Project 13: Using Kernel Debugging Commands with WinDbg (15 pts.)

What You Need

- A Windows 10 machine with Livekd working, as prepared in the previous project. This project should work on Win 7 or any later version, but I only tested it on Windows 10. If you are using Windows Server 2008, this project works with one minor correction, as noted in the instructions for image B below.

Purpose

Practice using simple WinDbg commands.

Starting Configuration

You should have Livekd running, which launched WinDbg, as you did at the end of the previous project.

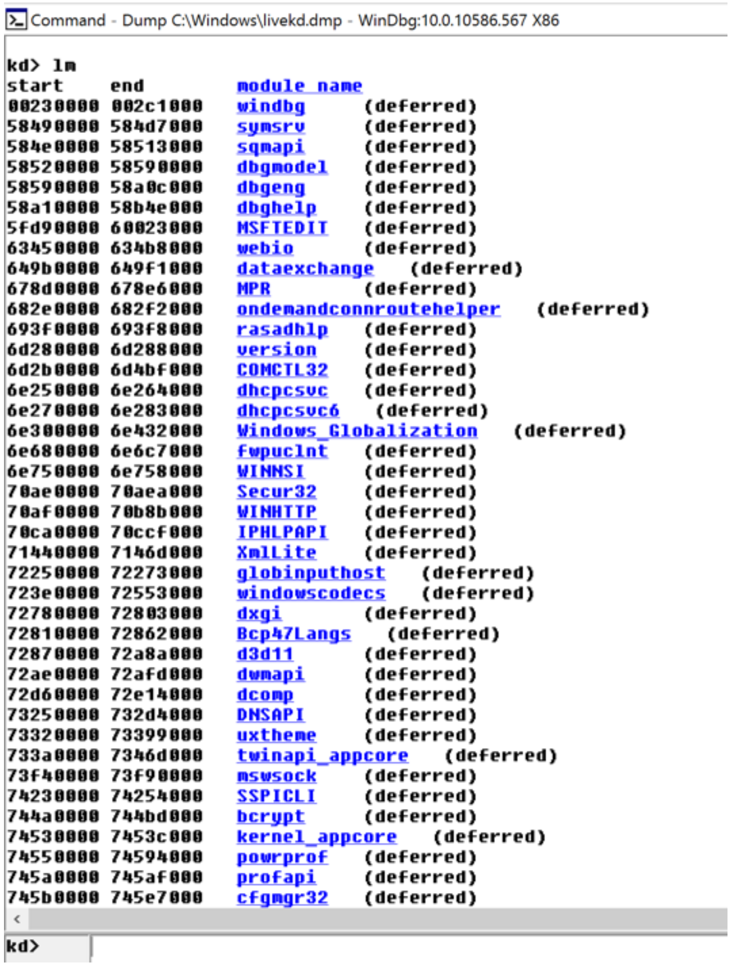
Listing Modules with lm

At the bottom of the Command window, in the command bar, execute this command:

```
lm
```

A long list of loaded modules scrolls by.

Scroll back to see the **lm** command you entered, and the first few loaded kernel modules, as shown below.

