**ARAZIM MiniFilter Workshop**

Workshop Framework code + Solution is currently here:

<https://github.com/OmerBDeveloper/MiniFilterFS/tree/master/FsMinifilter/FsFilter>

**Prerequisites for participants**

**Visual Studio:**

1. Have a version of VS installed
   1. Free option is Visual Studio Community 2017:

<https://www.visualstudio.com/thank-you-downloading-visual-studio/?sku=Community&rel=15>

1. Install a new enough SDK:
   1. At least 10.0.10586 from here:

<https://developer.microsoft.com/en-us/windows/downloads/sdk-archive>

1. Install a new enough WDK:
   1. For example <https://go.microsoft.com/fwlink/p/?linkid=859232>
   2. Or just google windows WDK and download the newest

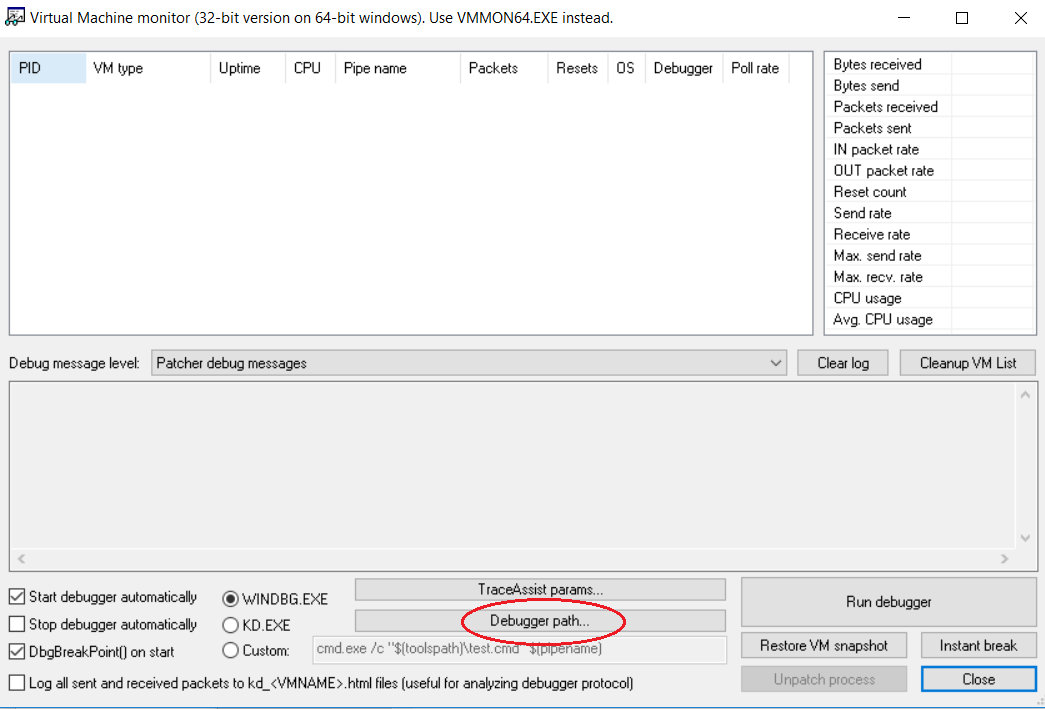
**VMWare Workstation:**

1. Install windows 7 32 bit from ISO to a new VM workstation Machine.
2. Create a snapshot “CLEAN”
3. Install VirtualKD on machine
   1. You should have a folder called VirtualKD with the version number
   2. Inside this folder you should have a folder called “Target”
   3. Copy the “Target” machine to the target VM
   4. Run vminstall.exe
4. Reboot
5. Create a snapshot “READY”

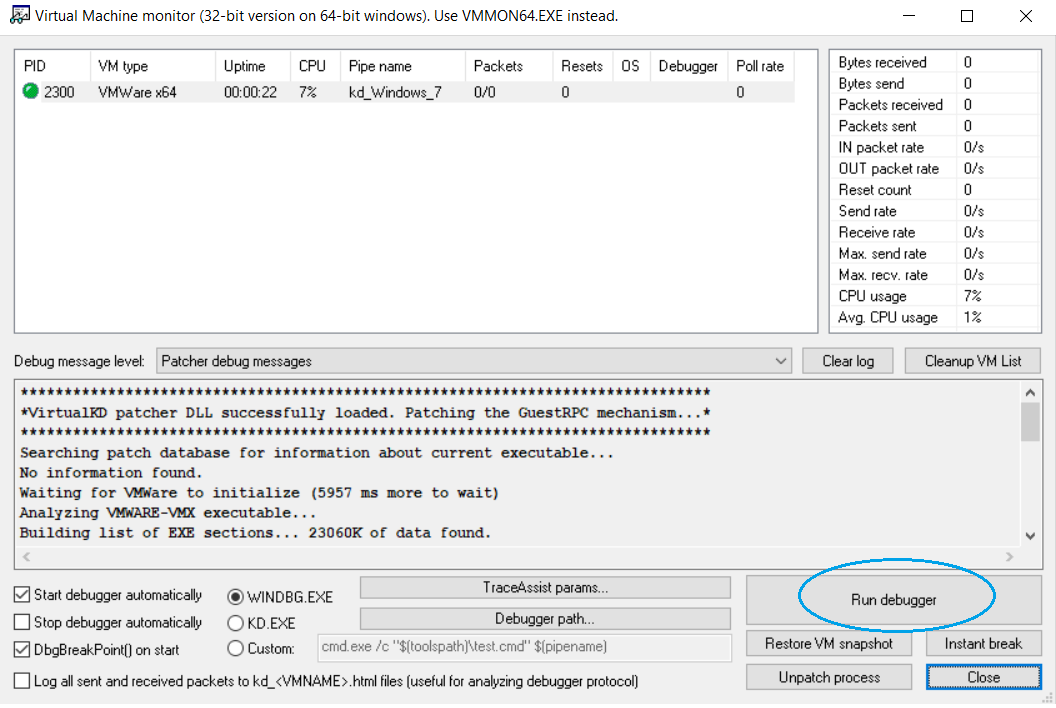
**Running with kernel debugger for the first time**

