff9`0a580000 00007ff9`0a72e000 windowscodecs (deferred)
00007ff9`0a770000 00007ff9`0a8f8000 Windows_UI (deferred)
00007ff9`0b5e0000 00007ff9`0b645000 Bcp47Langs (deferred)
00007ff9`0c0d0000 00007ff9`0c0e2000 npmproxy (deferred)
00007ff9`0c5b0000 00007ff9`0c816000 twinapi_appcore (pdb symbols)
C:\WinDbg.Docker.AWMDA6\mss\twinapi_appcore.pdb\4FD957C7E31B8C581E01125EC191B7F81\twinapi_appcore.pdb
00007ff9`0cda0000 00007ff9`0cdc8000 srvccli (deferred)
00007ff9`0cdd0000 00007ff9`0d082000 iertutil (deferred)
00007ff9`0d100000 00007ff9`0d2ee000 urlmon (deferred)
00007ff9`0d850000 00007ff9`0d869000 dhcpcsvc6 (deferred)
00007ff9`0db20000 00007ff9`0db3e000 dhcpcsvc (deferred)
00007ff9`0e250000 00007ff9`0e2c1000 netprofm (deferred)
00007ff9`0e670000 00007ff9`0e9dd000 CoreUIComponents (deferred)
00007ff9`0f140000 00007ff9`0f1d2000 msvcpl10_win (deferred)
00007ff9`0f6a0000 00007ff9`0f6d1000 prntvpt (deferred)
00007ff9`0f820000 00007ff9`0f857000 xmlite (deferred)
00007ff9`10930000 00007ff9`10a62000 CoreMessaging (deferred)
00007ff9`10d70000 00007ff9`10e01000 apphelp (deferred)
00007ff9`10e30000 00007ff9`10edc000 uxtheme (deferred)
00007ff9`113c0000 00007ff9`113ef000 dwmapi (deferred)
00007ff9`118d0000 00007ff9`119c7000 propsys (deferred)
00007ff9`11b70000 00007ff9`11cd6000 WinTypes (deferred)
00007ff9`11ce0000 00007ff9`12547000 windows_storage (deferred)
00007ff9`126b0000 00007ff9`126bc000 netutils (deferred)
00007ff9`127e0000 00007ff9`1280d000 IPHLPAPI (deferred)
00007ff9`12860000 00007ff9`12947000 dnsapi (deferred)
00007ff9`12c70000 00007ff9`12c88000 kernel_appcore (deferred)
00007ff9`12e90000 00007ff9`12ed2000 sspicli (deferred)
00007ff9`13200000 00007ff9`13229000 userenv (deferred)
00007ff9`13350000 00007ff9`1335c000 CRYPTBASE (deferred)
00007ff9`13550000 00007ff9`13577000 bcrypt (deferred)
00007ff9`139e0000 00007ff9`13a82000 sxs (deferred)
00007ff9`13aa0000 00007ff9`13ac1000 profapi (deferred)
00007ff9`13c30000 00007ff9`13c56000 win32u (pdb symbols) C:\WinDbg.Docker.AWMDA6\mss\win32u.pdb\045A07FC5CC3A90DFCCC8B4C1918F7421\win32u.pdb
00007ff9`13c60000 00007ff9`13cdf000 msvcpl10_win (deferred)
00007ff9`13d00000 00007ff9`13e12000 gdi32full (deferred)
00007ff9`13f90000 00007ff9`14304000 KERNELBASE (pdb symbols)
C:\WinDbg.Docker.AWMDA6\mss\kernelbase.pdb\AF202873637A4CAABB4ACB056CE0BCCA1\kernelbase.pdb
00007ff9`14380000 00007ff9`143ff000 bcryptPrimitives (deferred)
00007ff9`14400000 00007ff9`14511000 ucrtbase (deferred)
00007ff9`14590000 00007ff9`14908000 combase (private pdb symbols) C:\WinDbg.Docker.AWMDA6\mss\combase.pdb\FB29C6C2977E6207AAC857DCE3D9183C1\combase.pdb
00007ff9`14920000 00007ff9`149c3000 msvcrt (deferred)
00007ff9`14a60000 00007ff9`14b1d000 kernel32 (pdb symbols)
C:\WinDbg.Docker.AWMDA6\mss\kernel32.pdb\DC094362EEB8DA89986E90A2096ACE281\kernel32.pdb
00007ff9`14b20000 00007ff9`14b49000 gdi32 (deferred)
00007ff9`14b70000 00007ff9`1531e000 shell32 (deferred)
00007ff9`15320000 00007ff9`1540c000 comdlg32 (deferred)
00007ff9`15410000 00007ff9`1552e000 msctf (deferred)
00007ff9`15760000 00007ff9`157bd000 shlwapi (deferred)
00007ff9`157c0000 00007ff9`158aa000 SHCore (deferred)
00007ff9`15910000 00007ff9`15a30000 rpcrt4 (deferred)
00007ff9`15a30000 00007ff9`15a39000 nsi (deferred)
00007ff9`15a40000 00007ff9`15bec000 user32 (pdb symbols) C:\WinDbg.Docker.AWMDA6\mss\user32.pdb\9479B9C8D8218B8972152084F4D6840C1\user32.pdb
00007ff9`15bf0000 00007ff9`15c9f000 clbcatq (deferred)
00007ff9`15ca0000 00007ff9`15d3e000 sechost (deferred)
00007ff9`15d40000 00007ff9`15dee000 advapi32 (deferred)
00007ff9`15e60000 00007ff9`15f36000 oleaut32 (deferred)
00007ff9`15f70000 00007ff9`1610a000 ole32 (private pdb symbols) C:\WinDbg.Docker.AWMDA6\mss\ole32.pdb\FC5E57B29784316F10EE67D8895FF8341\ole32.pdb
00007ff9`16190000 00007ff9`161c1000 imm32 (deferred)
00007ff9`16680000 00007ff9`16890000 ntdll (pdb symbols) C:\WinDbg.Docker.AWMDA6\mss\ntdll.pdb\D522F52AE48D7166E35F4FB492B6398B1\ntdll.pdb