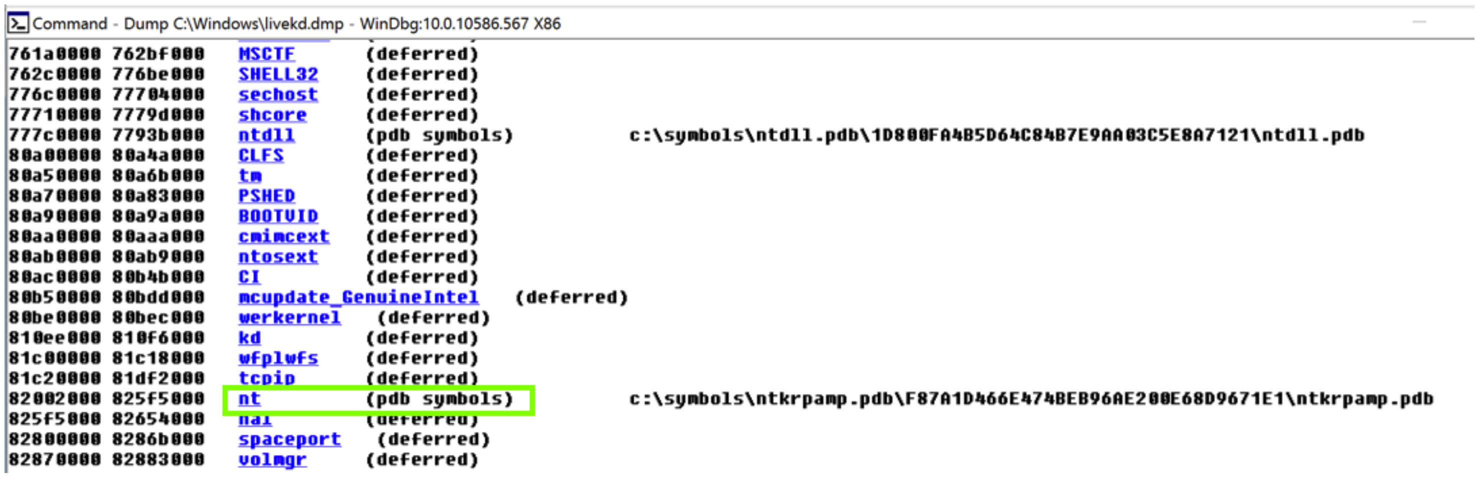


Command - Dump C:\Windows\livekd.dmp - WinDbg:10.0.10586.567 X86

```
kd> lm
start    end        module name
00230000 002c1000   windbg     (deferred)
58490000 584d7000   symsrv     (deferred)
584e0000 58513000   sqmapi     (deferred)
58520000 58590000   dbgmodel   (deferred)
58590000 58a0c000   dbgeng     (deferred)
58a10000 58b4e000   dbghelp    (deferred)
5fd90000 60023000   MSFTEDIT   (deferred)
63450000 634b8000   webio      (deferred)
649b0000 649f1000   dataexchange   (deferred)
678d0000 678e6000   MPR        (deferred)
682e0000 682f2000   ondemandconnroutehelper   (deferred)
693f0000 693f8000   rasadhlp   (deferred)
6d280000 6d288000   version    (deferred)
6d2b0000 6d4bf000   COMCTL32   (deferred)
6e250000 6e264000   dhcpcsvc   (deferred)
6e270000 6e283000   dhcpcsvc6   (deferred)
6e300000 6e432000   Windows_Globalization   (deferred)
6e680000 6e6c7000   fwpuclnt   (deferred)
6e750000 6e758000   WINNSI     (deferred)
70ae0000 70aea000   Secur32    (deferred)
70af0000 70b8b000   WINHTTP    (deferred)
70ca0000 70ccf000   IPHLPAPI   (deferred)
71440000 7146d000   XmlLite    (deferred)
72250000 72273000   globinputhost   (deferred)
723e0000 72553000   windowscodecs   (deferred)
72780000 72803000   dxgi       (deferred)
72810000 72862000   Bcp47Langs   (deferred)
72870000 72a8a000   d3d11      (deferred)
72ae0000 72afd000   dwmapi     (deferred)
72d60000 72e14000   dcomp      (deferred)
73250000 732d4000   DNSAPI     (deferred)
73320000 73399000   uxtheme    (deferred)
733a0000 7346d000   twinapi_appcore   (deferred)
73f40000 73f90000   mswsock    (deferred)
74230000 74254000   SSPICLI    (deferred)
744a0000 744bd000   bcrypt     (deferred)
74530000 7453c000   kernel_appcore   (deferred)
74550000 74594000   powrprof   (deferred)
745a0000 745af000   profapi    (deferred)
745b0000 745e7000   cfgmgr32   (deferred)
kd>
```

Scroll down to find the module named **nt**, as shown below. It's easy to spot because it'e one of the few modules that shows a Symbols path.

This is Ntoskrnl, the main kernel module.



Command - Dump C:\Windows\livekd.dmp - WinDbg:10.0.10586.567 X86

```
761a0000 762bf000   MSCTF      (deferred)
762c0000 776be000   SHELL32    (deferred)
776c0000 77704000   sechost    (deferred)
77710000 7779d000   shcore     (deferred)
777c0000 7793b000   ntdll      (pdb symbols)          c:\symbols\ntdll.pdb\1D800FA4B5D64C84B7E9AA03C5E8A7121\ntdll.pdb
80a00000 80a4a000   CLFS       (deferred)
80a50000 80a6b000   tm         (deferred)
80a70000 80a83000   PSHED      (deferred)
80a90000 80a9a000   BOOTVID    (deferred)
80aa0000 80aaa000   cmimcext   (deferred)
80ab0000 80ab9000   ntosext    (deferred)
80ac0000 80b4b000   CI         (deferred)
80b50000 80bdd000   mcupdate_GenuineIntel   (deferred)
80be0000 80bec000   werkernel   (deferred)
810ee000 810f6000   kd         (deferred)
81c00000 81c18000   wfplwfs    (deferred)
81c20000 81df2000   tcpip      (deferred)
82002000 825f5000   nt         (pdb symbols)          c:\symbols\ntkrpamp.pdb\F87A1D466E474BEB96AE200E68D9671E1\ntkrpamp.pdb
825f5000 82654000   hal        (deferred)
82800000 8286b000   spaceport   (deferred)
82870000 82883000   volmgr     (deferred)
```

Viewing Memory

In WinDbg, execute this command:

dd nt

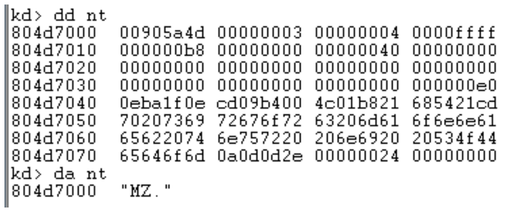
You see the first several bytes of Ntoskrnl.exe, as shown below.

