**Debugging Linux on Windows with Visual Studio 2019**

As mentioned earlier, we created a worker that could run as a worker service, and we ran it through the Remote extension in Visual Studio Code. There are, however, cases where you either cannot do everything you need through Visual Studio Code, or where the Linux host is not even running on the same machine that you will debug from.

This doesn't prevent you from debugging the code running in Linux, but there are an extra couple of hoops to jump through. We will look at using Visual Studio 2019 and connecting over SSH, which is a common protocol for remote connections to a Linux system.

We can still test using WSL2, so in this case we will still connect to our local machine. It is possible to do a similar setup for other Linux distributions. The following instructions are for enabling SSH on the Ubuntu 20.04 that we have already set up:

1. Enable the SSH server:

sudo apt-get install openssh-server unzip curl

1. Edit sshd\_config to allow a password login:

sudo vi /etc/ssh/sshd\_config

1. Find the line PasswordAuthentication no and change it to #PasswordAuthentication no. (Press Insert to allow editing.)
2. Exit vi by pressing **Esc**, followed by entering **:wq**.
3. Start the ssh service:

sudo service ssh restart

1. To check the IP address of the Ubuntu installation that we are using, use the command ip addr. This is the one found attached to inet. In Figure 2.43, it is 172.28.88.220:



1. Test that you can connect to the SSH server with the Windows 10 SSH client. See next figure. The SSH client is an optional feature in Windows, so make sure you have installed it. Then, enter the following command, either from PowerShell or from the command line:

ssh user@ipaddress

Here's how the output looks like:



Notice that the first line in the screenshot shows a Windows prompt (C:\), whereas the last line shows an Ubuntu shell (andreas@AH-BOOK).

Once this is in place, you can open Visual Studio 2019 and connect to our code:

1. To start the app you want to debug, open the Linux instance in Windows Terminal and run dotnet run inside the correct folder—in our example, /mnt/c/Code/Book/Chapter\_02\_Workers
2. Make sure it runs without any issues, and then open the same solution in Visual Studio 2019.
3. Press Ctrl+Alt+P to open the Attach Process window.
4. Select SSH as the Connection type.
5. Connect to the same SSH server as when we were testing it. Connect to user@ipaddress. Refer to Figure below as an example of the username and IP address:



1. You will be prompted to enter your password as well, and if things work you should see a list of running processes. See the following screenshot:



1. Locate dotnet run and click **Attach.**
2. If everything went to plan, you should be able to hit breakpoints, read variables, output, and so on, directly from Visual Studio 2019 on Windows.

**Windows Firewall**

The first time you open the remote debug dropdown (after opening the Attach to Process window), you will be prompted to allow the connections through the Windows Firewall. Accept this to allow the debugger to establish connectivity.

In this case, the Linux instance was running on WSL2, but Visual Studio 2019 does not recognize this as a special case, so it doesn't matter if you attach to a different host. This may not be as simple as Visual Studio Code, but it is useful for the use cases where you need to do more complicated things.