Unloaded modules:
00007ff9`01e70000 00007ff9`01e8d000 amsi.dll
00007ff9`01e70000 00007ff9`01e8d000 amsi.dll
00007ff9`01e70000 00007ff9`01e8d000 amsi.dll
00007ff9`01e70000 00007ff9`01e8d000 amsi.dll
00007ff9`06000000 00007ff9`06023000 compstui.dll
00007ff9`0f720000 00007ff9`0f727000 MSIMG32.dll
00007ff9`01e70000 00007ff9`01e8d000 amsi.dll
00007ff9`01e70000 00007ff9`01e8d000 amsi.dll
00007ff9`06000000 00007ff9`06023000 compstui.dll
00007ff9`0f720000 00007ff9`0f727000 MSIMG32.dll
00007ff9`01e70000 00007ff9`01e8d000 amsi.dll

```

Practical Exercises

00007ff9`02970000 00007ff9`029a8000 fms.dll

Note: **start** and **end** addresses show where modules are loaded in process virtual memory. You can see the module contents by using the **dc** command (**Unknown Component** pattern):

```
0:000> dc 00007ff6`d9ee0000 00007ff6`da1cd000
00007ff6`d9ee0000 00905a4d 00000003 00000004 0000ffff MZ.....
00007ff6`d9ee0010 000000b8 00000000 00000040 00000000 .....@.....
00007ff6`d9ee0020 00000000 00000000 00000000 00000000 .....
00007ff6`d9ee0030 00000000 00000000 00000000 000000f0 .....
00007ff6`d9ee0040 0eba1f0e cd09b400 4c01b821 685421cd .....!..L.!Th
00007ff6`d9ee0050 70207369 72676f72 63206d61 6f6e6e61 is program canno
00007ff6`d9ee0060 65622074 6e757220 206e6920 20534f44 t be run in DOS
00007ff6`d9ee0070 65646f6d 0a0d0d2e 00000024 00000000 mode....$.
00007ff6`d9ee0080 cc9a0e95 9ff46fd1 9ff46fd1 9ff46fd1 .....o...o...o..
00007ff6`d9ee0090 9ef71d02 9ff46fd5 9ef01d02 9ff46fce .....o.....o..
00007ff6`d9ee00a0 9ef11d02 9ff46fda 9ef51d02 9ff46ff6 .....o.....o..
00007ff6`d9ee00b0 9ff56fd1 9ff46b3f 9efc1d02 9ff46fa2 .o..?k.....o..
00007ff6`d9ee00c0 9f0b1d02 9ff46fd0 9ef61d02 9ff46fd0 .....o.....o..
00007ff6`d9ee00d0 68636952 9ff46fd1 00000000 00000000 Rich.o.....
00007ff6`d9ee00e0 00000000 00000000 00000000 00000000 .....
00007ff6`d9ee00f0 00004550 00078664 b930df5e 00000000 PE..d...^.0.....
00007ff6`d9ee0100 00000000 002200f0 1c0e020b 0009f000 .....".
00007ff6`d9ee0110 0024d000 00000000 0000bda0 00001000 ..$.
[...]
```

Note: The list of unloaded modules is an example of historical information that may indicate past system behavior.

12. We can check verbose module information using the **lmv** command or use **lmv m <module name>** to check an individual module (**Not My Version** pattern):

```
0:000> lmv m wordpad
Browse full module list
start          end          module name
00007ff6`d9ee0000 00007ff6`da1cd000 wordpad (pdb symbols)
C:\WinDbg.Docker.AWMDA6\mss\wordpad.pdb\B193BA11D609CB39E8D086A748A191651\wordpad.pdb
Loaded symbol image file: wordpad.exe
Image path: C:\Program Files\Windows NT\Accessories\wordpad.exe
Image name: wordpad.exe
Browse all global symbols functions data
Image was built with /Brepro flag.
Timestamp:      B930DF5E (This is a reproducible build file hash, not a timestamp)
Checksum:        002F073E
ImageSize:       002ED000
File version:    10.0.22000.1
Product version: 10.0.22000.1
File flags:      0 (Mask 3F)
```

```

File OS:          40004 NT Win32
File type:        1.0 App
File date:        00000000.00000000
Translations:     0409.04b0
Information from resource tables:
  CompanyName:    Microsoft Corporation
  ProductName:    Microsoft® Windows® Operating System
  InternalName:   wordpad
  OriginalFilename: WORDPAD.EXE
  ProductVersion: 10.0.22000.1
  FileVersion:    10.0.22000.1 (WinBuild.160101.0800)
  FileDescription: Windows Wordpad Application
  LegalCopyright: © Microsoft Corporation. All rights reserved

```

Note: If we see a real timestamp rather than a reproducible build file hash, it may indicate a third-party module.

13. Sometimes the **!mv** command doesn't show much, and the **!mi** command might give extra information, such as development-related information:

```

0:000> !mi wordpad
Loaded Module Info: [wordpad]
  Module: wordpad
  Base Address: 00007ff6d9ee0000
  Image Name: wordpad.exe
  Machine Type: 34404 (X64)
  Time Stamp: b930df5e (This is a reproducible build file hash, not a true timestamp)
  Size: 2ed000
  CheckSum: 2f073e
Characteristics: 22
Debug Data Dirs:
  Type      Size      VA      Pointer
  CODEVIEW  24, e7b2c,   e7b2c RSDS - GUID: {B193BA11-D609-CB39-E8D0-86A748A19165}
  Age: 1, Pdb: wordpad.pdb
  POGO      48c, e7b50,   e7b50 [Data not mapped]
  REPRO      24, e7fdc,   e7fdc Reproducible build
Image Type: MEMORY - Image read successfully from loaded memory.
Symbol Type: PDB   - Symbols loaded successfully from symbol server.

