System setup instructions

This document contains detailed instructions for setting up your system for the hands-on labs and to attend your upcoming CodeMachine training. If you have any questions about the setup, please email codemachine@outlook.com and we will respond within 24 hours.

System requirements

You will need to use your laptop or desktop system to attend the training and perform the hands-on labs. The hands-on lab exercises will not damage or negatively impact your host system as the labs will be performed inside the virtual machine (VM) guest OS. The table below describes the system requirements.

CPU	CPU with support for hardware virtualization. CPU must be fast enough to run at least one VM without any performance degradation.
Memory	At least 8 GB RAM. Preferably 16 GB RAM.
Storage	At least 60 GB of free space. Preferably SSD/NVMe.
Display	Resolution of at least 1920 x 1080 (HD). Display size at least 14 inches diagonal.
Network	WiFi. Preferably Ethernet. The system must have reliable Internet access.
Host OS	Windows 10 64-bit Home, Professional or Enterprise Edition. Windows need not be activated. A trial version is fine.
Virtualization	VMware (preferred) OR Hyper-V (built into the Professional and Enterprise edition of Windows 10).

Assumptions

The instructions in this document make certain assumptions about the setup you will be using. We strongly recommend that you set up your system as per the instructions in this document. If for some reason you are unable to do so and would like to set it up differently, please sure that have a working kernel debugging setup BEFORE the start of the class. Here are some of the assumptions made in this document:

- Host OS is Windows 10 64-bit.
  Some students prefer to use Linux/macOS with Windows VMs, where one VM acts as the Windows host system and the second VM acts as the Windows guest system. Windows kernel debugging over named pipes does work in this setup. However, this document assumes that your host system is Windows 10 64-bit and will be running a single guest VM. If you plan to use Linux or macOS, you must set up two Windows VMs and configure kernel debugging to work between them. This configuration is beyond the scope of this document.
- Virtualization software is VMware Workstation Player 16.
  A pre-configured VMware VM has been made available to you which requires VMware Workstation Player 16 to be installed on your host. If you prefer to use Hyper-V instead, please refer to the article https://codemachine.com/articles/system_setup_for_kernel_development.html for the steps to set up your Hyper-V VM.
- Kernel debugging uses a serial port over a named pipe (KDCOM).

The VM that has been made available to you has been configured for KDCOM using serial port COM1 mapped to the name pipe \\.*pipe**winlabvm*. If you choose to use a different kernel debugging transport such as KDNET or VirtualKD. Please make the necessary changes to the VM configuration and to the WinDBG workspace file (*winlabvm.debugtarget*).

Host system setup

Host OS

The latest version of Windows 10 64-bit Home/Professional/Enterprise edition must be installed on the host system and the latest updates must be applied. OS installation images can be obtained from MSDN. Windows need not be activated. A trial version of Windows will work fine.

Internet access

The host system must have reliable internet access throughout the course to look up MSDN, for the debugger to download symbols from Microsoft's public symbol server and to attend the online training.

Enterprise Windows Driver Kit (EWDK)

The EWDK contains all the tools required to build kernel software drivers on Windows. Download the "EWDK for Windows 10, version 2004 with Visual Studio Build Tools 16.7" (Size=~13GB) from the link below and extract the contents of the .ISO file into the folder *c:\EWDK*. The EWDK uses XCOPY deployment, therefore no installation is necessary. The instructions in the rest of this document assume that you are using the EWDK, and it is available at *c:\EWDK*.
https://docs.microsoft.com/en-us/legal/windows/hardware/enterprise-wdk-license-2019

Windows debugger

Download the WinDBG preview from the Windows App Store link below and install it with default settings.
https://www.microsoft.com/en-us/p/windbg-preview/9pgjgd53tn86

Debugging symbols

Run the following command in an administrative command (*cmd.exe*) window to set up the environment variable for the debugger to download symbols from Microsoft's public symbol server.

```
setx _NT_SYMBOL_PATH SRV*c:\symbols*https://msdl.microsoft.com/download/symbols
```

Source code editor

The hands-on labs for most CodeMachine courses involve programming in C/C++. Therefore, a good code editor must be installed on your host system. You can use your favorite source code editor. If you prefer to use Visual Studio Code, you can download the "Visual Studio Code" "System Installer" "64-bit" for "Windows 7, 8, 10" from https://code.visualstudio.com/#alt-downloads and run the installer with default settings.
All the lab exercises will be built using the Enterprise Windows Driver Kit (EWDK) and live debugging will be performed using WinDBG Preview. Therefore, Visual Studio 2019 is completely optional. If you choose to use Visual Studio 2019 IDE for editing and building, please install the Windows 10 SDK and Windows Driver Kit (WDK Version 2004) Extension components.

