**Remote Debugging Docker image hosted in Linux Container via SSH**

**Activate the docker internal ip address to external world.**

First we need to expose the internal docker ip’s to the external world via Mobylinux VM (Docker’s Default VM in windows Machine).

That is simple just type in the following command.

route /P add 172.20.128.2 MASK 255.0.0.0 10.0.75.2

**Prerequisites**

* Visual Studio 2017 is installed & updated to the most recent version
* Docker for Windows is installed
* Linux containers are enabled (right-click system tray icon -> Switch to Linux containers)
* PuTTYgen.exe is downloaded (used to generate SSH keys)

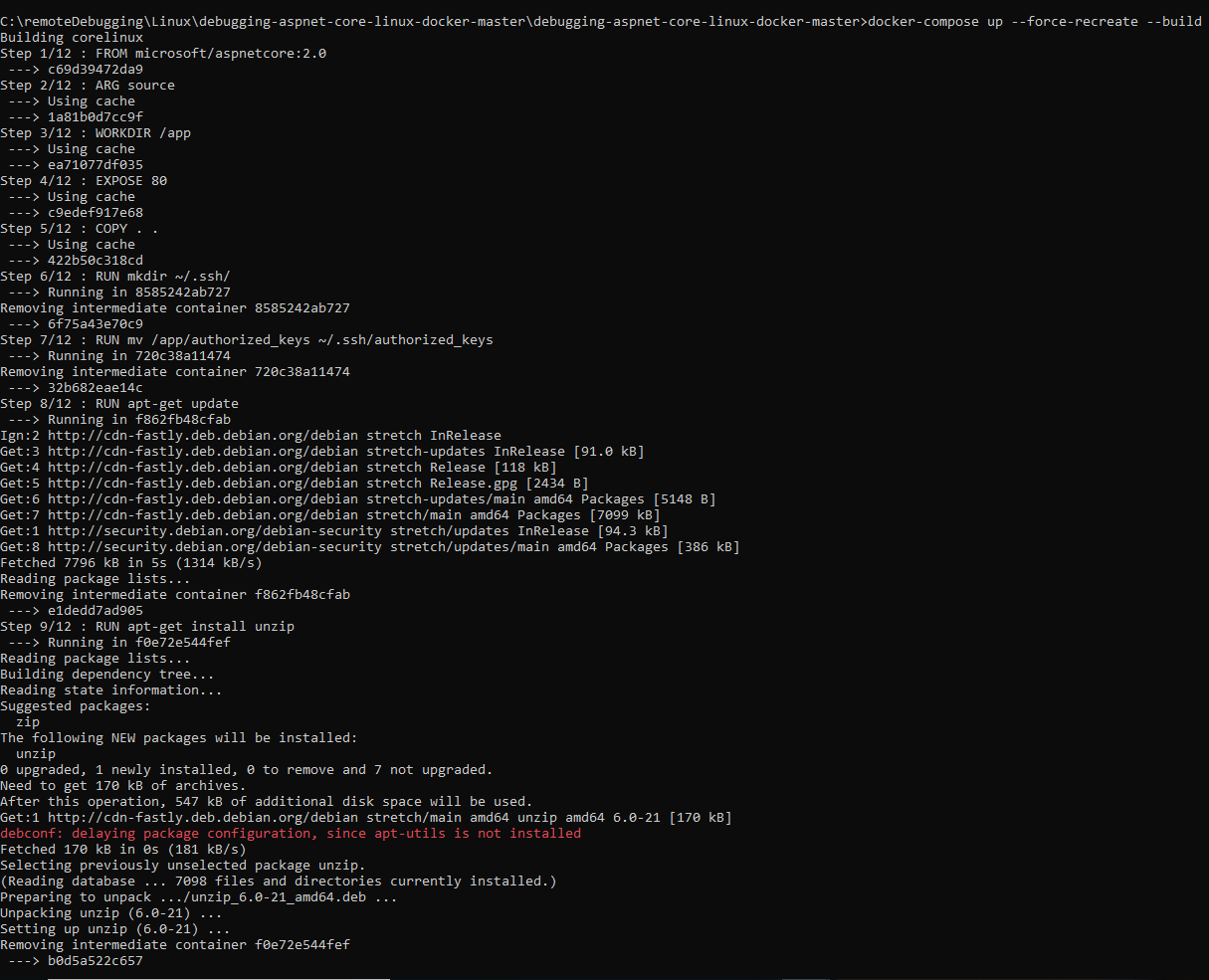
**1 - Generate SSH key**

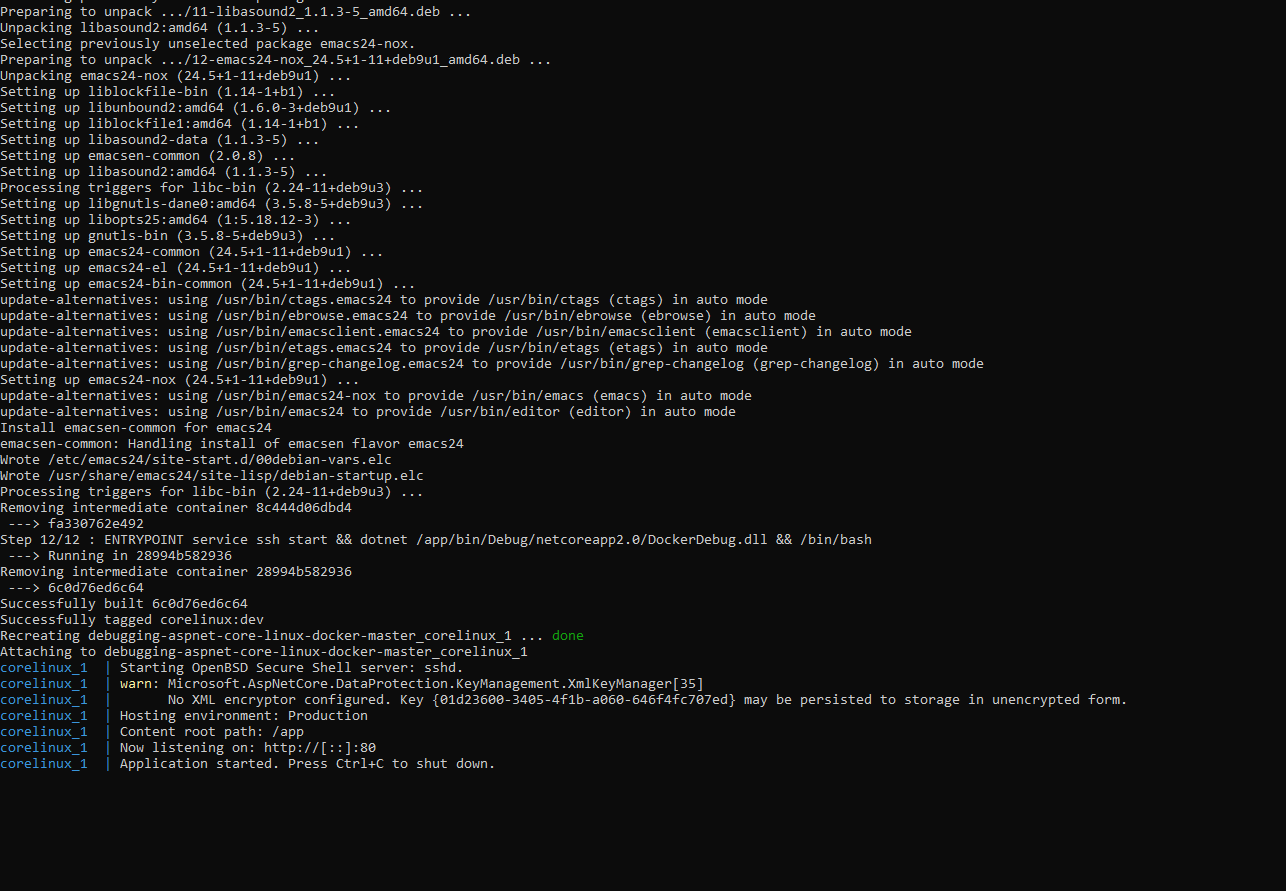
* Open PuTTYgen.exe
* Select SSH-2 RSA and 2048 bits
* Click Generate and follow instructions to generate a key
* Clone this repository
* Copy the public key (entire contents of the textarea) into DockerDebug/authorized\_keys
* Conversions -> Export OpenSSH key -> (no passphrase) -> overwrite DockerDebug/openssh\_privatekey

