

# Windows Kernel Exploitation Tutorial Part 1: Setting up the Environment

📅 June 19, 2017 👤 rootkit

## Intro

Recently, I had the pleasure to attend the training on Windows Kernel Exploitation at [nullcon](#) by the [Hack-SysTeam](#). The training was well executed, and I got the intro into the world of kernel. But, as you know, nobody could teach you internals about Kernel Exploitation in a couple of days. So I thought of diving into the kernel, and share everything that I learn in the process. The series would be coming in parts, as I find the time to learn and document everything that I encounter.

## Prerequisites

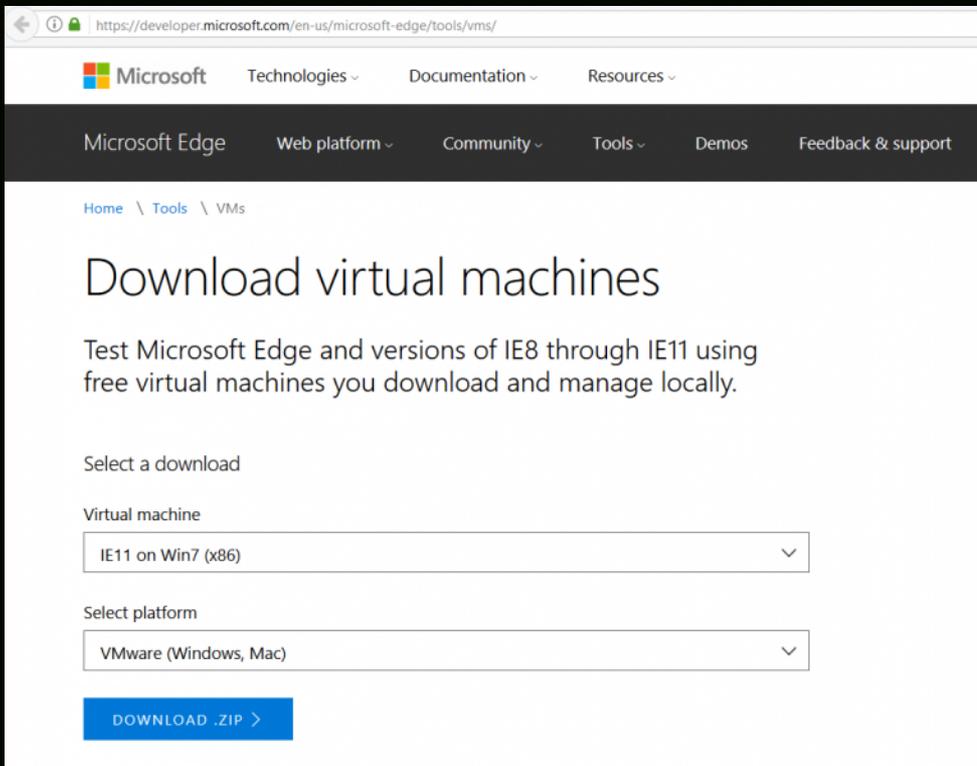
- VMWare or Virtualbox (I'll be using VMWare for this series)
- Windows 7 x86 VM
- Internet Connection for downloading symbols
- Powerful enough machine to run the VMs
- Basic know-hows in day to day computing tasks.

## Why VMs?

Visualize kernel as the heart of OS. Now, if you have done any application exploitation in the past, you'd know that you basically crash the application and try to exploit the crash. Applications can easily be recovered once crashed, just double click to run again. Now, if you accidentally crash the kernel, it's like stopping the heart, the OS would just halt/crash/BSOD, and could lead to loss of data, corruption etc. in your machine, and you'd be constantly rebooting the whole machine. VMs are easily setup, isolated and causes no harm if corrupted. Many people just run the Debuggee VM (the machine which you'd be crashing alot) in the VM, and keep their host as the Debugger machine. I'd be running the setup where both of them would be VM, just to keep things neat and tidy.

## Steps

1. Install Windows 7 x86 in the VM, free download is available at Microsoft [VM download page](https://developer.microsoft.com/en-us/microsoft-edge/tools/vms/).