This may be more familiar in ASCII.

In WinDbg, execute this command:

da nt

You see the characters "MZ" --they are at the start of every EXE file.

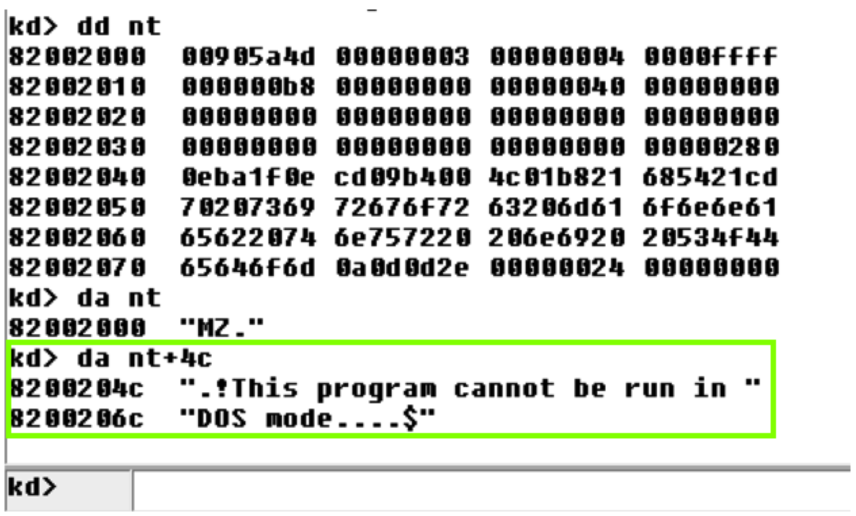


```
kd> dd nt
804d7000  00905a4d 00000003 00000004 0000ffff
804d7010  000000b8 00000000 00000040 00000000
804d7020  00000000 00000000 00000000 00000000
804d7030  00000000 00000000 00000000 000000e0
804d7040  0eba1f0e cd09b400 4c01b821 685421cd
804d7050  70207369 72676f72 63206d61 6f6e6e61
804d7060  65622074 6e757220 206e6920 20534f44
804d7070  65646f6d 0a0d0d2e 00000024 00000000
kd> da nt
804d7000  "MZ."
```

In WinDbg, execute this command:

da nt+4c

You see the message "**This program cannot be run in DOS mode**", as shown below:



```
kd> dd nt
82002000  00905a4d 00000003 00000004 0000ffff
82002010  000000b8 00000000 00000040 00000000
82002020  00000000 00000000 00000000 00000000
82002030  00000000 00000000 00000000 00000280
82002040  0eba1f0e cd09b400 4c01b821 685421cd
82002050  70207369 72676f72 63206d61 6f6e6e61
82002060  65622074 6e757220 206e6920 20534f44
82002070  65646f6d 0a0d0d2e 00000024 00000000
kd> da nt
82002000  "MZ."
kd> da nt+4c
8200204c  ".!This program cannot be run in "
8200206c  "DOS mode....$"

kd>
```

Saving a Screen Image

Make sure you can see the message "**This program cannot be run in DOS mode**", as shown above.

On your keyboard, press the PrntScrn key.

Open Paint and paste the image in.

YOU MUST SUBMIT WHOLE-DESKTOP IMAGES TO GET FULL CREDIT.

Save the image with a filename of "**Proj 13a from YOUR NAME**".

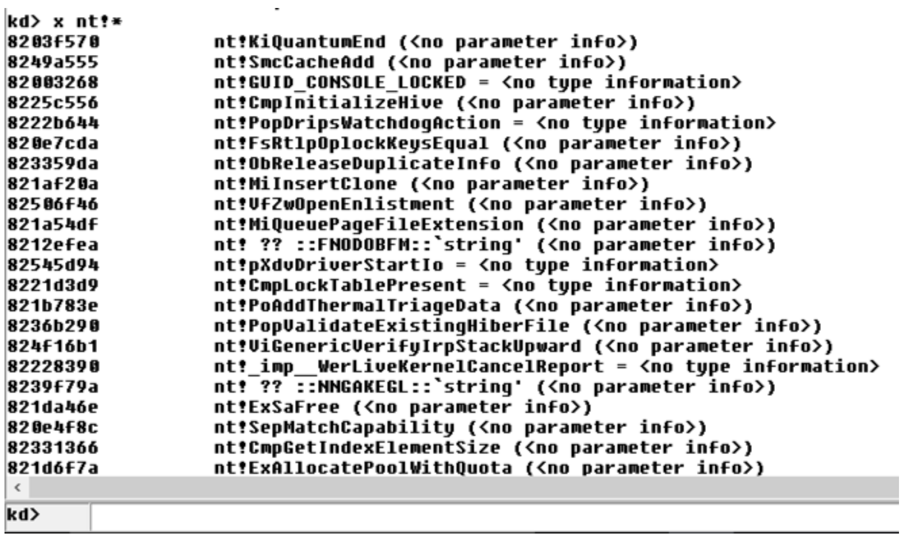
Searching for Functions

In WinDbg, execute this command:

x nt!*

This finds all the functions in Ntoskrnl.

