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

What You Need

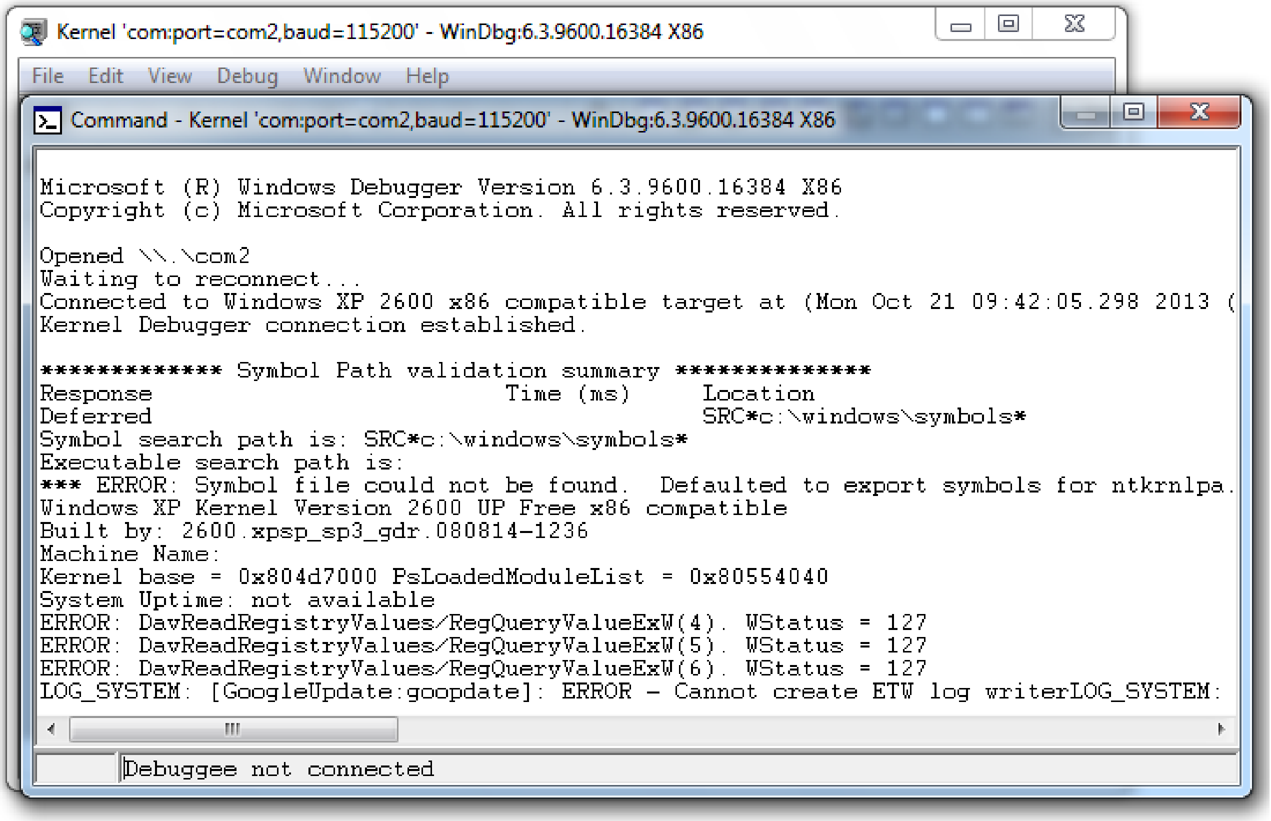
- A WINDBG machine with WinDbg installed, connected to a TARGET Windows XPSP3 virtual machine, as prepared in the previous project.

Purpose

Practice using simple WinDbg commands.

Starting Configuration

You should have a Windows XP SP3 TARGET machine running in debug mode, connected to a Windows WINDBG machine running WinDbg, showing the message "**Connected to Windows XP**, as shown below:


Kernel 'com:port=com2,baud=115200' - WinDbg:6.3.9600.16384 X86
File Edit View Debug Window Help
Command - Kernel 'com:port=com2,baud=115200' - WinDbg:6.3.9600.16384 X86

```
Microsoft (R) Windows Debugger Version 6.3.9600.16384 X86
Copyright (c) Microsoft Corporation. All rights reserved.

Opened \\.\com2
Waiting to reconnect...
Connected to Windows XP 2600 x86 compatible target at (Mon Oct 21 09:42:05.298 2013 (
Kernel Debugger connection established.

************** Symbol Path validation summary ***************
Response                         Time (ms)     Location
Deferred                                       SRC*c:\windows\symbols*
Symbol search path is: SRC*c:\windows\symbols*
Executable search path is:
*** ERROR: Symbol file could not be found.  Defaulted to export symbols for ntkrnlpa.
Windows XP Kernel Version 2600 UP Free x86 compatible
Built by: 2600.xpsp_sp3_gdr.080814-1236
Machine Name:
Kernel base = 0x804d7000 PsLoadedModuleList = 0x80554040
System Uptime: not available
ERROR: DavReadRegistryValues/RegQueryValueExW(4). WStatus = 127
ERROR: DavReadRegistryValues/RegQueryValueExW(5). WStatus = 127
ERROR: DavReadRegistryValues/RegQueryValueExW(6). WStatus = 127
LOG_SYSTEM: [GoogleUpdate:goopdate]: ERROR - Cannot create ETW log writerLOG_SYSTEM:
```

```
Debuggee not connected
```

Notice the status bar at the bottom of the WinDbg window, saying "Debugee not connected".