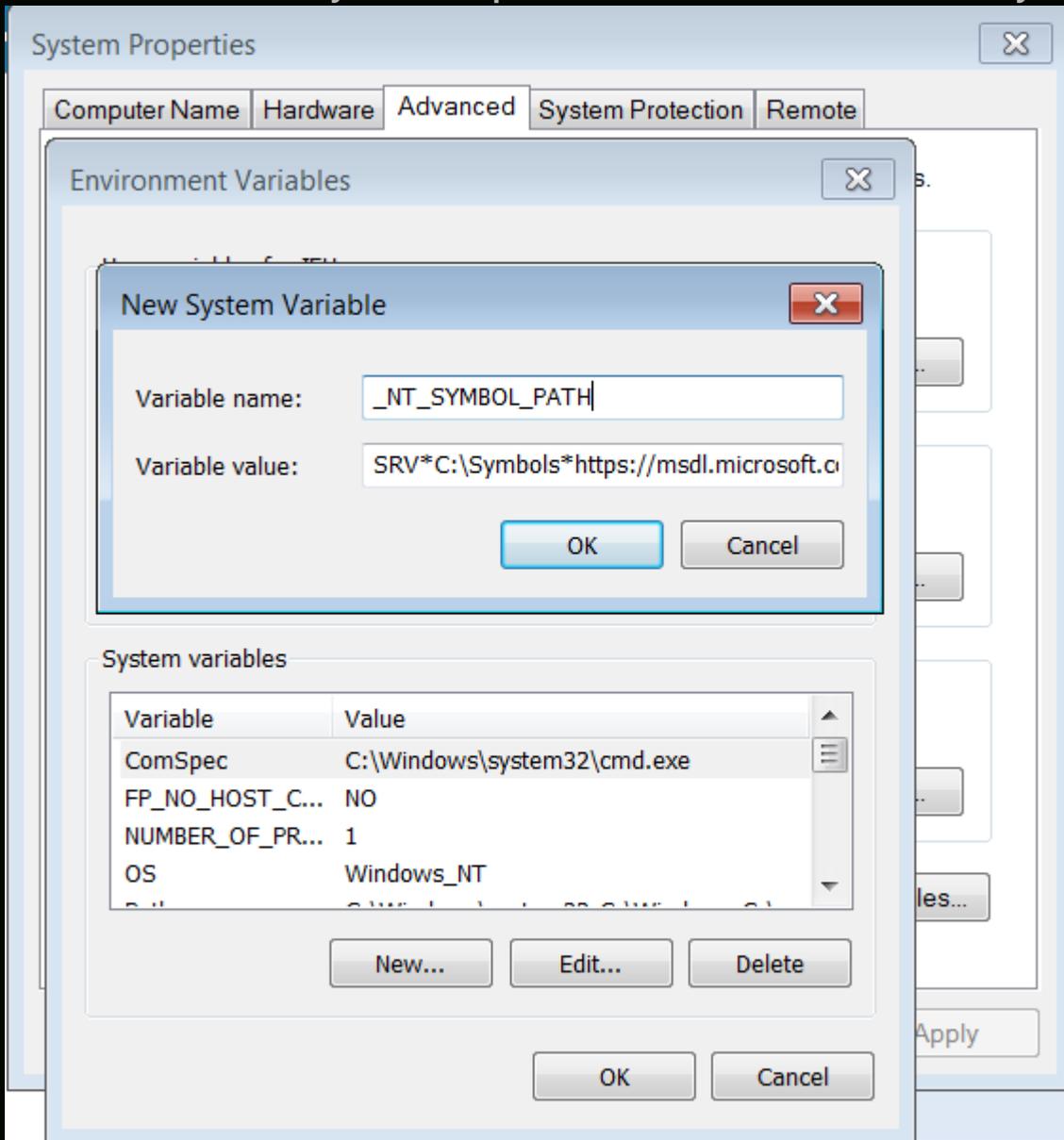


2. After the Debugger VM is setup and ready to boot, we'd need to install WinDbg, get it [here](#).

3. We'd also need to setup Debugging Symbols in the Debugger VM. Fortunately, Microsoft provides public debugging symbols.

- Go to Computer -> Properties -> Advanced system settings -> Environment Variables.
- Create a new System Variable as follows:
  - Variable Name: **\_NT\_SYMBOL\_PATH**