C:\WinDbg.Docker.AWMDA6\mss\wordpad.pdb\B193BA11D609CB39E8D086A748A191651\wordpad.pdb
Load Report: public symbols , not source indexed

C:\WinDbg.Docker.AWMDA6\mss\wordpad.pdb\B193BA11D609CB39E8D086A748A191651\wordpad.pdb

```

Note: We can also use the **!mt** command variant if we are interested only in timestamps.

14. Sometimes **Environment Hint** pattern can give troubleshooting suggestions related to environment variables and DLL paths. **!peb** command (**P**rocess **E**nvironment **B**lock):

```

0:000> !peb
PEB at 00000037155ff000
  InheritedAddressSpace: No
  ReadImageFileExecOptions: No
  BeingDebugged: No
  ImageBaseAddress: 00007ff6d9ee0000
  NtGlobalFlag: 400
  NtGlobalFlag2: 0
  Ldr: 00007ff9167fa120

```

Practical Exercises

```

Ldr.Initialized:                Yes
Ldr.InInitializationOrderModuleList: 000001649e3d1fd0 . 00000164a097d4d0
Ldr.InLoadOrderModuleList:       000001649e3d2150 . 00000164a097d4b0
Ldr.InMemoryOrderModuleList:     000001649e3d2160 . 00000164a097d4c0

```

Base	TimeStamp	Module
7ff6d9ee0000	b930df5e Jun 15 13:11:10 2068	C:\Program Files\Windows NT\Accessories\wordpad.exe
7ff916680000	931cda92 Mar 18 10:55:14 2048	C:\WINDOWS\SYSTEM32\ntdll.dll
7ff914a60000	7b65e245 Aug 09 13:17:09 2035	C:\WINDOWS\System32\KERNEL32.DLL
7ff913f90000	72a6f702 Dec 15 06:00:34 2030	C:\WINDOWS\System32\KERNELBASE.dll
7ff910d70000	3c3af44a Jan 08 13:29:46 2002	C:\WINDOWS\SYSTEM32\apphelp.dll
7ff8f4cb0000	0e564edf Aug 16 00:55:11 1977	C:\WINDOWS\SYSTEM32\AcGenral.dll
7ff914920000	90483ed2 Sep 15 20:49:38 2046	C:\WINDOWS\System32\msvcrt.dll
7ff915ca0000	31ec7be5 Jul 17 06:36:37 1996	C:\WINDOWS\System32\sechost.dll
7ff915760000	5d809272 Sep 17 08:59:46 2019	C:\WINDOWS\System32\SHLWAPI.dll
7ff915a40000	95c2e8f0 Aug 14 19:33:20 2049	C:\WINDOWS\System32\USER32.dll
7ff913c30000	2eab7211 Oct 24 09:36:33 1994	C:\WINDOWS\System32\win32u.dll
7ff914b20000	0b2998f3 Dec 08 12:58:27 1975	C:\WINDOWS\System32\GDI32.dll
7ff913c00000	f03395da Sep 13 13:08:58 2097	C:\WINDOWS\System32\gdi32full.dll