Training package

Download the file *host.zip* (Size=~500MB) from the link below:
https://1drv.ms/u/s!AtvUy24DJoLGoVqNSTDS3yKHN-mB
The downloaded files *host.zip* contain some publicly available tools which may be blocked by Windows Defender. Therefore, before extracting the contents of *host.zip*, you must add the destination path to the list of excluded folders in Windows Defender. You must also remove any Internet Zone information from the downloaded file. To do so, on the host system, run the following commands in an administrative

PowerShell command window and then unzip the contents of *host.zip* into the folder *c:\cm* using the password *CodeMachine*.

```
Add-MpPreference -ExclusionPath c:\cm
Unblock-File host.zip
```

Virtual Machine

A pre-configured VM with all the necessary tools and symbols has been made available. To use this VM you will need VMWare Workstation 16 Player. Recent versions of VMware Workstation Player can run on a system with Hyper-V and Virtualization Based Security (Core Isolation) enabled. So, you will NOT need to disable Hyper-V or any of its features to use VMware. VMWare Workstation 16 Player will detect the presence of Hyper-V and use the "Windows Hypervisor Platform (WHP)" feature.
Download the file *winlabvm.zip* (Size=~7GB) from the link below:
https://1drv.ms/u/s!AtvUy24DJoLGoVqNSTDS3yKHN-mB
Windows Defender Real-time protection may interfere with this download, so it is recommended that you disable that feature while downloading the ZIP file. This ZIP file includes all the files required to install VMware VM workstation player, run the pre-configured Windows 10 Enterprise x64 VM (*winlabvm*) and perform kernel debugging of the VM using a serial debugging transport. You must remove any Internet Zone information from the downloaded file. To do so, on the host system, run the following commands in an administrative PowerShell command window and then unzip *winlabvm.zip* into the folder *c:\cm\winlabvm*.

```
Unblock-File winlabvm.zip
```

1. If VMware is not already installed on your system, run the Windows installer *c:\cm\winlabvm\VMware-player-16.1.1-17801498.exe* to install VMware Workstation Player. When prompted, click on "Install Windows Hypervisor Platform (WHP) automatically". It is not necessary to install the "Enhanced Keyboard Driver".
2. Start VMware Workstation Player, select the appropriate license, and click on "Open a Virtual Machine"
3. Point it to the folder *c:\cm\winlabvm*, select the file *winlabvm.vmx* and click "Open". *Winlabvm* should be added to the VM list. Do NOT power on the VM yet.

Kernel Debugging

1. Start WinDBG Preview
2. In WinDBG, select "File" -> "Open workspace", point it to the folder *c:\cm\winlabvm*, select the file *winlabvm.debugtarget* and click on "Open".
3. WinDBG should display the following message:
   ```
   Waiting for pipe \\.\pipe\WINLABVM
   Waiting to reconnect...
   ```
4. In VMware Workstation Player, select *winlabvm* from the VM list and click on "Play virtual machine".
5. You may be prompted to select "I Moved It" OR "I Copied It". Select "I Copied It" so that all the VM unique identifiers are regenerated. As the VM starts up you should start seeing debug spew in WinDBG. If you are prompted for "Software Updates", select "Remind Me Later".
6. During the VM startup you will be asked to "Choose an operating system", select "Windows 10 [debugger]".
7. Login into the VM using the username *student* and the password *student*.
8. In WinDBG, select "Home" -> "Break" to break into the VM
9. When the WinDBG prompt is displayed, type *.sympath* (that's dot sympath) and ensure that the symbol path is similar to the one displayed below:
   ```
   SRV*c:\symbols*https://msdl.microsoft.com/download/symbols
   ```
10. In WinDBG, type *!process -1 0*, which should display information about the current process.
11. In WinDBG, type *g* to continue running the VM.