There are a lot of them, as shown below:



```
kd> x nt!*
8203f570          nt!KiQuantumEnd (<no parameter info>)
8249a555          nt!SmcCacheAdd (<no parameter info>)
82083268          nt!GUID_CONSOLE_LOCKED = <no type information>
8225c556          nt!CmpInitializeHive (<no parameter info>)
8222b644          nt!PopDripsWatchdogAction = <no type information>
820e7cda          nt!FsRtlpOplockKeysEqual (<no parameter info>)
823359da          nt!ObReleaseDuplicateInfo (<no parameter info>)
821af20a          nt!MiInsertClone (<no parameter info>)
82506f46          nt!VfZwOpenEnlistment (<no parameter info>)
821a54df          nt!MiQueuePageFileExtension (<no parameter info>)
8212efea          nt! ?? ::FNODOBFM::`string' (<no parameter info>)
82545d94          nt!pXdvDriverStartIo = <no type information>
8221d3d9          nt!CmpLockTablePresent = <no type information>
821b783e          nt!PoAddThermalTriageData (<no parameter info>)
8236b290          nt!PopValidateExistingHiberFile (<no parameter info>)
824f16b1          nt!ViGenericVerifyIrpStackUpward (<no parameter info>)
82228390          nt!_imp__WerLiveKernelCancelReport = <no type information>
8239f79a          nt! ?? ::NNGAKEGL::`string' (<no parameter info>)
821da46e          nt!ExSaFree (<no parameter info>)
820e4f8c          nt!SepMatchCapability (<no parameter info>)
82331366          nt!CmpGetIndexElementSize (<no parameter info>)
821d6f7a          nt!ExAllocatePoolWithQuota (<no parameter info>)
kd>
```

In WinDbg, execute this command:

x nt!*Create*

This finds all the functions in Ntoskrnl that contain the word "Create".

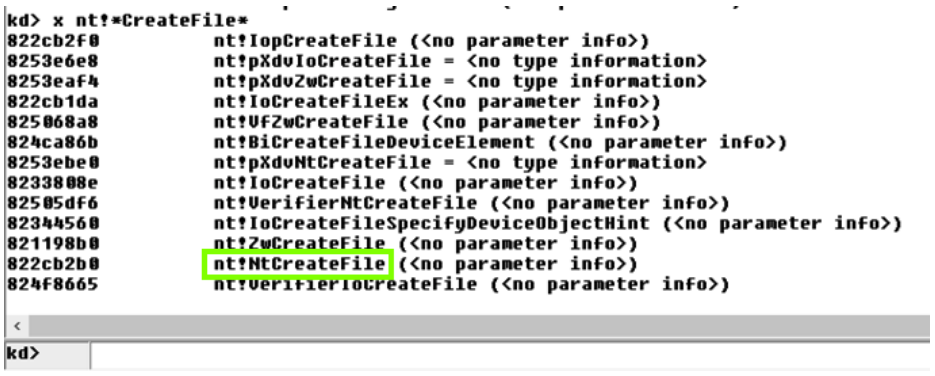
There are a lot of them, too.

In WinDbg, execute this command:

x nt!*CreateFile*

This finds all the functions in Ntoskrnl that contain the word "CreateFile".

There are only about ten of those, including "nt!NtCreateFile", as shown below:



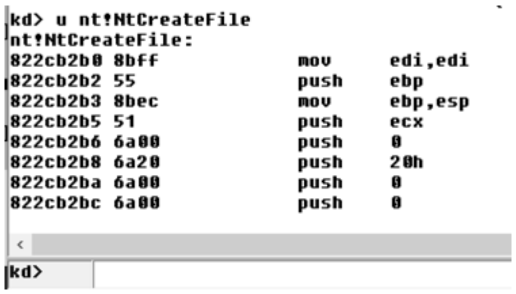
```
kd> x nt!*CreateFile*
822cb2f0          nt!IopCreateFile (<no parameter info>)
8253e6e8          nt!pXdvIoCreateFile = <no type information>
8253eaf4          nt!pXdvZwCreateFile = <no type information>
822cb1da          nt!IoCreateFileEx (<no parameter info>)
825068a8          nt!VfZwCreateFile (<no parameter info>)
824ca86b          nt!BiCreateFileDeviceElement (<no parameter info>)
8253ebe0          nt!pXdvNtCreateFile = <no type information>
8233808e          nt!IoCreateFile (<no parameter info>)
82505df6          nt!VerifierNtCreateFile (<no parameter info>)
82344560          nt!IoCreateFileSpecifyDeviceObjectHint (<no parameter info>)
821198b0          nt!ZwCreateFile (<no parameter info>)
822cb2b0          nt!NtCreateFile (<no parameter info>)
824f8665          nt!VerifierIoCreateFile (<no parameter info>)
kd>
```