7ff913c60000	1fb7fd57 Nov 12 03:53:59 1986	C:\WINDOWS\System32\msvc_p_win.dll
7ff914400000	00e78ce9 Jun 25 16:14:49 1970	C:\WINDOWS\System32\ucrtbase.dll
7ff915f70000	8dfb3d4d Jun 26 02:18:05 2045	C:\WINDOWS\System32\ole32.dll
7ff914590000	426c1ced Apr 24 23:25:49 2005	C:\WINDOWS\System32\combase.dll
7ff915910000	7ff0ec4a Jan 07 16:46:02 2038	C:\WINDOWS\System32\RPCRT4.dll
7ff914b70000	8cba58e5 Oct 25 16:38:13 2044	C:\WINDOWS\System32\SHELL32.dll
7ff915d40000	ce622c7b Sep 21 17:46:51 2079	C:\WINDOWS\System32\ADVAPI32.dll
7ff913200000	a3572516 Nov 02 13:28:54 2056	C:\WINDOWS\SYSTEM32\USERENV.dll
7ff904070000	4ab16881 Sep 16 23:36:49 2009	C:\WINDOWS\SYSTEM32\MPR.dll
7ff912e90000	e2336ada Apr 04 21:21:46 2090	C:\WINDOWS\SYSTEM32\SspiCli.dll
7ff916190000	356942c7 May 25 11:07:03 1998	C:\WINDOWS\System32\IMM32.DLL
7ff915320000	b5c44fd4 Aug 20 15:53:08 2066	C:\WINDOWS\System32\COMDLG32.dll
7ff9157c0000	d40bc30a Sep 25 06:43:38 2082	C:\WINDOWS\System32\shcore.dll
7ff915e60000	f6e2d5cf Apr 04 13:30:07 2101	C:\WINDOWS\System32\OLEAUT32.dll
7ff90f140000	7a1c0743 Dec 02 07:26:59 2034	C:\WINDOWS\SYSTEM32\msvc_p110_win.dll
7ff8d3140000	f91a937d Jun 09 04:54:37 2102	C:\WINDOWS\SYSTEM32\MFC42u.dll
7ff900880000	150b8699 Mar 10 12:54:49 1981	C:\WINDOWS\WinSxS\amd64_microsoft.windows.common-
controls_6595b64144ccf1df_6.0.22000.120_none_9d947278b86cc467\COMCTL32.dll		
7ff908130000	4b928681 Mar 06 16:44:49 2010	C:\WINDOWS\SYSTEM32\WINMM.dll
7ff90d100000	cc1588be Jul 02 05:55:26 2078	C:\WINDOWS\SYSTEM32\urlmon.dll
7ff90f820000	ced9ec48 Dec 21 12:44:56 2079	C:\WINDOWS\SYSTEM32\XmlLite.dll
7ff90cdd0000	5a2fa526 Dec 12 09:45:10 2017	C:\WINDOWS\SYSTEM32\iertutil.dll
7ff90cda0000	35be966e Jul 29 04:26:38 1998	C:\WINDOWS\SYSTEM32\svcl.dll
7ff9118d0000	c2756dbe May 20 04:15:10 2073	C:\WINDOWS\SYSTEM32\PROPSYS.dll
7ff9126b0000	813aa4df Sep 14 20:09:19 2038	C:\WINDOWS\SYSTEM32\netutils.dll
7ff8fb530000	cc168813 Jul 03 00:04:51 2078	C:\WINDOWS\SYSTEM32\ninput.dll
7ff912c70000	fb20135b Jul 06 17:42:03 2103	C:\WINDOWS\SYSTEM32\kernel.appcore.dll
7ff914380000	a34302f0 Oct 18 07:57:52 2056	C:\WINDOWS\System32\bcryptPrimitives.dll