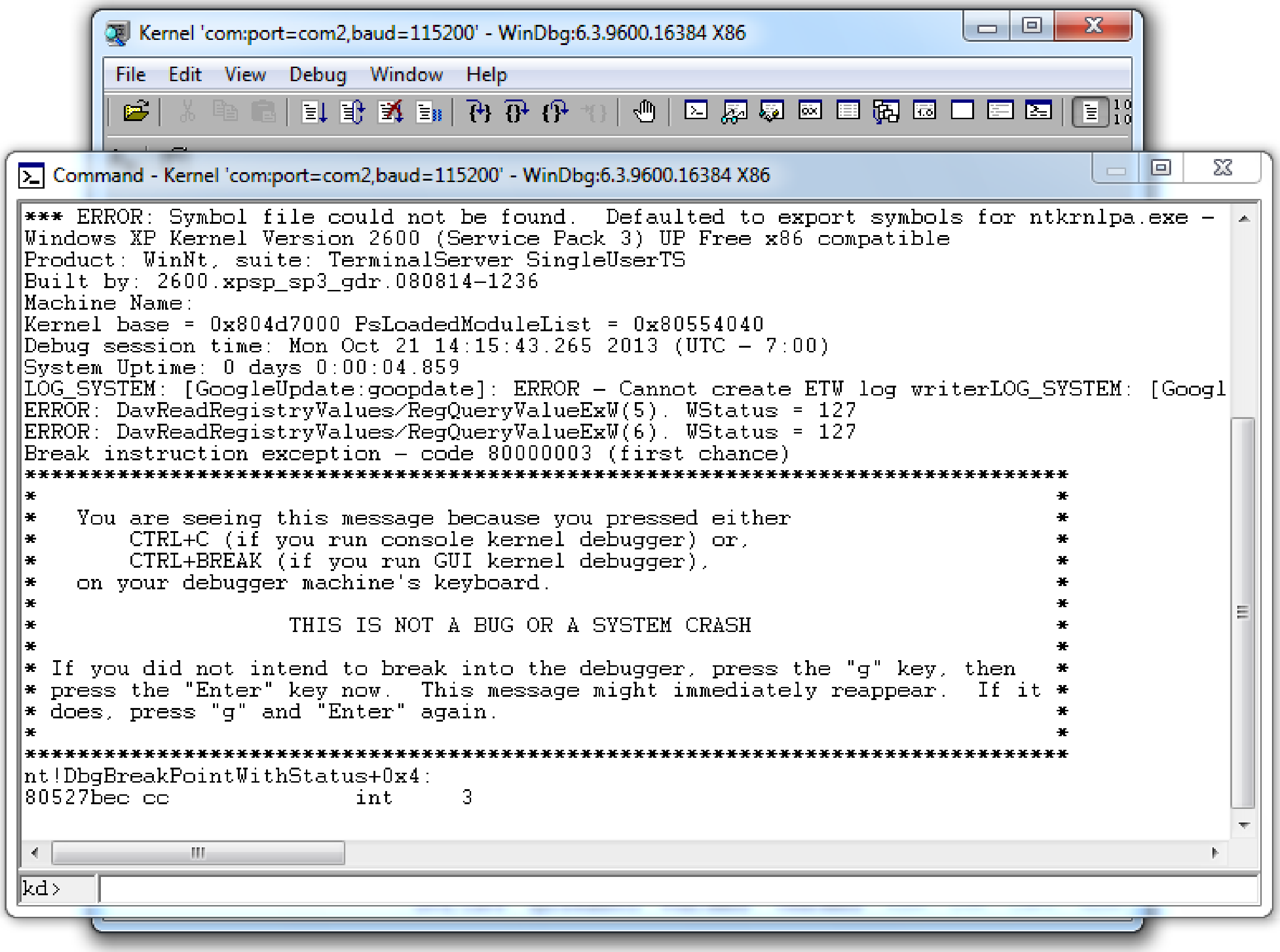
That means that even though the serial connection is working, the kernel on the target machine has not been interrupted by WinDbg yet.

To start executing WinDbg commands, you need to break in to the kernel.

Breaking In

From the WinDbg menu bar, click **Debug**, **Break**.

A message appears, saying you have broken in, ending with an "int 3" instruction, showing that you have hit a breakpoint, as shown below:


Kernel 'com:port=com2,baud=115200' - WinDbg:6.3.9600.16384 X86
File Edit View Debug Window Help
Command - Kernel 'com:port=com2,baud=115200' - WinDbg:6.3.9600.16384 X86
kd>

```
*** ERROR: Symbol file could not be found.  Defaulted to export symbols for ntkrnlpa.exe -
Windows XP Kernel Version 2600 (Service Pack 3) UP Free x86 compatible
Product: WinNt, suite: TerminalServer SingleUserTS
Built by: 2600.xpsp_sp3_gdr.080814-1236
Machine Name:
Kernel base = 0x804d7000 PsLoadedModuleList = 0x80554040
Debug session time: Mon Oct 21 14:15:43.265 2013 (UTC - 7:00)
System Uptime: 0 days 0:00:04.859
LOG_SYSTEM: [GoogleUpdate:goopdate]: ERROR - Cannot create ETW log writerLOG_SYSTEM: [Googl
ERROR: DavReadRegistryValues/RegQueryValueExW(5). WStatus = 127
ERROR: DavReadRegistryValues/RegQueryValueExW(6). WStatus = 127
Break instruction exception - code 80000003 (first chance)
*******************************************************************************
*                                                                             *
*   You are seeing this message because you pressed either                    *
*       CTRL+C (if you run console kernel debugger) or,                       *
*       CTRL+BREAK (if you run GUI kernel debugger),                          *
*   on your debugger machine's keyboard.                                      *
*                                                                             *
*                   THIS IS NOT A BUG OR A SYSTEM CRASH                       *
*                                                                             *
* If you did not intend to break into the debugger, press the "g" key, then   *
* press the "Enter" key now.  This message might immediately reappear.  If it *
* does, press "g" and "Enter" again.                                          *
*                                                                             *
*******************************************************************************
nt!DbgBreakPointWithStatus+0x4:
80527bec cc              int     3
```

Press **Enter** to get a **kd>** prompt, as shown above.

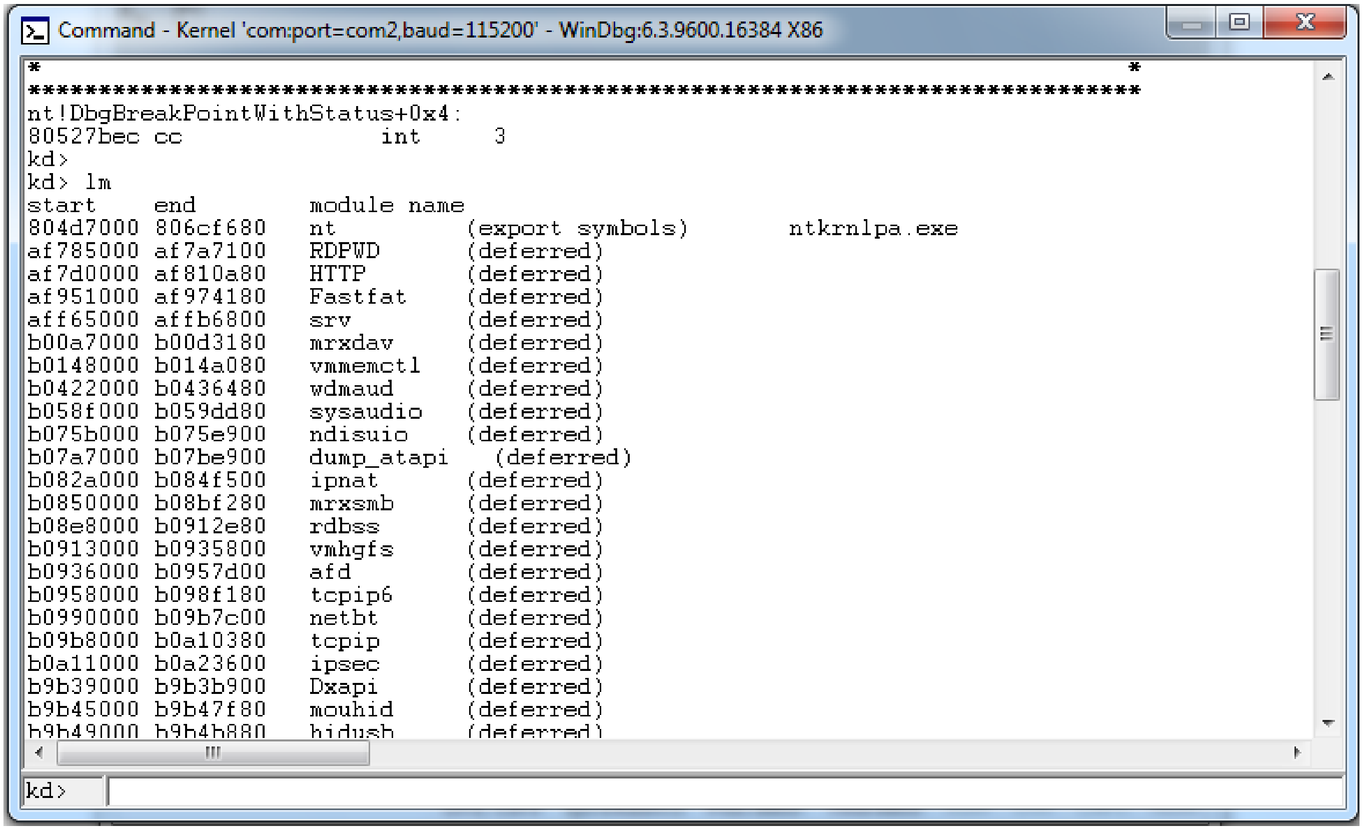
Listing Modules with lm

With the focus on WinDbg, type

lm

and then press the Enter key.