- Variable Value: `SRV*C:\Symbols*https://msdl.microsoft.com/download/symbols`



4. After WinDbg is installed, we would need to enable debugging in BCD:

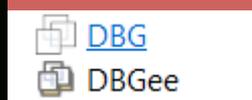
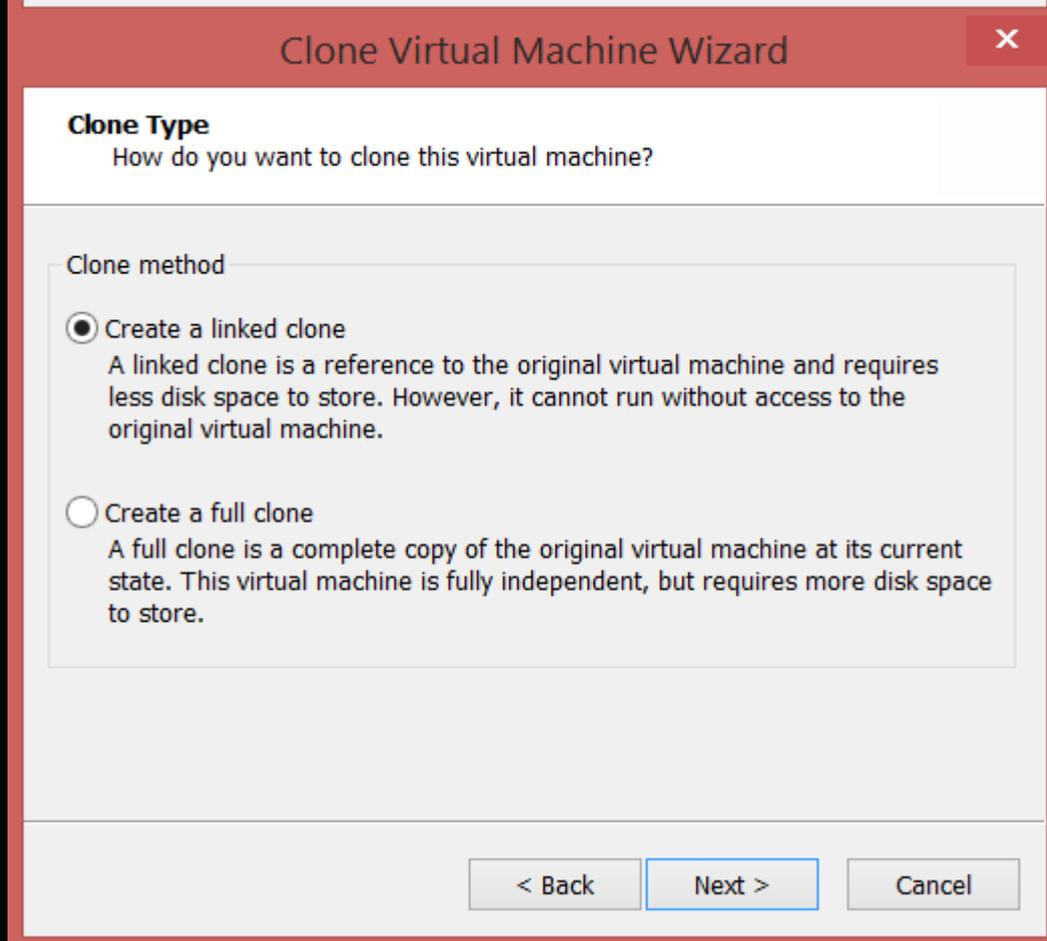
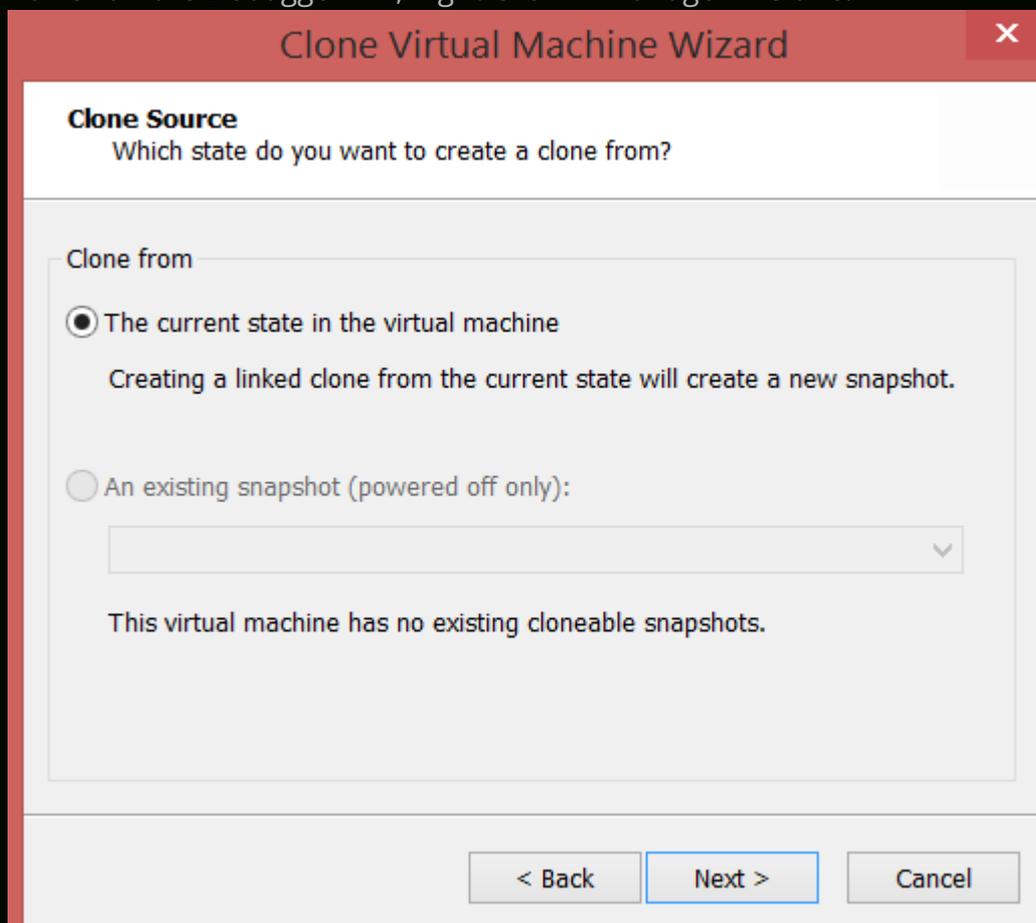
- Run `cmd` as administrator, and execute the following commands:

```
1 bcdedit /copy {current} /d "Win7Dbg"  
2 bcdedit /debug {0275ed04-3c06-11e3-a1c0-b6bd309a633d} on  
3 bcdedit /dbgsettings
```

```
Administrator: C:\Windows\System32\cmd.exe  
Microsoft Windows [Version 6.1.7601]  
Copyright (c) 2009 Microsoft Corporation. All rights reserved.  
  
C:\Windows\system32>bcdedit /copy {current} /d "Win7Dbg"  
The entry was successfully copied to {0275ed04-3c06-11e3-a1c0-b6bd309a633d}.  
  
C:\Windows\system32>bcdedit /debug {0275ed04-3c06-11e3-a1c0-b6bd309a633d} on  
The operation completed successfully.  
  
C:\Windows\system32>bcdedit /dbgsettings  
debugtype Serial  
debugport 1  
baudrate 115200  
The operation completed successfully.  
  
C:\Windows\system32>_
```