7ff910e30000	e2c027fe Jul 20 15:26:06 2090	C:\WINDOWS\system32\uxtheme.dll
7ff915bf0000	1d473905 Jul 26 07:21:57 1985	C:\WINDOWS\System32\clbcatq.dll
7ff8e0ec0000	9e59ff97 Mar 09 19:44:23 2054	C:\WINDOWS\System32\msxml3.dll
7ff913550000	54fe428f Mar 10 01:02:07 2015	C:\WINDOWS\System32\bcrypt.dll
7ff8f7470000	fc008760 Dec 23 22:44:48 2103	C:\WINDOWS\SYSTEM32\MSFTEDIT.DLL
7ff8baf60000	9a0b9171 Nov 24 14:36:33 2051	C:\WINDOWS\system32\UIRibbon.dll
7ff8f6630000	dc58dab9 Feb 23 03:51:21 2087	
C:\WINDOWS\WinSxS\amd64_microsoft.windows.gdiplus_6595b64144ccf1df_1.1.22000.434_none_ce836c1412fb9b57\gdiplus.dll		
7ff915410000	81def127 Jan 17 10:06:31 2039	C:\WINDOWS\System32\MSCTF.dll
7ff907360000	6627ed04 Apr 23 18:16:52 2024	C:\WINDOWS\SYSTEM32\TextShaping.dll
7ff908360000	10fbb3fc Jan 11 19:31:08 1979	C:\Windows\System32\Windows.Globalization.dll
7ff8f7d70000	c9e14921 Apr 30 06:05:37 2077	C:\WINDOWS\SYSTEM32\globinpuhost.dll
7ff90b5e0000	10434404 Aug 24 22:56:20 1978	C:\WINDOWS\SYSTEM32\Bcp47Langs.dll
7ff8fa3c0000	511f48d8 Feb 16 08:52:40 2013	C:\WINDOWS\system32\dataexchange.dll
7ff90c5b0000	d6129e9c Oct 23 20:14:36 2083	C:\WINDOWS\system32\twinapi.appcore.dll
7ff90f4e0000	63938554 Dec 09 18:58:28 2022	C:\WINDOWS\SYSTEM32\textinputframework.dll
7ff8fa420000	d4726d59 Dec 12 02:41:29 2082	C:\Windows\System32\oleacc.dll
7ff8f6620000	f95e4869 Jul 30 13:28:25 2102	C:\WINDOWS\SYSTEM32\atlthunk.dll
7ff9113c0000	09360bc9 Nov 24 14:55:05 1974	C:\WINDOWS\system32\dwmap.dll
7ff8fcd40000	bd9922fe Oct 19 08:36:30 2070	C:\Windows\System32\msxml6.dll
7ff90a580000	1decf0c2 Nov 28 23:09:54 1985	C:\WINDOWS\system32\windowscodecs.dll
7ff90a770000	2a4aa2e7 Jun 26 05:53:59 1992	C:\Windows\System32\Windows.UI.dll
7ff910930000	9e78ed02 Apr 02 07:45:22 2054	C:\WINDOWS\SYSTEM32\CoreMessaging.dll
7ff90e670000	6685eb5c Jul 04 01:22:52 2024	C:\WINDOWS\SYSTEM32\CoreUIComponents.dll
7ff911b70000	b3354271 Apr 10 19:01:21 2065	C:\WINDOWS\SYSTEM32\wintypes.dll
7ff913350000	14759998 Nov 16 19:35:52 1980	C:\WINDOWS\SYSTEM32\CRYPTBASE.DLL