The characters you type appear in the status bar, at the bottom of the window, but when you press Enter they move into the main window and show the output, as shown below:


Command - Kernel 'com:port=com2,baud=115200' - WinDbg:6.3.9600.16384 X86

```
*                                                                             *
*******************************************************************************
nt!DbgBreakPointWithStatus+0x4:
80527bec cc              int     3
kd>
kd> lm
start    end        module name
804d7000 806cf680   nt         (export symbols)       ntkrnlpa.exe
af785000 af7a7100   RDPWD      (deferred)
af7d0000 af810a80   HTTP       (deferred)
af951000 af974180   Fastfat    (deferred)
aff65000 affb6800   srv        (deferred)
b00a7000 b00d3180   mrxdav     (deferred)
b0148000 b014a080   vmmemctl   (deferred)
b0422000 b0436480   wdmaud     (deferred)
b058f000 b059dd80   sysaudio   (deferred)
b075b000 b075e900   ndisuio    (deferred)
b07a7000 b07be900   dump_atapi   (deferred)
b082a000 b084f500   ipnat      (deferred)
b0850000 b08bf280   mrxsmb     (deferred)
b08e8000 b0912e80   rdbss      (deferred)
b0913000 b0935800   vmhgfs     (deferred)
b0936000 b0957d00   afd        (deferred)
b0958000 b098f180   tcpip6     (deferred)
b0990000 b09b7c00   netbt      (deferred)
b09b8000 b0a10380   tcpip      (deferred)
b0a11000 b0a23600   ipsec      (deferred)
b9b39000 b9b3b900   Dxapi      (deferred)
b9b45000 b9b47f80   mouhid     (deferred)
b9b49000 b9b4b880   hidusb     (deferred)
```

```
kd>
```

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

You should see the module named **nt** at the top, as shown above.

This is Ntoskrnl, the main kernel module.

Viewing Memory

In WinDbg, execute this command:

dd nt

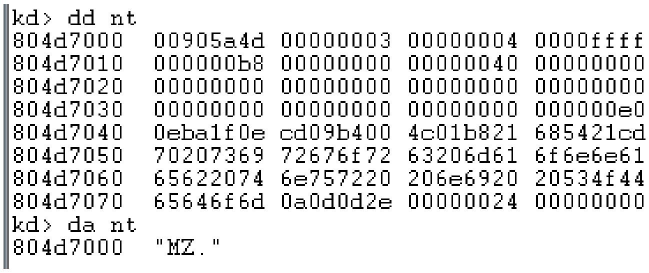
You see the first several bytes of Ntoskrnl.exe, as shown below.

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.

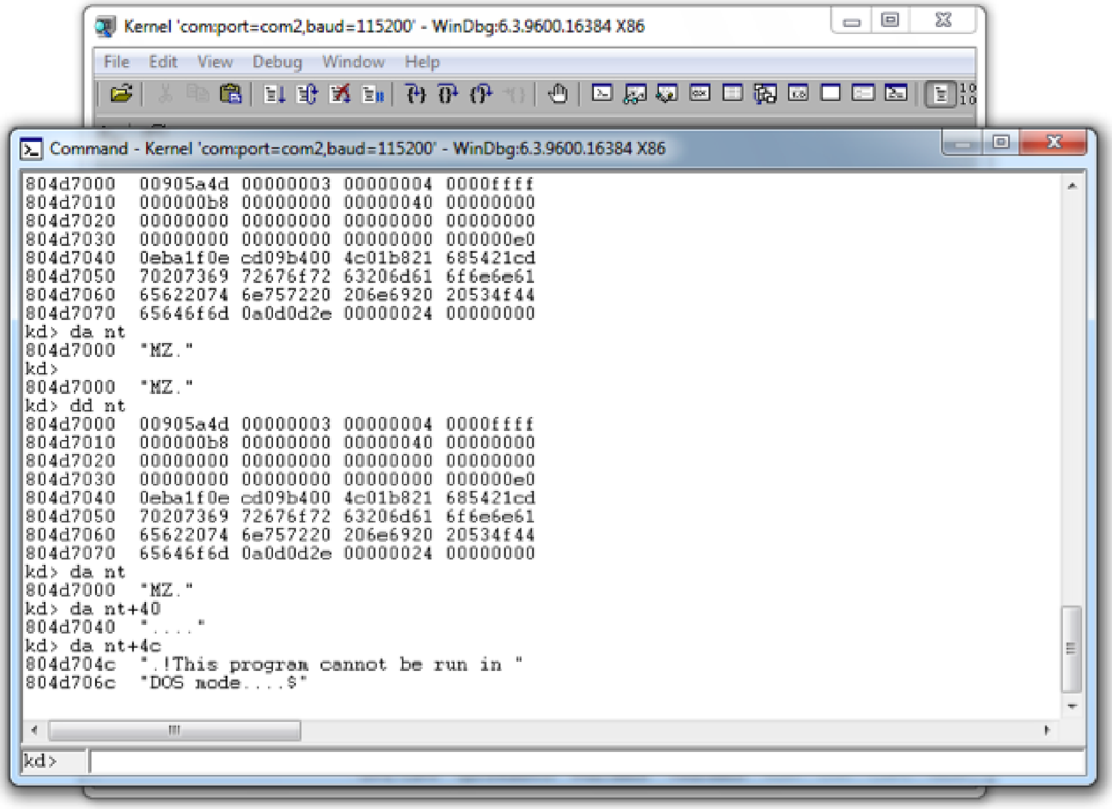


```
kd> dd nt
804d7000  00905a4d 00000003 00000004 0000ffff
804d7010  000000b8 00000000 00000040 00000000
804d7020  00000000 00000000 00000000 00000000
804d7030  00000000 00000000 00000000 000000e0
804d7040  0eba1f0e cd09b400 4c01b821 685421cd
804d7050  70207369 72676f72 63206d61 6f6e6e61
804d7060  65622074 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:



Kernel 'com:port=com2,baud=115200' - WinDbg:6.3.9600.16384 X86
File Edit View Debug Window Help
Command - Kernel 'com:port=com2,baud=115200' - WinDbg:6.3.9600.16384 X86

```
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."
kd>
804d7000  "MZ."
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."
kd> da nt+40
804d7040  "...."
kd> da nt+4c
804d704c  ".!This program cannot be run in "
804d706c  "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.

Click **Start**, type in **PAINT**, and open Paint.

Press **Ctrl+V** to paste in the image of your desktop.