5. Now, we'll create the Debugee VM, by creating a linked clone of the Debugger VM.

6. Power off the Debugger VM, Right Click -> Manage -> Clone.



7. Now, we need to enable Serial Ports on both the VMs, so as to make them communicate using a Virtual Serial Port.

- For the Debugger VM, Right Click -> Settings -> Add -> Serial Port

Virtual Machine Settings

Hardware Options

Device	Summary
Memory	1 GB
Processors	1
Hard Disk (IDE)	127 GB
CD/DVD (SATA)	
Network Adapter	
USB Controller	
Sound Card	
Display	

Memory

Specify the amount of memory allocated to this virtual machine. The memory size must be a multiple of 4 MB.

1024 MB

Recommended memory  
ping may  
his size.)  
memory  
Recommended minimum

### Add Hardware Wizard

**Serial Port Type**

What media should this serial port access?

Serial port

Use physical serial port on the host

Output to file

Output to named pipe

< Back   Next >   Cancel

Add...   Remove

OK   Cancel   Help

Hardware Options

Device	Summary
 Memory	1 GB
 Processors	1
 Hard Disk (IDE)	127 GB
 CD/DVD (SATA)	
 Network Adapter	
 USB Controller	
 Sound Card	
 Display	

## Memory

Specify the amount of memory allocated to this virtual machine. The memory size must be a multiple of 4 MB.

1024 MB

## Add Hardware Wizard



## Specify Socket

Which socket should this serial port connect to?

## Named pipe

 ▾ ▾

## Device status

 Connect at power on

&lt; Back

Finish

Cancel



Remove

OK

Cancel

Help

Hardware Options

Device	Summary
 Memory	1 GB
 Processors	1
 Hard Disk (IDE)	127 GB
 CD/DVD (SATA)	Auto detect
 Network Adapter	NAT
 USB Controller	Present
 Sound Card	Auto detect
 Serial Port	Using named pipe \\.\pipe\KernelDbg
 Display	Auto detect

 Add...

Remove

## Device status

- Connected
- Connect at power on

## Connection

- Use physical serial port:

Auto detect 

- Use output file:

- Use named pipe:

\\.\pipe\KernelDbg

This end is the server. The other end is an application. 

## I/O mode

- Yield CPU on poll

Allow the guest operating system to use this serial port in polled mode (as opposed to interrupt mode).

OK

Cancel

Help

- For the Debugee VM, Right Click -> Settings -> Add -> Serial Port

The image shows a Windows-style dialog box titled "Virtual Machine Settings" with a close button (X) in the top right corner. The dialog has two tabs: "Hardware" and "Options", with "Hardware" selected. On the left, there is a list of hardware components with a "Summary" column:

Device	Summary
Memory	1 GB
Processors	1
Hard Disk (IDE)	127 GB
CD/DVD (SATA)	
Network Adapter	
USB Controller	
Sound Card	
Display	

On the right, there is a "Memory" section with the text: "Specify the amount of memory allocated to this virtual machine. The memory size must be a multiple of 4 MB." Below this is a spin box set to "1024" MB. At the bottom of the main dialog are "Add..." and "Remove" buttons.

Overlaid on top of the main dialog is a smaller dialog box titled "Add Hardware Wizard" with a close button (X) in the top right corner. It has a section titled "Serial Port Type" with the question "What media should this serial port access?". Below this is a "Serial port" section with three radio button options:

- Use physical serial port on the host
- Output to file
- Output to named pipe

At the bottom of the "Add Hardware Wizard" dialog are three buttons: "< Back", "Next >", and "Cancel". At the bottom of the "Virtual Machine Settings" dialog are three buttons: "OK", "Cancel", and "Help".



Hardware Options

Device	Summary
Memory	1 GB
Processors	1
Hard Disk (IDE)	127 GB
CD/DVD (SATA)	
Network Adapter	
USB Controller	
Sound Card	
Display	

Memory

Specify the amount of memory allocated to this virtual machine. The memory size must be a multiple of 4 MB.

Memory size: 1024 MB

### Add Hardware Wizard

**Specify Socket**  
Which socket should this serial port connect to?

Named pipe

This end is the client.

The other end is an application.