Unassembling a Function: Image B

In WinDbg, execute this command:

u nt!NtCreateFile

This shows the first few bytes of the function, disassembled, as shown below:



```
kd> u nt!NtCreateFile
nt!NtCreateFile:
822cb2b0 8bff            mov     edi,edi
822cb2b2 55              push    ebp
822cb2b3 8bec            mov     ebp,esp
822cb2b5 51              push    ecx
822cb2b6 6a00            push    0
822cb2b8 6a20            push    20h
822cb2ba 6a00            push    0
822cb2bc 6a00            push    0
kd>
```

To see more of this function, it helps to use the WinDbg Disassembly window.

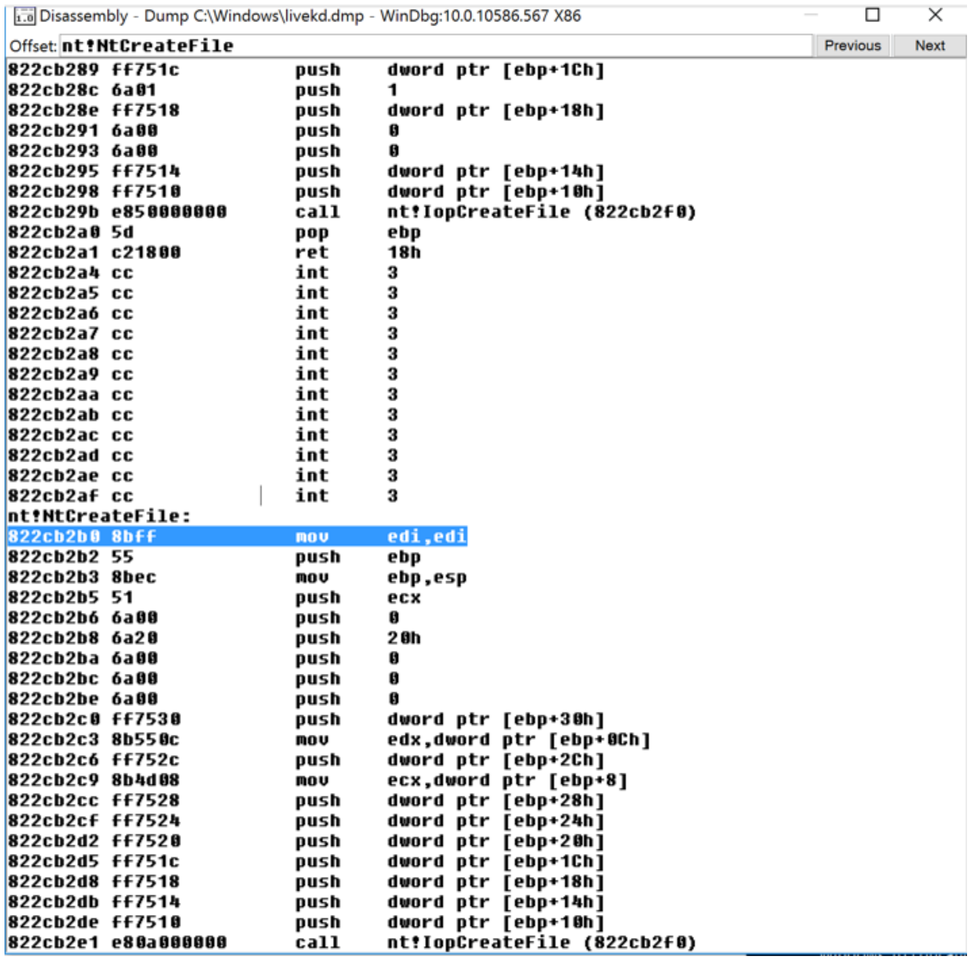
If the Command window is maximized, make it smaller.

From the WinDbg menu bar, click **View**, **Disassembly**.