```

7ff911ce0000 42c927b5 Jul 04 13:12:37 2005 C:\WINDOWS\SYSTEM32\windows.storage.dll
7ff8ff650000 fdeb754 Dec 30 13:40:36 2104 C:\WINDOWS\SYSTEM32\WINSPOOL.DRV
7ff90e250000 2e513767 Aug 16 23:02:15 1994 C:\WINDOWS\System32\netprofm.dll
7ff90c0d0000 f288926c Dec 10 07:57:32 2098 C:\WINDOWS\System32\npmproxy.dll
7ff9127e0000 8c5d7fee Aug 16 06:23:58 2044 C:\WINDOWS\SYSTEM32\IPHLAPI.DLL
7ff915a30000 1de8145c Nov 25 06:40:28 1985 C:\WINDOWS\System32\NSI.dll
7ff90d850000 bda0ed88 Oct 25 06:26:32 2070 C:\WINDOWS\SYSTEM32\dhcpcsvc6.DLL
7ff90db20000 7d8aeb85 Sep 29 00:11:01 2036 C:\WINDOWS\SYSTEM32\dhcpcsvc.DLL
7ff912860000 64c37de6 Jul 28 09:35:50 2023 C:\WINDOWS\SYSTEM32\DNSAPI.dll
7ff8f5680000 922ff4f6 Sep 20 19:20:38 2047
C:\WINDOWS\System32\DriverStore\FileRepository\prnms003.inf_amd64_03994fdd96c52654\Amd64\PrintConfig.dll
7ff908120000 12bfcbe0 Dec 20 17:37:36 1979 C:\WINDOWS\SYSTEM32\VERSION.dll
7ff90f6a0000 c491446b Jul 03 15:42:51 2074 C:\WINDOWS\SYSTEM32\prntvpt.dll
7ff901170000 0549fc92 Oct 24 01:34:26 1972 C:\Windows\System32\jscript.dll
7ff913aa0000 47c07815 Feb 23 19:46:29 2008 C:\WINDOWS\SYSTEM32\profapi.dll
7ff901df0000 8e940251 Oct 19 23:23:13 2045 C:\ProgramData\Microsoft\Windows
Defender\Platform\4.18.23050.5-0\MpOav.dll
7ff9139e0000 a2eb73f0 Aug 12 22:00:32 2056 C:\WINDOWS\SYSTEM32\sxs.dll
7ff8f55a0000 89c5dd1c Mar 31 21:16:28 2043
C:\WINDOWS\System32\DriverStore\FileRepository\ntprint.inf_amd64_69e8e0efb212ba16\Amd64\mxwdwdrv.dll
7ff908530000 13ccbe72 Jul 11 18:40:02 1980 C:\WINDOWS\SYSTEM32\Dwrite.dll
7ff908790000 9028fb7a Aug 23 03:42:02 2046 C:\Windows\System32\OneCoreUAPCommonProxyStub.dll
7ff8e0c80000 399cd7ea Aug 18 07:30:02 2000 C:\WINDOWS\SYSTEM32\opcservices.dll
7ff901840000 50fd98e3 Jan 21 19:37:07 2013 C:\WINDOWS\SYSTEM32\XpsPushLayer.dll
7ff8f5320000 f6f60cb7 Apr 19 03:17:27 2101 C:\WINDOWS\SYSTEM32\xpsservices.dll
7ff8fb8d0000 7c246416 Jan 01 00:21:42 2036 C:\WINDOWS\system32\FontSub.dll
SubSystemData: 0000000000000000
ProcessHeap: 000001649e3d0000
ProcessParameters: 000001649e3d64d0
CurrentDirectory: 'C:\Users\dumpa\Documents\'
WindowTitle: 'C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Accessories\Wordpad.lnk'
ImageFile: 'C:\Program Files\Windows NT\Accessories\wordpad.exe'
CommandLine: '"C:\Program Files\Windows NT\Accessories\wordpad.exe" '
DllPath: '< Name not readable >'
Environment: 000001649e3d11f0
=:.=:\
ALLUSERSPROFILE=C:\ProgramData
APPDATA=C:\Users\dumpa\AppData\Roaming
CommonProgramFiles=C:\Program Files\Common Files
CommonProgramFiles(x86)=C:\Program Files (x86)\Common Files
CommonProgramW6432=C:\Program Files\Common Files
COMPUTERNAME=DESKTOP-OGPC0LO
ComSpec=C:\WINDOWS\system32\cmd.exe
DriverData=C:\Windows\System32\Drivers\DriverData
FPS_BROWSER_APP_PROFILE_STRING=Internet Explorer
FPS_BROWSER_USER_PROFILE_STRING=Default
HOMEDRIVE=C:
HOMEPATH=\Users\dumpa
LOCALAPPDATA=C:\Users\dumpa\AppData\Local
LOGONSERVER=\\DESKTOP-OGPC0LO
NUMBER_OF_PROCESSORS=2
OneDrive=C:\Users\dumpa\OneDrive
OS=Windows_NT