Device status

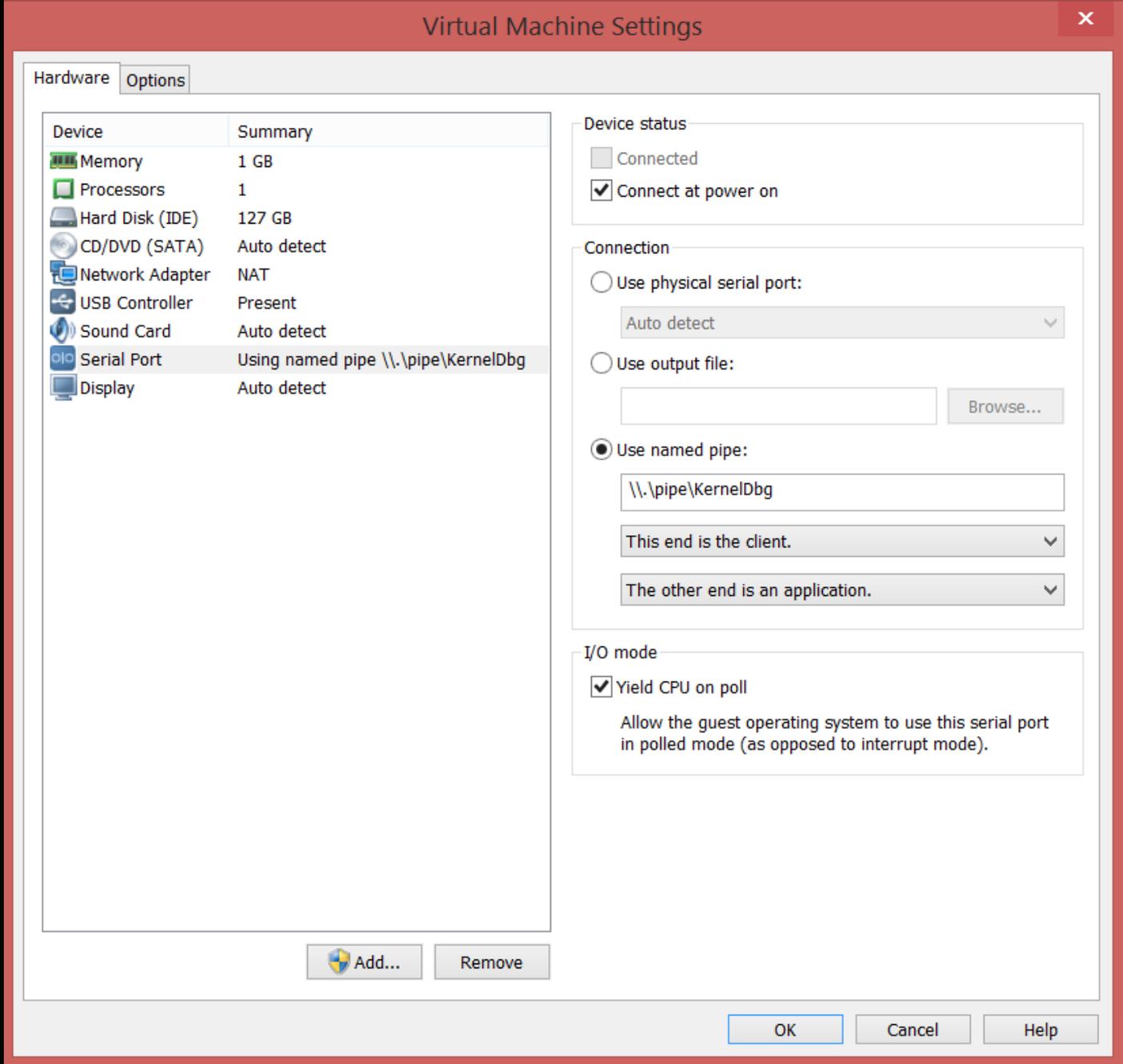
Connect at power on

< Back Finish Cancel

Add... Remove

OK Cancel Help

mmended memory  
ping may  
his size.)  
memory  
mmended minimum