In the Offset bar at the top, enter

nt!NtCreateFile

This shows the assembly code before and after the start of the NtCreateFile function, as shown below:



Disassembly - Dump C:\Windows\livekd.dmp - WinDbg:10.0.10586.567 X86
Previous
Next

```
Offset: nt!NtCreateFile
822cb289 ff751c          push    dword ptr [ebp+1Ch]
822cb28c 6a01            push    1
822cb28e ff7518          push    dword ptr [ebp+18h]
822cb291 6a00            push    0
822cb293 6a00            push    0
822cb295 ff7514          push    dword ptr [ebp+14h]
822cb298 ff7510          push    dword ptr [ebp+10h]
822cb29b e850000000      call    nt!IopCreateFile (822cb2f0)
822cb2a0 5d              pop     ebp
822cb2a1 c21800          ret     18h
822cb2a4 cc              int     3
822cb2a5 cc              int     3
822cb2a6 cc              int     3
822cb2a7 cc              int     3
822cb2a8 cc              int     3
822cb2a9 cc              int     3
822cb2aa cc              int     3
822cb2ab cc              int     3
822cb2ac cc              int     3
822cb2ad cc              int     3
822cb2ae cc              int     3
822cb2af cc              int     3
nt!NtCreateFile:
822cb2b0 8bff            mov     edi,edi
822cb2b2 55              push    ebp
822cb2b3 8bec            mov     ebp,esp
822cb2b5 51              push    ecx
822cb2b6 6a00            push    0
822cb2b8 6a20            push    20h
822cb2ba 6a00            push    0
822cb2bc 6a00            push    0
822cb2be 6a00            push    0
822cb2c0 ff7530          push    dword ptr [ebp+30h]
822cb2c3 8b550c          mov     edx,dword ptr [ebp+0Ch]
822cb2c6 ff752c          push    dword ptr [ebp+2Ch]
822cb2c9 8b4d08          mov     ecx,dword ptr [ebp+8]
822cb2cc ff7528          push    dword ptr [ebp+28h]
822cb2cf ff7524          push    dword ptr [ebp+24h]
822cb2d2 ff7520          push    dword ptr [ebp+20h]
822cb2d5 ff751c          push    dword ptr [ebp+1Ch]
822cb2d8 ff7518          push    dword ptr [ebp+18h]
822cb2db ff7514          push    dword ptr [ebp+14h]
822cb2de ff7510          push    dword ptr [ebp+10h]
822cb2e1 e80a000000      call    nt!IopCreateFile (822cb2f0)
```

In the Offset bar at the top, enter

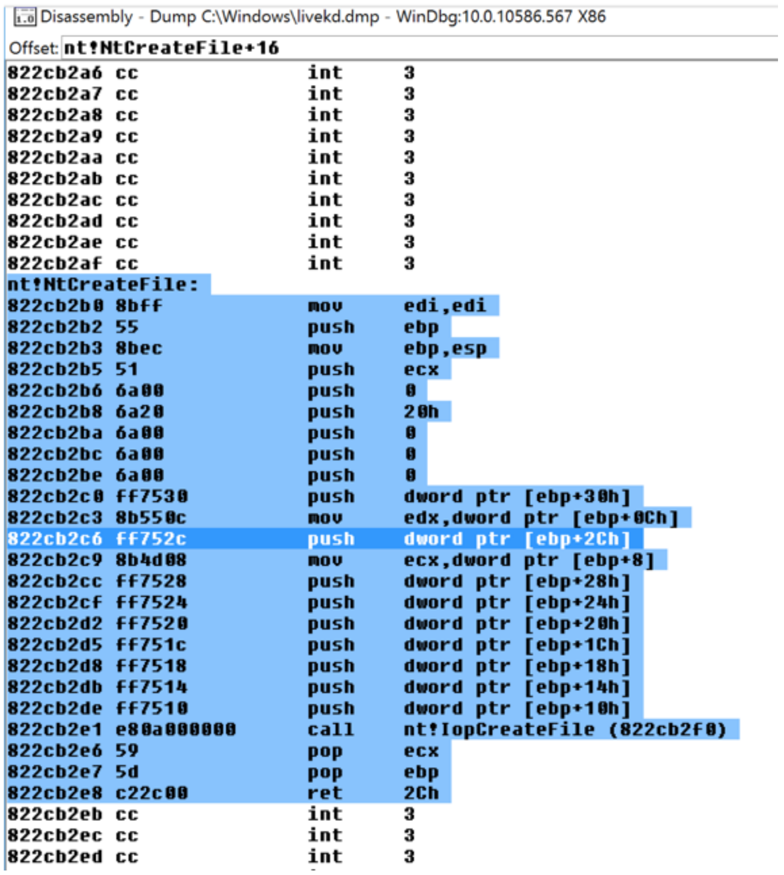
nt!NtCreateFile+16

Resize this window to make the entire function visible. Drag the mouse through it to highlight the entire function, as shown below.