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

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

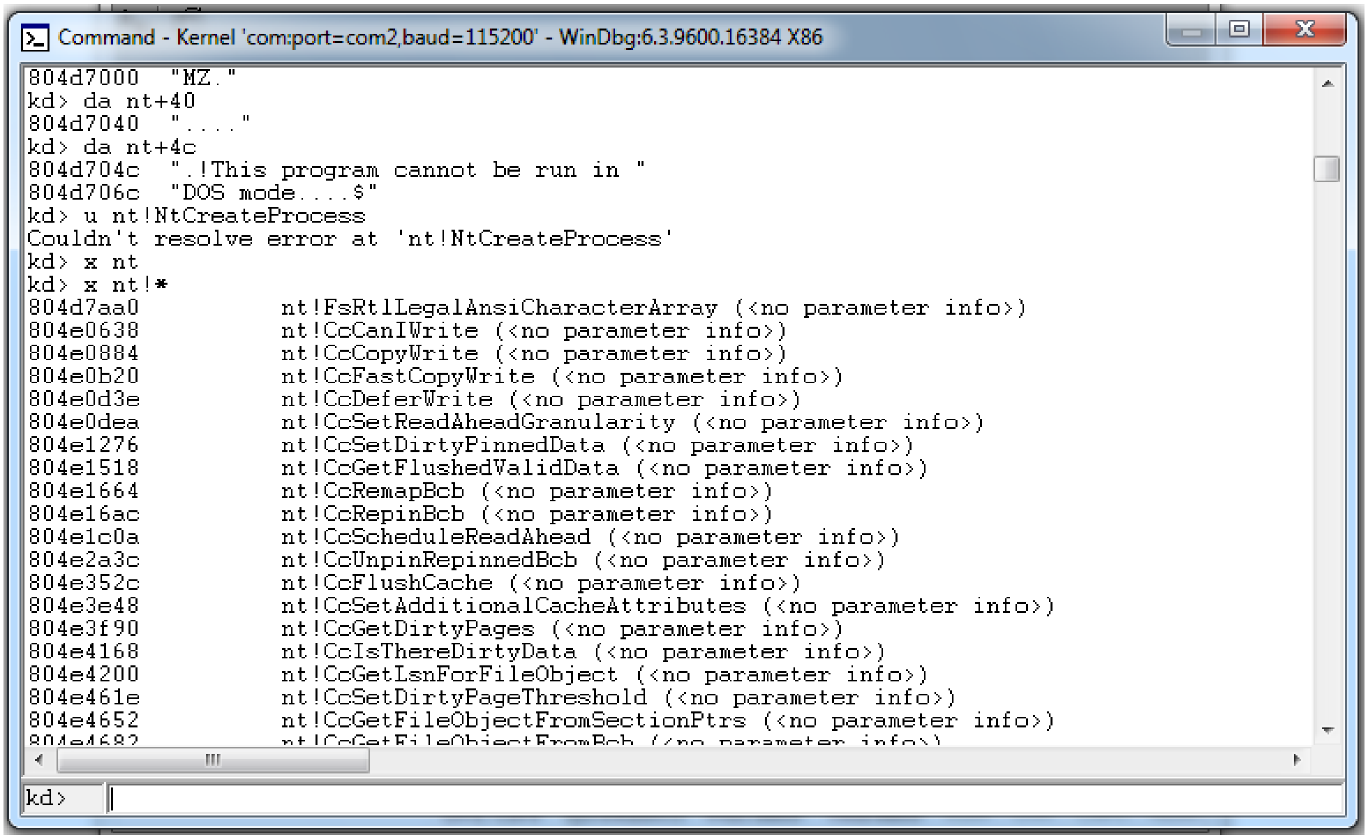
Searching for Functions

In WinDbg, execute this command:

x nt!*

This finds all the functions in Ntoskrnl.

There are a lot of them, as shown below:


Command - Kernel 'com:port=com2,baud=115200' - WinDbg:6.3.9600.16384 X86

```
804d7000  "MZ."
kd> da nt+40
804d7040  "...."
kd> da nt+4c
804d704c  ".!This program cannot be run in "
804d706c  "DOS mode....$"
kd> u nt!NtCreateProcess
Couldn't resolve error at 'nt!NtCreateProcess'
kd> x nt
kd> x nt!*
804d7aa0          nt!FsRtlLegalAnsiCharacterArray (<no parameter info>)
804e0638          nt!CcCanIWrite (<no parameter info>)
804e0884          nt!CcCopyWrite (<no parameter info>)
804e0b20          nt!CcFastCopyWrite (<no parameter info>)
804e0d3e          nt!CcDeferWrite (<no parameter info>)
804e0dea          nt!CcSetReadAheadGranularity (<no parameter info>)
804e1276          nt!CcSetDirtyPinnedData (<no parameter info>)
804e1518          nt!CcGetFlushedValidData (<no parameter info>)
804e1664          nt!CcRemapBcb (<no parameter info>)
804e16ac          nt!CcRepinBcb (<no parameter info>)
804e1c0a          nt!CcScheduleReadAhead (<no parameter info>)
804e2a3c          nt!CcUnpinRepinnedBcb (<no parameter info>)
804e352c          nt!CcFlushCache (<no parameter info>)
804e3e48          nt!CcSetAdditionalCacheAttributes (<no parameter info>)
804e3f90          nt!CcGetDirtyPages (<no parameter info>)
804e4168          nt!CcIsThereDirtyData (<no parameter info>)
804e4200          nt!CcGetLsnForFileObject (<no parameter info>)
804e461e          nt!CcSetDirtyPageThreshold (<no parameter info>)
804e4652          nt!CcGetFileObjectFromSectionPtrs (<no parameter info>)
804e4682          nt!CcGetFileObjectFromBcb (<no parameter info>)
```