8. Now, turn on the Debugger VM first (**always**), and select the first option without the *[debugger enabled]*.

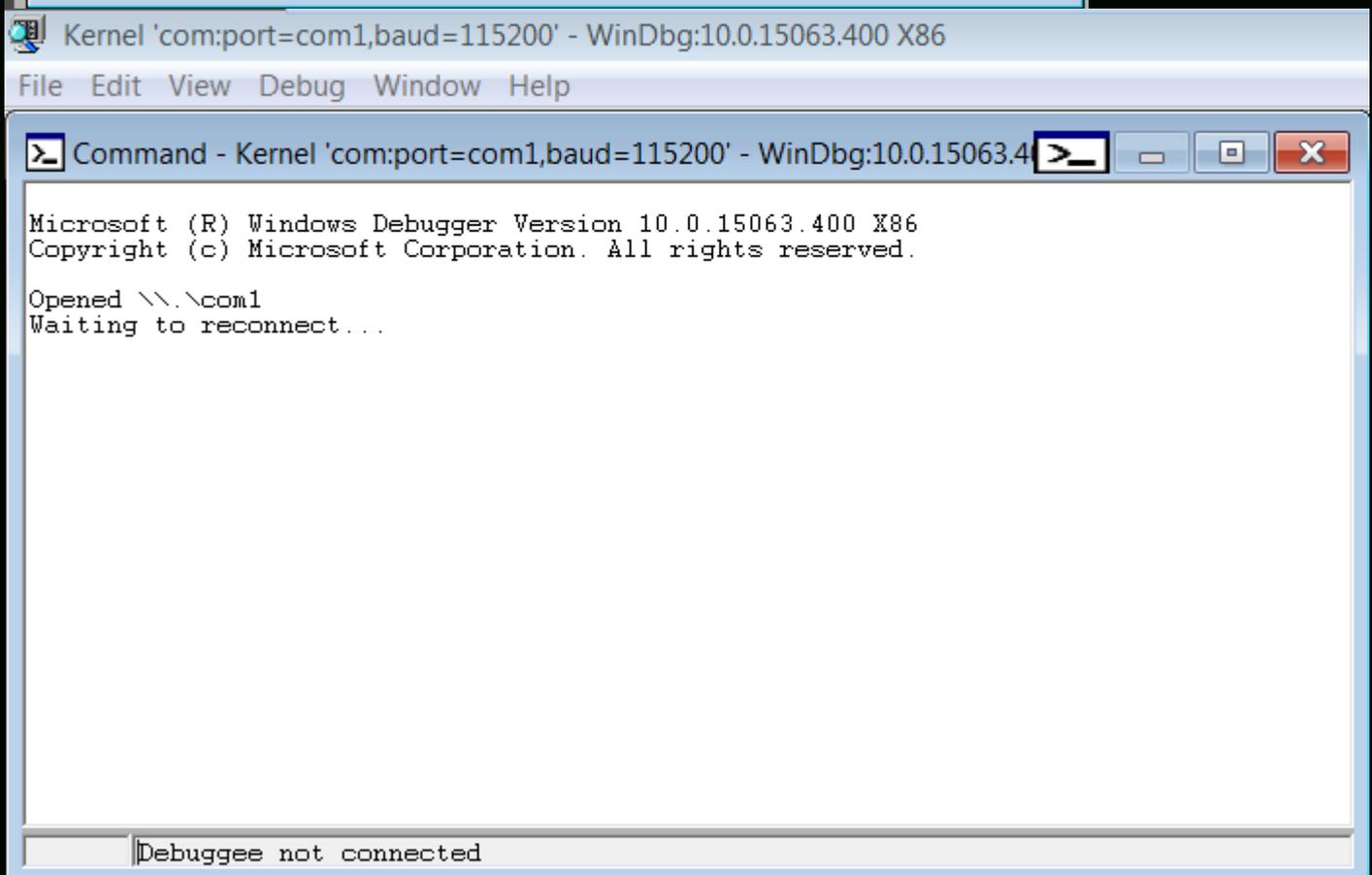
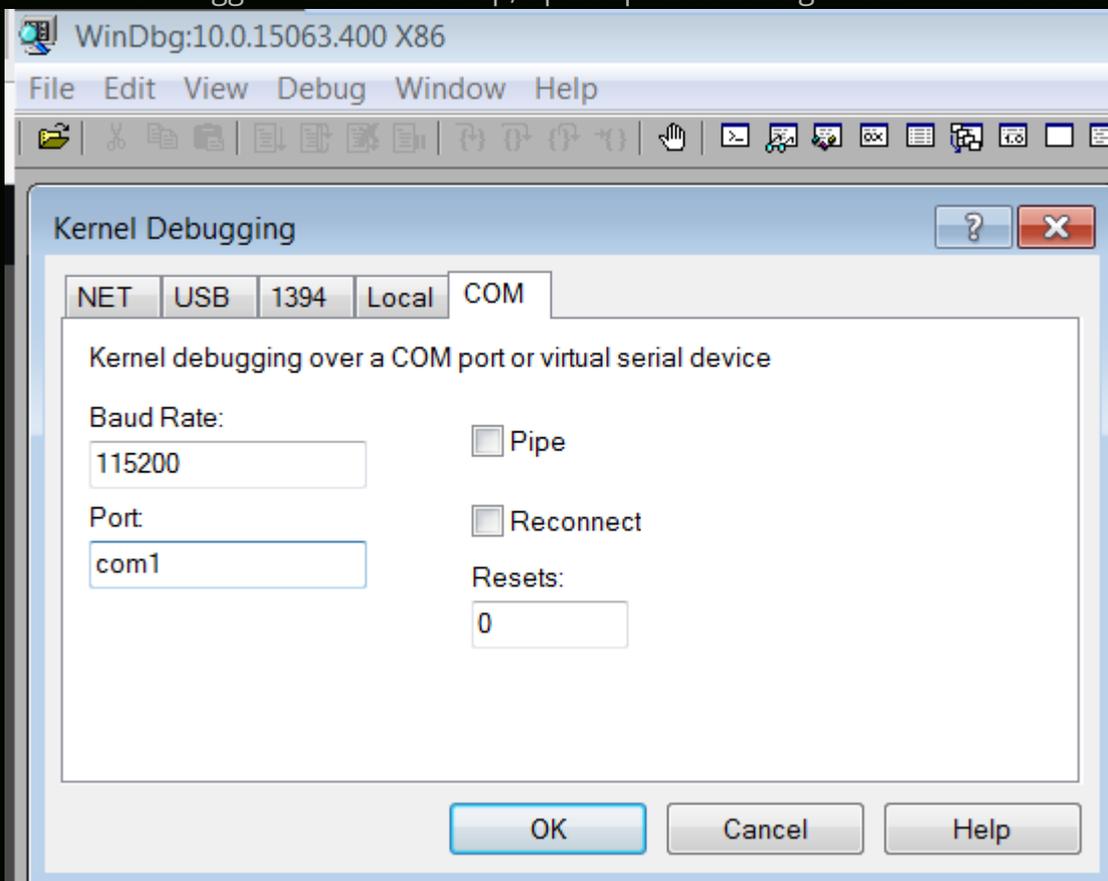
### Windows Boot Manager