For Windows Server 2008 Users

Use this offset instead:

nt!NtCreateFile+11



Disassembly - Dump C:\Windows\livekd.dmp - WinDbg:10.0.10586.567 X86

Offset: nt!NtCreateFile+16

```
822cb2a6 cc          int     3
822cb2a7 cc          int     3
822cb2a8 cc          int     3
822cb2a9 cc          int     3
822cb2aa cc          int     3
822cb2ab cc          int     3
822cb2ac cc          int     3
822cb2ad cc          int     3
822cb2ae cc          int     3
822cb2af cc          int     3
nt!NtCreateFile:
822cb2b0 8bff        mov     edi,edi
822cb2b2 55          push    ebp
822cb2b3 8bec        mov     ebp,esp
822cb2b5 51          push    ecx
822cb2b6 6a00        push    0
822cb2b8 6a20        push    20h
822cb2ba 6a00        push    0
822cb2bc 6a00        push    0
822cb2be 6a00        push    0
822cb2c0 ff7530      push    dword ptr [ebp+30h]
822cb2c3 8b550c      mov     edx,dword ptr [ebp+0Ch]
822cb2c6 ff752c      push    dword ptr [ebp+2Ch]
822cb2c9 8b4d08      mov     ecx,dword ptr [ebp+8]
822cb2cc ff7528      push    dword ptr [ebp+28h]
822cb2cf ff7524      push    dword ptr [ebp+24h]
822cb2d2 ff7520      push    dword ptr [ebp+20h]
822cb2d5 ff751c      push    dword ptr [ebp+1Ch]
822cb2d8 ff7518      push    dword ptr [ebp+18h]
822cb2db ff7514      push    dword ptr [ebp+14h]
822cb2de ff7510      push    dword ptr [ebp+10h]
822cb2e1 e80a000000  call    nt!IopCreateFile (822cb2f0)
822cb2e6 59          pop     ecx
822cb2e7 5d          pop     ebp
822cb2e8 c22c00      ret     2Ch
822cb2eb cc          int     3
822cb2ec cc          int     3
822cb2ed cc          int     3
```

Saving a Screen Image

Make sure you have highlighted the entire function, as shown above.

On your keyboard, press the PrntScrn key.

Open Paint and paste the image in.

YOU MUST SUBMIT WHOLE-DESKTOP IMAGES TO GET FULL CREDIT.

Save the image with a filename of "**Proj 13b from YOUR NAME**".

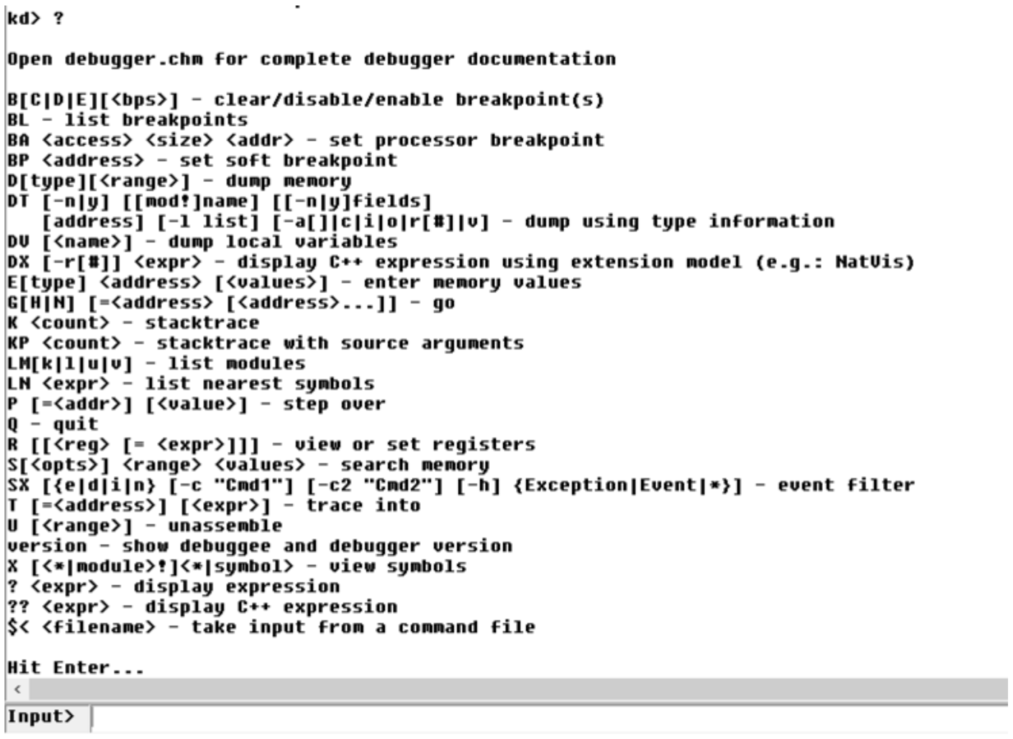
Online Help

Close the Disassembly window.

In WinDbg, execute this command:

?