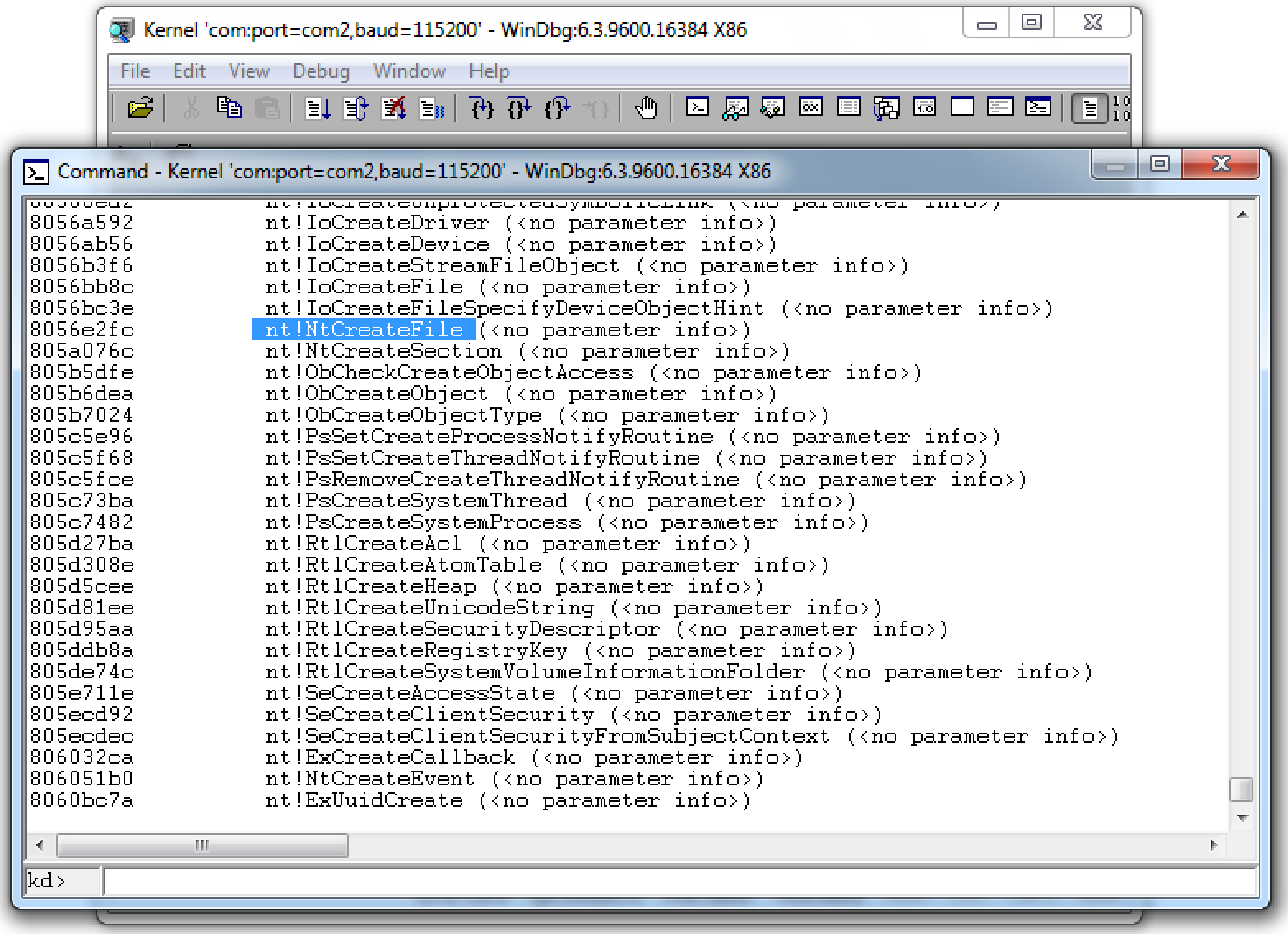
```
kd>
```

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, including "nt!NtCreateFile", as highlighted below:


Kernel 'com:port=com2,baud=115200' - WinDbg:6.3.9600.16384 X86
File Edit View Debug Window Help
Command - Kernel 'com:port=com2,baud=115200' - WinDbg:6.3.9600.16384 X86

```
8056a592           nt!IoCreateDriver (<no parameter info>)
8056ab56           nt!IoCreateDevice (<no parameter info>)
8056b3f6           nt!IoCreateStreamFileObject (<no parameter info>)
8056bb8c           nt!IoCreateFile (<no parameter info>)
8056bc3e           nt!IoCreateFileSpecifyDeviceObjectHint (<no parameter info>)
8056e2fc           nt!NtCreateFile (<no parameter info>)
805a076c           nt!NtCreateSection (<no parameter info>)
805b5dfe           nt!ObCheckCreateObjectAccess (<no parameter info>)
805b6dea           nt!ObCreateObject (<no parameter info>)
805b7024           nt!ObCreateObjectType (<no parameter info>)
805c5e96           nt!PsSetCreateProcessNotifyRoutine (<no parameter info>)
805c5f68           nt!PsSetCreateThreadNotifyRoutine (<no parameter info>)
805c5fce           nt!PsRemoveCreateThreadNotifyRoutine (<no parameter info>)
805c73ba           nt!PsCreateSystemThread (<no parameter info>)
805c7482           nt!PsCreateSystemProcess (<no parameter info>)
805d27ba           nt!RtlCreateAcl (<no parameter info>)
805d308e           nt!RtlCreateAtomTable (<no parameter info>)
805d5cee           nt!RtlCreateHeap (<no parameter info>)
805d81ee           nt!RtlCreateUnicodeString (<no parameter info>)
805d95aa           nt!RtlCreateSecurityDescriptor (<no parameter info>)
805ddb8a           nt!RtlCreateRegistryKey (<no parameter info>)
805de74c           nt!RtlCreateSystemVolumeInformationFolder (<no parameter info>)
805e711e           nt!SeCreateAccessState (<no parameter info>)
805ecd92           nt!SeCreateClientSecurity (<no parameter info>)
805ecdec           nt!SeCreateClientSecurityFromSubjectContext (<no parameter info>)
806032ca           nt!ExCreateCallback (<no parameter info>)
806051b0           nt!NtCreateEvent (<no parameter info>)
8060bc7a           nt!ExUuidCreate (<no parameter info>)
kd>
```

Unassembling a Function

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:
8056e2fc 8bff             mov     edi,edi
8056e2fe 55               push    ebp
8056e2ff 8bec             mov     ebp,esp
8056e301 33c0             xor     eax,eax
8056e303 50               push    eax
8056e304 50               push    eax
8056e305 50               push    eax
8056e306 ff7530           push    dword ptr [ebp+30h]
```

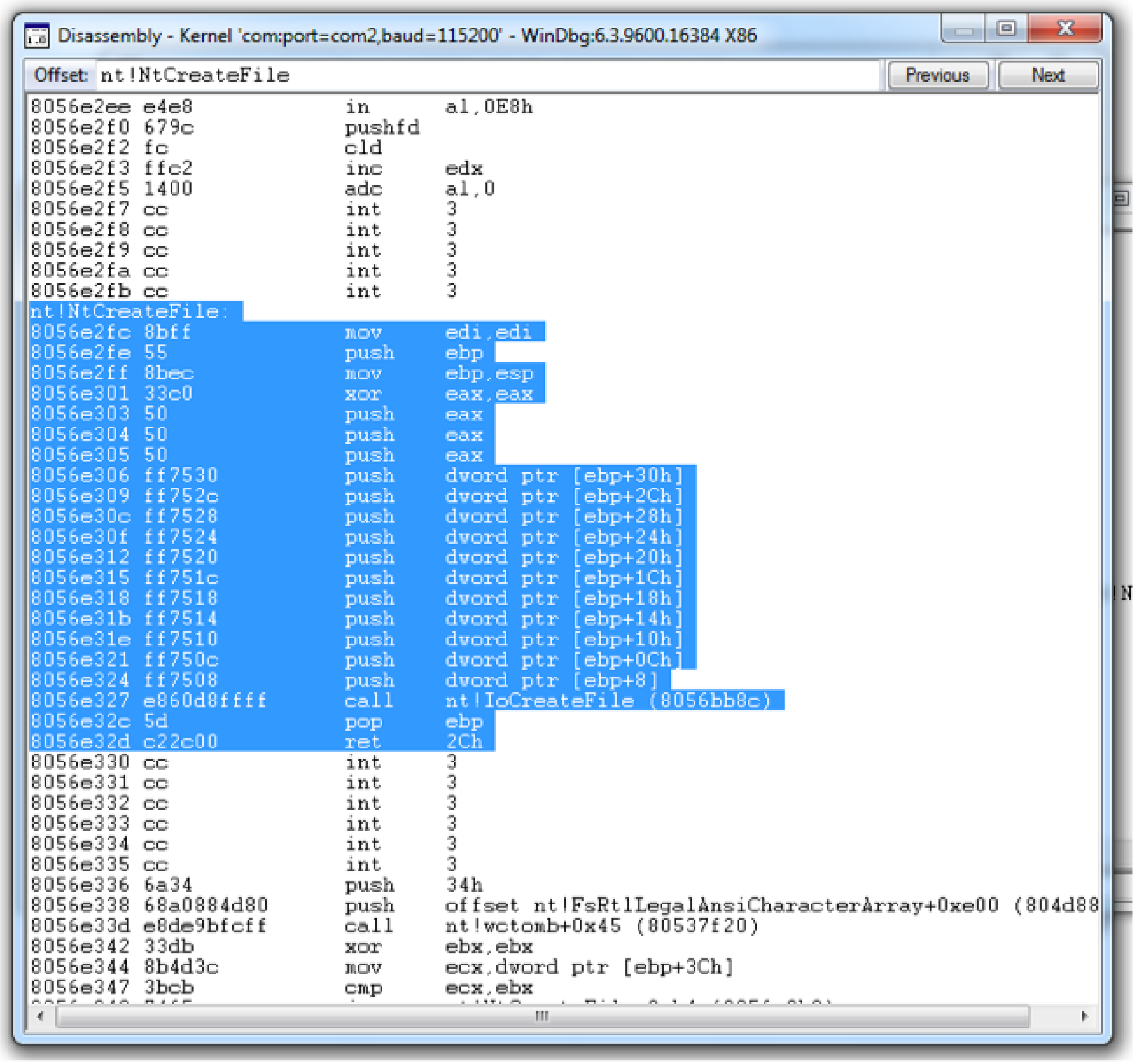
To see more of this function, it helps to use the WinDbg Disassembly window.

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

In the Offset bar at the top, enter

nt!NtCreateFile

Resize this window to make the entire function visible, as highlighted below:



Disassembly - Kernel 'com:port=com2,baud=115200' - WinDbg:6.3.9600.16384 X86
Offset: nt!NtCreateFile
Previous
Next