Choose an operating system to start, or press TAB to select a tool:  
(Use the arrow keys to highlight your choice, then press ENTER.)

```
Windows 7 >
win7Dbg [debugger enabled]
```

To specify an advanced option for this choice, press F8.

9. After the Debugger VM is booted up, open up the WinDbg -> File -> Kernel Debug -> COM.



10. Now, boot up the Debugee VM, and select the second option with *[debugger enabled]*.

Windows Boot Manager

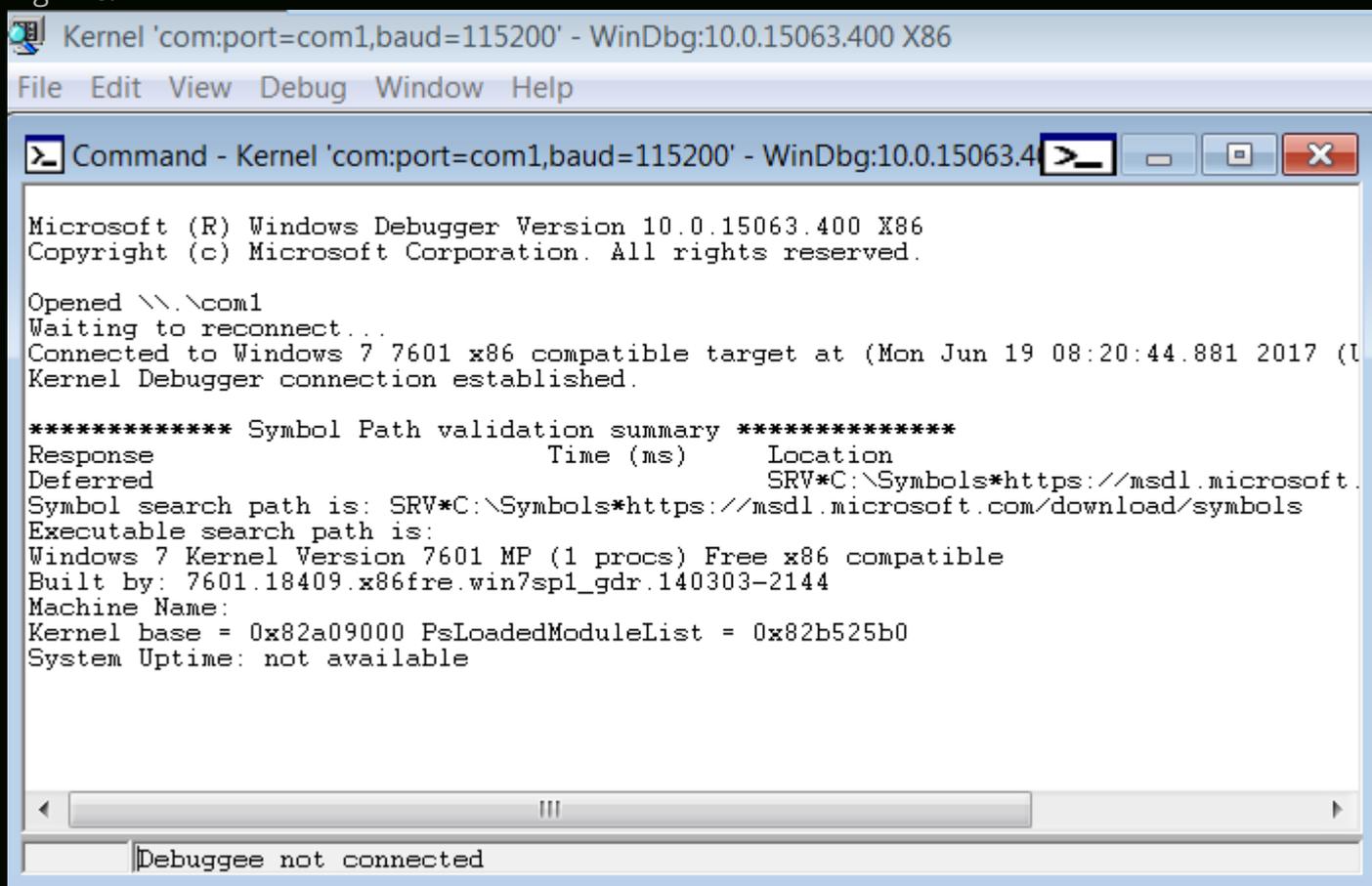
Choose an operating system to start, or press TAB to select a tool:  
(Use the arrow keys to highlight your choice, then press ENTER.)