You see the first page of the online help, as shown below:



```
kd> ?

Open debugger.chm for complete debugger documentation

B[C|D|E][<bps>] - clear/disable/enable breakpoint(s)
BL - list breakpoints
BA <access> <size> <addr> - set processor breakpoint
BP <address> - set soft breakpoint
D[type][<range>] - dump memory
DT [-n|y] [[mod!]name] [[-n|y]fields]
   [address] [-l list] [-a[]|c|i|o|r[#]|v] - dump using type information
DV [<name>] - dump local variables
DX [-r[#]] <expr> - display C++ expression using extension model (e.g.: NatVis)
E[type] <address> [<values>] - enter memory values
G[H|N] [=<address> [<address>...]] - go
K <count> - stacktrace
KP <count> - stacktrace with source arguments
LM[k|l|u|v] - list modules
LN <expr> - list nearest symbols
P [=<addr>] [<value>] - step over
Q - quit
R [[<reg> [= <expr>]]] - view or set registers
S[<opts>] <range> <values> - search memory
SX [{e|d|i|n} [-c "Cmd1"] [-c2 "Cmd2"] [-h] {Exception|Event|*}] - event filter
T [=<address>] [<expr>] - trace into
U [<range>] - unassemble
version - show debuggee and debugger version
X [<*|module>!]<*|symbol> - view symbols
? <expr> - display expression
?? <expr> - display C++ expression
$< <filename> - take input from a command file

Hit Enter...
```

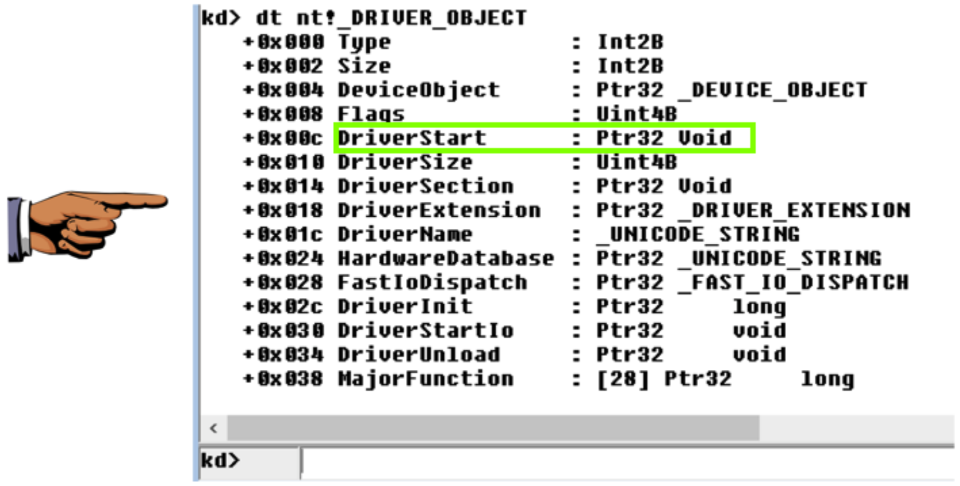
Press Enter to see the other page.

Viewing Type Information for a Structure

In WinDbg, execute this command:

dt nt!_DRIVER_OBJECT

This shows the first few lines of a driver object structure, which stores information about a kernel driver, as shown below. Notice the **DriverStart** pointer--this contains the location of the driver in memory.



```
kd> dt nt!_DRIVER_OBJECT
   +0x000 Type             : Int2B
   +0x002 Size             : Int2B
   +0x004 DeviceObject     : Ptr32 _DEVICE_OBJECT
   +0x008 Flags            : Uint4B
   +0x00c DriverStart      : Ptr32 Void
   +0x010 DriverSize       : Uint4B
   +0x014 DriverSection    : Ptr32 Void
   +0x018 DriverExtension  : Ptr32 _DRIVER_EXTENSION
   +0x01c DriverName       : _UNICODE_STRING
   +0x024 HardwareDatabase : Ptr32 _UNICODE_STRING
   +0x028 FastIoDispatch   : Ptr32 _FAST_IO_DISPATCH
   +0x02c DriverInit       : Ptr32     long
   +0x030 DriverStartIo    : Ptr32     void
   +0x034 DriverUnload     : Ptr32     void
   +0x038 MajorFunction    : [28] Ptr32     long
```

Saving a Screen Image

Make sure the **DriverStart** pointer is visible, as shown above.

On your keyboard, press the PrntScrn key.

Open Paint and paste the image in.

YOU MUST SUBMIT WHOLE-DESKTOP IMAGES TO GET FULL CREDIT.

Save the image with a filename of "**Proj 13c from YOUR NAME**".

Turning in Your Project

Email the images to: **cnit.126sam@gmail.com** with a subject line of **Proj 13 From Your Name**, replacing Your Name with your own first and last name. Send a Cc to yourself.

Posted 4-19-17 by Sam Bowne