```
8056e2ee e4e8            in      al,0E8h
8056e2f0 679c            pushfd
8056e2f2 fc              cld
8056e2f3 ffc2            inc     edx
8056e2f5 1400            adc     al,0
8056e2f7 cc              int     3
8056e2f8 cc              int     3
8056e2f9 cc              int     3
8056e2fa cc              int     3
8056e2fb cc              int     3
nt!NtCreateFile:
8056e2fc 8bff            mov     edi,edi
8056e2fe 55              push    ebp
8056e2ff 8bec            mov     ebp,esp
8056e301 33c0            xor     eax,eax
8056e303 50              push    eax
8056e304 50              push    eax
8056e305 50              push    eax
8056e306 ff7530          push    dword ptr [ebp+30h]
8056e309 ff752c          push    dword ptr [ebp+2Ch]
8056e30c ff7528          push    dword ptr [ebp+28h]
8056e30f ff7524          push    dword ptr [ebp+24h]
8056e312 ff7520          push    dword ptr [ebp+20h]
8056e315 ff751c          push    dword ptr [ebp+1Ch]
8056e318 ff7518          push    dword ptr [ebp+18h]
8056e31b ff7514          push    dword ptr [ebp+14h]
8056e31e ff7510          push    dword ptr [ebp+10h]
8056e321 ff750c          push    dword ptr [ebp+0Ch]
8056e324 ff7508          push    dword ptr [ebp+8]
8056e327 e860d8ffff      call    nt!IoCreateFile (8056bb8c)
8056e32c 5d              pop     ebp
8056e32d c22c00          ret     2Ch
8056e330 cc              int     3
8056e331 cc              int     3
8056e332 cc              int     3
8056e333 cc              int     3
8056e334 cc              int     3
8056e335 cc              int     3
8056e336 6a34            push    34h
8056e338 68a0884d80      push    offset nt!FsRtlLegalAnsiCharacterArray+0xe00 (804d88
8056e33d e8de9bfcff      call    nt!wctomb+0x45 (80537f20)
8056e342 33db            xor     ebx,ebx
8056e344 8b4d3c          mov     ecx,dword ptr [ebp+3Ch]
8056e347 3bcb            cmp     ecx,ebx
```

Saving a Screen Image

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

On your keyboard, press the PrntScrn key.

Click **Start**, type in **PAINT**, and open Paint.

Press **Ctrl+V** to paste in the image of your desktop.

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

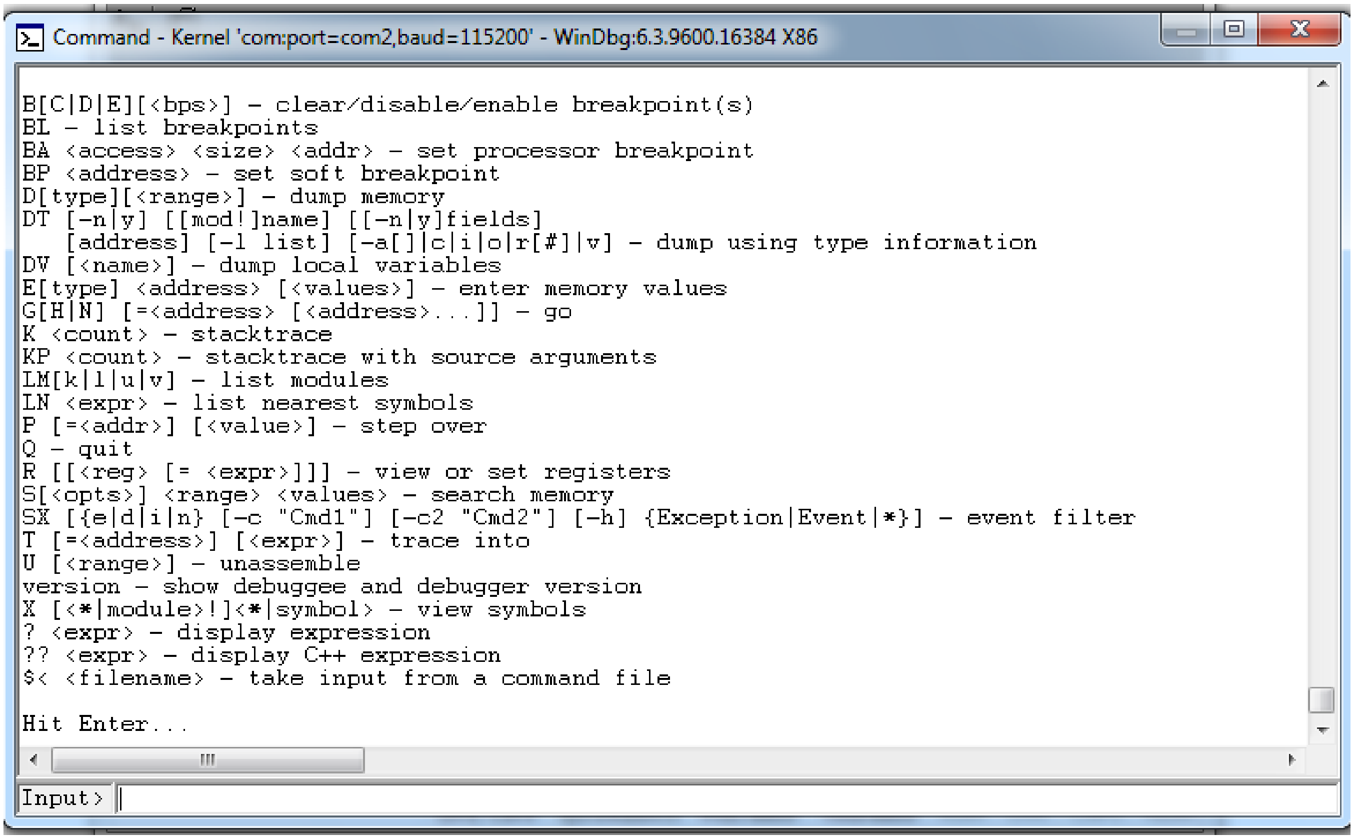
Save the image with a filename of "**Proj 13b from YOUR NAME**".

Online Help

In WinDbg, execute this command:

?

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


Command - Kernel 'com:port=com2,baud=115200' - WinDbg:6.3.9600.16384 X86
Input>