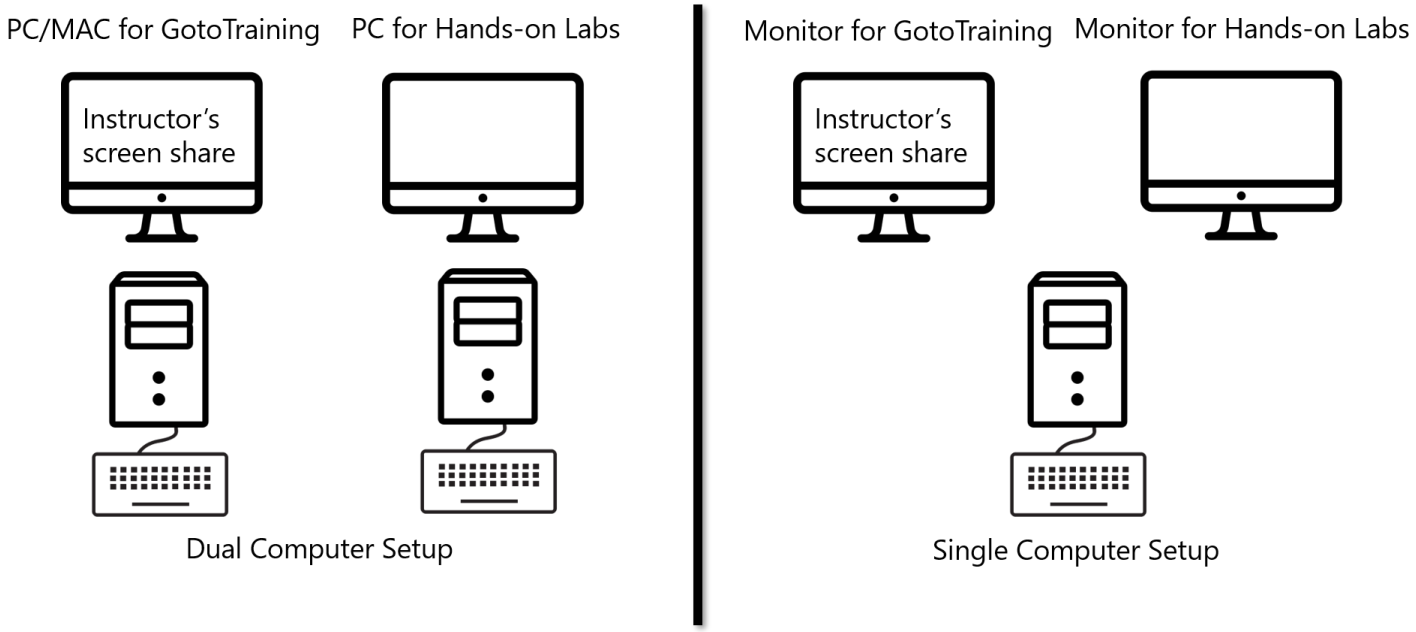
Now you have your virtual machine and kernel debugging working.

Attending the training

You will need a reliable Internet connection, a Windows PC with a working audio device, and the GotoTraining native client installed locally. While GotoTraining does provide a browser-based client, your training experience may not be optimal. During the training, you can ask questions through the online chat channel or over computer audio, both of which are supported by the GotoTraining native client. We do NOT use video on either side, so audio and screen-share only.

We advocate that you run the GotoTraining client on a different PC or Mac than the Windows 10 PC you will be using for the hands-on labs. If that is not feasible, we recommend a dual-monitor setup with one monitor dedicated for GotoTraining. This will ensure that you can clearly see the instructor's screen share on the GotoTraining monitor while performing the hands-on labs. Please see the illustration below for recommended setups.

Recommended setup for online (virtual) training using GotoTraining



CodeMachine courses are intensive and run on a tight schedule. There is no time during the class to deal with system or software configuration issues. Please read this document carefully and set up your system as per the instructions BEFORE the class.

Look forward to having you in class.

CodeMachine Team
codemachine@outlook.com