1. Run vmmon (part of virtual kd) on the host machine. You need to run that version with the same bit architecture as the target machine (the vm).
2. Set the debugger path of the vmmon to windbg

The path should be something like C:\Program Files (x86)\Windows Kits\10\Debuggers\x86



1. While vmmon is running restart the vm
2. Press Run debugger



1. Vmmon should lunch windbg and you can start debugging

**Kernel Debugging:**

1. Allow receiving out kernel dbgprint messages:
   1. “ed Kd\_DEFAULT\_Mask 8”

**Windbg useful commands:**

1. g \ F5 – for continue the run
2. F10 – next step (debugging)
3. F11 – step into (debugging)
4. .sympath+ c:\path\to\driver.pdb – adds the symbols of the driver
   1. Or just add the path using ctrl + s
5. .srcpath+ c:\path\to\driver\code – adds the source to the debugger
   1. Or just add the path using ctrl + p
6. .reload /f - for telling windbg to load the new symbols we gave it
7. Ctrl + O – open source file
8. Ctrl + break \ *debug -> break* option in the menu – in order to stop the vm

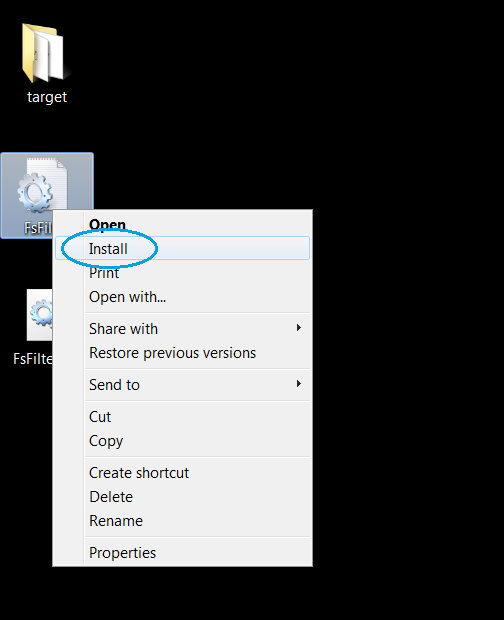
**Installing the driver on the vm:**

After compiling with Visual Studio the driver framework you have few output files, among them:

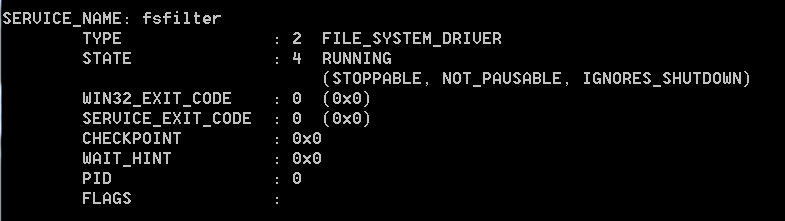
1. FsFilter.inf – a file which let you install the driver on the machine
2. FsFilter.sys – the driver itself
3. FsFilter.pdb the driver symbols – for debugging

In order to install the driver:

1. Copy both FsFilter.inf and FsFilter.sys to the same folder on the vm (you can use desktop)
2. Right-click on the FsFilter.inf and click install



1. Now you have the driver installed on the machine, but **not** running.
2. Run cmd as administrator and type to following command:  
   sc start fsfilter
3. The output should be something like:



1. In order to upgrade you driver to new on, first you should stop the driver
2. Run cmd as administrator and type to following command:  
   sc stop fsfilter
3. Then repeat step 1 with the new files
4. Remember – when you upgrade the driver you need to tell windbg to reload the new symbols, using the .reload /f command

**Instructions for tutor**

**Make sure all workshop participants have a computer with required setup.**

**Recommendations**

In order for this workshop to be useful we students should know by now:

1. That there is a different between kernel mode and user mode
2. What is a virtual machine
3. Basic debugging abilities (i.e, what is a break point)
4. Have seen cmd before

**Workshop steps:**

1. Create a clean vm with virtual kd installed and a clean **SNAPSHOT**!
   1. Play some with the debugger – stop the machine, continue
2. Release the code framework for the workshop participants.
3. Compile the framework
4. Do a simple installation of the clean driver (which does nothing) on the machine
5. Add a debug print in driver entry – look at the debugger and see that the line is printed
6. Do the pre-create operation
   1. Debug
   2. Debug
   3. Debug
   4. Why????
   5. Oh why ☹
   6. ????
   7. PROFIT!
7. If someone finished everything in the workshop you can give him the following exercise:

Now monitor the ReadFile operation and check whether the work “virus” appears in the data of the file.

Even If they will finish this, they will see that the performance of the vm is terrible. Now you can challenge them and let them think why (they scan the same buffer a lot of times) and how to solve this (caching).

**Pitfalls**

1. Always break the debugger and reload the symbols after changing the driver
2. They won’t see any debugging output without typing the magic command mentioned in the kernel debugging section

**Extra questions**

1. “where does the debugging output goes to in driver?”  
   you can see the output using the kernel debugger or dbgview with the right settings
2. “can anyone just install a driver on a given computer?”  
   <https://docs.microsoft.com/en-us/windows-hardware/drivers/install/kernel-mode-code-signing-requirements--windows-vista-and-later->