```
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
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...
```

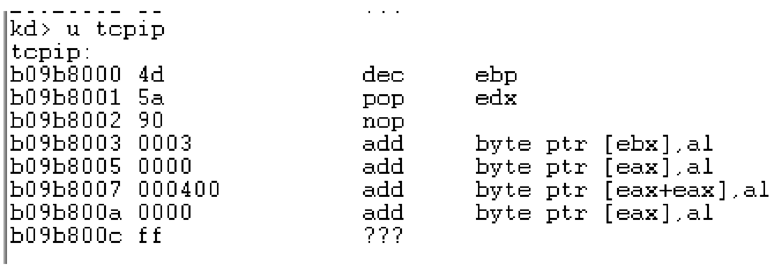
Press Enter to see the other page.

Examining the tcpip Module

In WinDbg, execute this command:

u tcpip

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



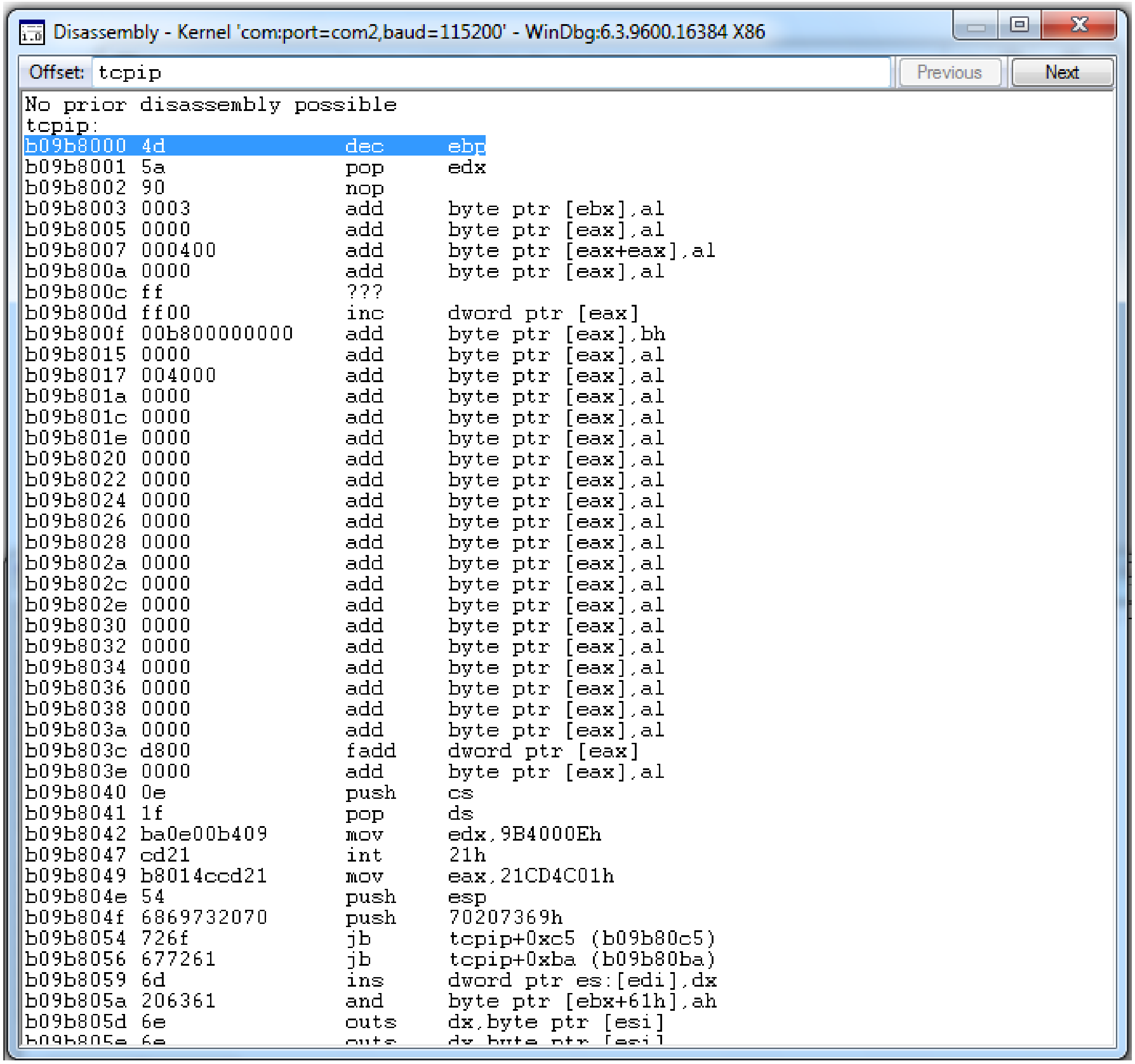
```
kd> u tcpip
tcpip:
b09b8000 4d              dec     ebp
b09b8001 5a              pop     edx
b09b8002 90              nop
b09b8003 0003            add     byte ptr [ebx],al
b09b8005 0000            add     byte ptr [eax],al
b09b8007 000400          add     byte ptr [eax+eax],al
b09b800a 0000            add     byte ptr [eax],al
b09b800c ff              ???
```

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

In the Offset bar, enter

tcpip

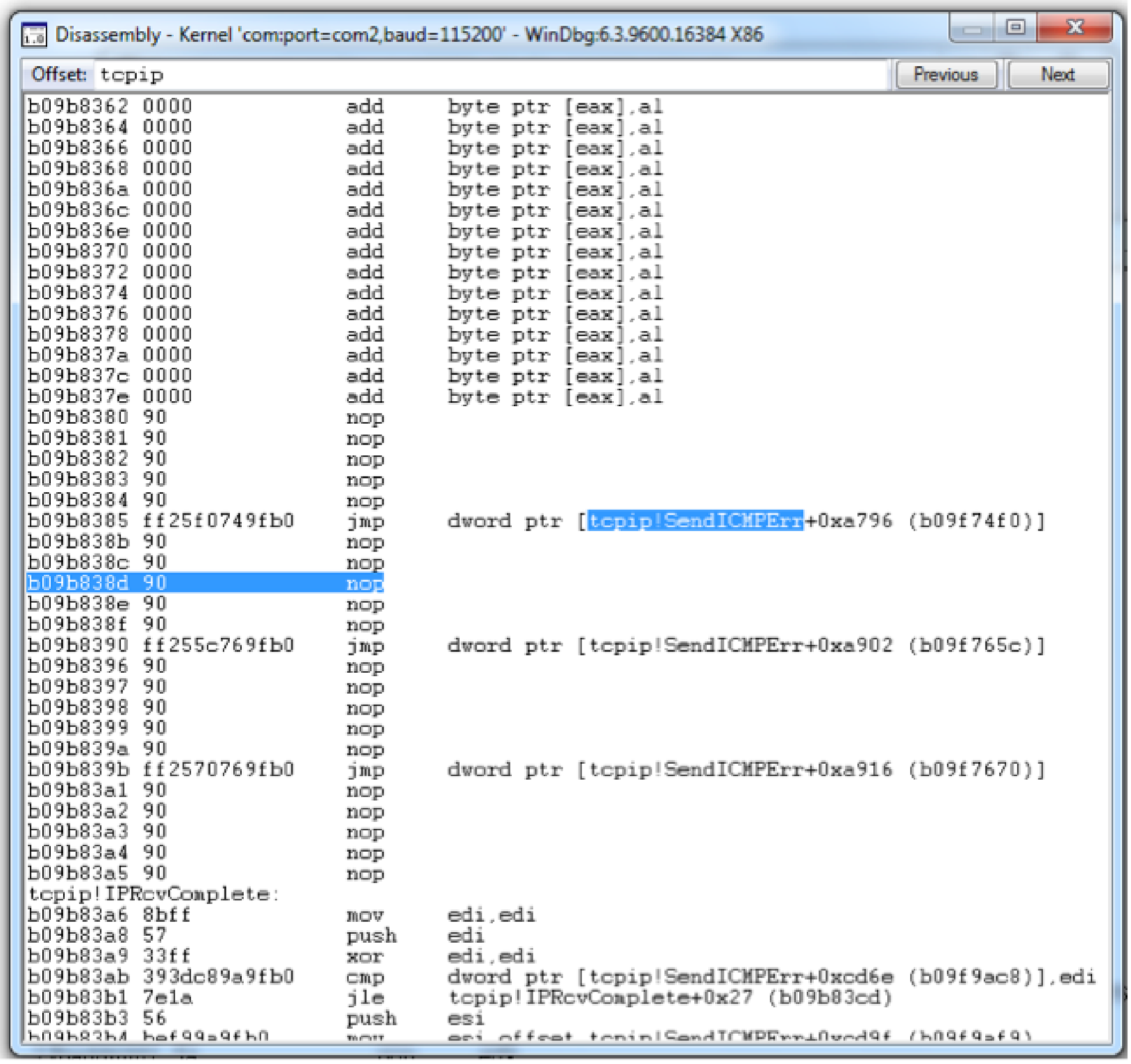
You should see the first portion of the tcpip module, as shown below (you may have to wait a few seconds for it to appear, or even close and re-open the Disassembly window):


Disassembly - Kernel 'com:port=com2,baud=115200' - WinDbg:6.3.9600.16384 X86
Previous
Next

Offset: tcpip