Windows 7

win7Dbg [debugger enabled]

To specify an advanced option for this choice, press F8.

11. Now, if you see the following output in the WinDbg in your Debugger VM, congrats, everything is working fine.



The screenshot shows the WinDbg application window titled "Kernel 'com:port=com1,baud=115200' - WinDbg:10.0.15063.400 X86". The menu bar includes File, Edit, View, Debug, Window, and Help. A command window is open, displaying the following text:

```
Microsoft (R) Windows Debugger Version 10.0.15063.400 X86
Copyright (c) Microsoft Corporation. All rights reserved.

Opened \\.\com1
Waiting to reconnect...
Connected to Windows 7 7601 x86 compatible target at (Mon Jun 19 08:20:44.881 2017 (UTC))
Kernel Debugger connection established.

***** Symbol Path validation summary *****
Response           Time (ms)          Location
Deferred           0                  SRV*C:\Symbols*https://msdl.microsoft.com/download/symbols
Symbol search path is: SRV*C:\Symbols*https://msdl.microsoft.com/download/symbols
Executable search path is:
Windows 7 Kernel Version 7601 MP (1 procs) Free x86 compatible
Built by: 7601.18409.x86fre.win7sp1_gdr.140303-2144
Machine Name:
Kernel base = 0x82a09000 PsLoadedModuleList = 0x82b525b0
System Uptime: not available
```

At the bottom of the window, a status bar indicates "Debuggee not connected".

12. Now, after the Debugee VM is booted up, hit the *Break* button, and you should get an interactive *kd>* prompt, ready to take commands.

```
Kernel 'com:port=com1,baud=115200' - WinDbg:10.0.15063.400 X86
File Edit View Debug Window Help
Break (Ctrl+Break)
Microsoft (R) Windows Debugger Version 10.0.15063.400 X86
Copyright (c) Microsoft Corporation. All rights reserved.

Opened \\.\com1
Waiting to reconnect...
Connected to Windows 7 7601 x86 compatible target at (Mon Jun 19 08:20:44.881 2017 (UTC - 7:00)), ptr64 FALSE
Kernel Debugger connection established.

***** Symbol Path validation summary *****
Response                               Time (ms)      Location
Deferred                               SRV*C:\Symbols*https://msdl.microsoft.com/download/symbols
Symbol search path is: SRV*C:\Symbols*https://msdl.microsoft.com/download/symbols
Executable search path is:
Windows 7 Kernel Version 7601 MP (1 procs) Free x86 compatible
Built by: 7601.18409.x86fre.win7sp1_gdr.140303-2144
Machine Name:
Kernel base = 0x82a09000 PsLoadedModuleList = 0x82b525b0
System Uptime: not available
```

```
Kernel 'com:port=com1,baud=115200' - WinDbg:10.0.15063.400 X86
File Edit View Debug Window Help
Command - Kernel 'com:port=com1,baud=115200' - WinDbg:10.0.15063.400 X86
Symbol search path is: SRV*C:\Symbols*https://msdl.microsoft.com/download/symbols
Executable search path is:
Windows 7 Kernel Version 7601 MP (1 procs) Free x86 compatible
Built by: 7601.18409.x86fre.win7sp1_gdr.140303-2144
Machine Name:
Kernel base = 0x82a09000 PsLoadedModuleList = 0x82b525b0
System Uptime: not available
KDTARGET: Refreshing KD connection
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!RtlpBreakWithStatusInstruction:
82a837b8 cc int 3
kd>
```

13. Now, just to be sure that the symbols have been loaded correctly, run the following commands:

```
1 !sym noisy
2 .reload
```

```
Kernel 'com:port=com1,baud=115200' - WinDbg:10.0.15063.400 X86
File Edit View Debug Window Help
82a837b8 cc          int      3
kd> !sym noisy
noisy mode - symbol prompts on
kd> .reload
Connected to Windows 7 7601 x86 compatible target at (Mon Jun 19 08:29:56.224 2017 (UTC - 7:00)), ptr
SYMSRV:  BYINDEX: 0x5
          c:\symbols*https://msdl.microsoft.com/download/symbols
          ntkrpamp.pdb
          AB1263CE4C444E518224A213E053A5D72
SYMSRV:  PATH: c:\symbols\ntkrpamp.pdb\AB1263CE4C444E518224A213E053A5D72\ntkrpamp.pdb
SYMSRV:  RESULT: 0x00000000

DBGHELP: nt - public symbols
          c:\symbols\ntkrpamp.pdb\AB1263CE4C444E518224A213E053A5D72\ntkrpamp.pdb
Loading Kernel Symbols
.....

Press ctrl-c (cdb, kd, ntsd) or ctrl-break (windbg) to abort symbol loads that take too long.
Run !sym noisy before .reload to track down problems loading symbols.

.....
Loading User Symbols
.....
Loading unloaded module list
kd>
```

## Conclusion

Congrats, we have successfully setup Kernel Debugging. The next part would be coming up soon, digging deeper into the kernel, and analyzing the Stack Overflow in Kernel Space.

Posted in [Kernel, Tutorial](#) Tagged [Exploitation, Kernel, Tutorial, Windows](#)