Path=C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\System32\WindowsPowerShell\v1.0\;C:\WINDOWS\Sy
stem32\OpenSSH\;C:\Program Files\dotnet\;C:\Program Files
(x86)\dotnet\;C:\Users\dumpa\AppData\Local\Microsoft\WindowsApps;C:\Users\dumpa\.dotnet\tools;
PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTURE=AMD64
PROCESSOR_IDENTIFIER=Intel64 Family 6 Model 142 Stepping 10, GenuineIntel
PROCESSOR_LEVEL=6
PROCESSOR_REVISION=8e0a
ProgramData=C:\ProgramData
ProgramFiles=C:\Program Files
ProgramFiles(x86)=C:\Program Files (x86)
ProgramW6432=C:\Program Files
PSModulePath=C:\Program Files\WindowsPowerShell\Modules;C:\WINDOWS\system32\WindowsPowerShell\v1.0\Modules
PUBLIC=C:\Users\Public
SESSIONNAME=Console
SystemDrive=C:
SystemRoot=C:\WINDOWS
TEMP=C:\Users\dumpa\AppData\Local\Temp

```

Practical Exercises

```
TMP=C:\Users\dumpa\AppData\Local\Temp
USERDOMAIN=DESKTOP-OGPC0LO
USERDOMAIN_ROAMINGPROFILE=DESKTOP-OGPC0LO
USERNAME=Training
USERPROFILE=C:\Users\dumpa
windir=C:\WINDOWS
```

15. To launch classic help from the WinDbg app, type the **.hh** command.
16. We close logging before exiting WinDbg:

```
0:000> .logclose
Closing open log file C:\AwMDA-Dumps\Process\x64\wordpad.log
```

Note: If you close a log and later reopen it using the **.logopen** command, its contents will be lost. To append new output to an already existing log, please use the **.logappend** WinDbg command.