```
No prior disassembly possible
tcpip:
b09b8000 4d              dec     ebp
b09b8001 5a              pop     edx
b09b8002 90              nop
b09b8003 0003            add     byte ptr [ebx],al
b09b8005 0000            add     byte ptr [eax],al
b09b8007 000400          add     byte ptr [eax+eax],al
b09b800a 0000            add     byte ptr [eax],al
b09b800c ff              ???
b09b800d ff00            inc     dword ptr [eax]
b09b800f 00b800000000    add     byte ptr [eax],bh
b09b8015 0000            add     byte ptr [eax],al
b09b8017 004000          add     byte ptr [eax],al
b09b801a 0000            add     byte ptr [eax],al
b09b801c 0000            add     byte ptr [eax],al
b09b801e 0000            add     byte ptr [eax],al
b09b8020 0000            add     byte ptr [eax],al
b09b8022 0000            add     byte ptr [eax],al
b09b8024 0000            add     byte ptr [eax],al
b09b8026 0000            add     byte ptr [eax],al
b09b8028 0000            add     byte ptr [eax],al
b09b802a 0000            add     byte ptr [eax],al
b09b802c 0000            add     byte ptr [eax],al
b09b802e 0000            add     byte ptr [eax],al
b09b8030 0000            add     byte ptr [eax],al
b09b8032 0000            add     byte ptr [eax],al
b09b8034 0000            add     byte ptr [eax],al
b09b8036 0000            add     byte ptr [eax],al
b09b8038 0000            add     byte ptr [eax],al
b09b803a 0000            add     byte ptr [eax],al
b09b803c d800            fadd    dword ptr [eax]
b09b803e 0000            add     byte ptr [eax],al
b09b8040 0e              push    cs
b09b8041 1f              pop     ds
b09b8042 ba0e00b409      mov     edx,9B4000Eh
b09b8047 cd21            int     21h
b09b8049 b8014ccd21      mov     eax,21CD4C01h
b09b804e 54              push    esp
b09b804f 6869732070      push    70207369h
b09b8054 726f            jb      tcpip+0xc5 (b09b80c5)
b09b8056 677261          jb      tcpip+0xba (b09b80ba)
b09b8059 6d              ins     dword ptr es:[edi],dx
b09b805a 206361          and     byte ptr [ebx+61h],ah
b09b805d 6e              outs    dx,byte ptr [esi]
b09b805e 6e              outs    dx,byte ptr [esi]
```

Press the PageDown key about 14 times, depending on the size of your window, until you find a reference to **tcpip!SendICMPErr**, as highlighted below:



Disassembly - Kernel 'com:port=com2,baud=115200' - WinDbg:6.3.9600.16384 X86
Previous
Next

Offset: tcpip

```
b09b8362 0000            add     byte ptr [eax],al
b09b8364 0000            add     byte ptr [eax],al
b09b8366 0000            add     byte ptr [eax],al
b09b8368 0000            add     byte ptr [eax],al
b09b836a 0000            add     byte ptr [eax],al
b09b836c 0000            add     byte ptr [eax],al
b09b836e 0000            add     byte ptr [eax],al
b09b8370 0000            add     byte ptr [eax],al
b09b8372 0000            add     byte ptr [eax],al
b09b8374 0000            add     byte ptr [eax],al
b09b8376 0000            add     byte ptr [eax],al
b09b8378 0000            add     byte ptr [eax],al
b09b837a 0000            add     byte ptr [eax],al
b09b837c 0000            add     byte ptr [eax],al
b09b837e 0000            add     byte ptr [eax],al
b09b8380 90              nop
b09b8381 90              nop
b09b8382 90              nop
b09b8383 90              nop
b09b8384 90              nop
b09b8385 ff25f0749fb0    jmp     dword ptr [tcpip!SendICMPErr+0xa796 (b09f74f0)]
b09b838b 90              nop
b09b838c 90              nop
b09b838d 90              nop
b09b838e 90              nop
b09b838f 90              nop
b09b8390 ff255c769fb0    jmp     dword ptr [tcpip!SendICMPErr+0xa902 (b09f765c)]
b09b8396 90              nop
b09b8397 90              nop
b09b8398 90              nop
b09b8399 90              nop
b09b839a 90              nop
b09b839b ff2570769fb0    jmp     dword ptr [tcpip!SendICMPErr+0xa916 (b09f7670)]
b09b83a1 90              nop
b09b83a2 90              nop
b09b83a3 90              nop
b09b83a4 90              nop
b09b83a5 90              nop
tcpip!IPRcvComplete:
b09b83a6 8bff            mov     edi,edi
b09b83a8 57              push    edi
b09b83a9 33ff            xor     edi,edi
b09b83ab 393dc89a9fb0    cmp     dword ptr [tcpip!SendICMPErr+0xcd6e (b09f9ac8)],edi
b09b83b1 7e1a            jle     tcpip!IPRcvComplete+0x27 (b09b83cd)
b09b83b3 56              push    esi
b09b83b4 bef99a9fb0      mov     esi,offset tcpip!SendICMPErr+0xcd9f (b09f9af9)
```

Saving a Screen Image

Make sure you have highlighted **tcpip!SendICMPErr**, as shown above.

On your keyboard, press the PrntScrn key.

Click **Start**, type in **PAINT**, and open Paint.

Press **Ctrl+V** to paste in the image of your desktop.

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.

Last Modified: 10-21-13 9:47 am