**2 - Starting dotnet & attaching the debugger**

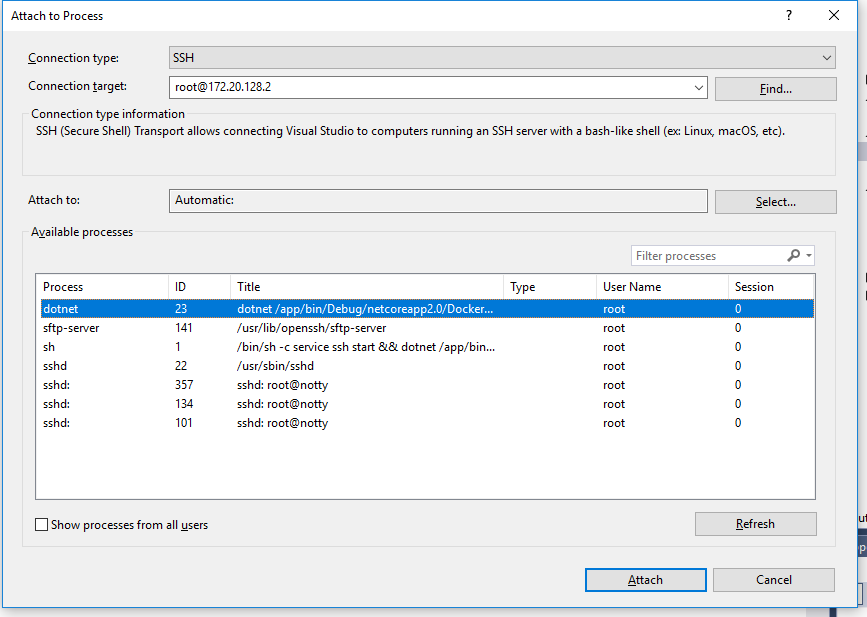
* Open Visual Studio 2017 in administrator mode and build the DockerDebug solution ( <https://github.com/riskfirst/debugging-aspnet-core-linux-docker> )
* Go to the project location where docker compose file is located and run the following command

docker-compose up --force-recreate –build

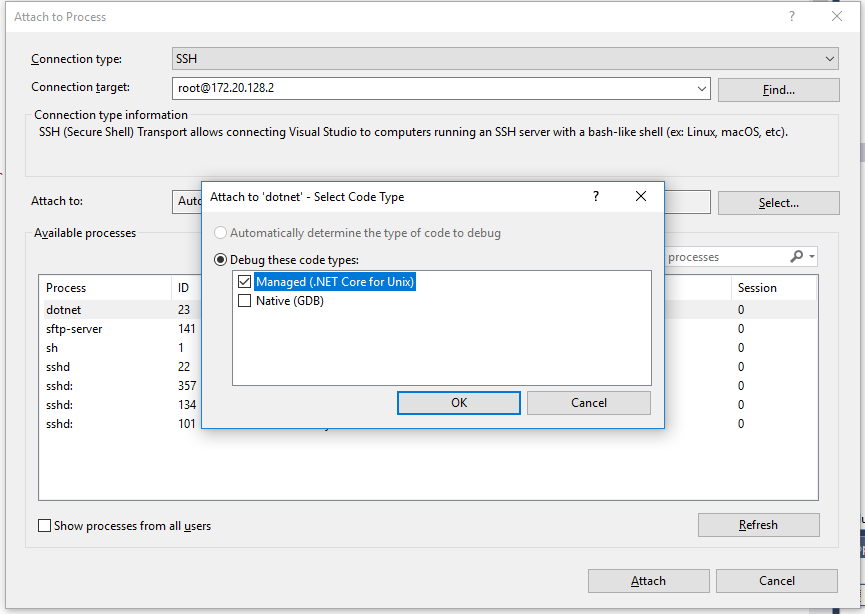




* Visit <http://172.20.128.2/api/test/get> to check that the site is running
* Attach to Process (ctrl+alt+p)
* Connection Type: SSH, Connection Target: 172.20.128.2 -> press enter
* Host name: 172.20.128.2, Port: 22, User name: root, Authentication type: Private Key, Private key file: browse to openssh\_privatekey
* Click Connect



* Attach to dotnet process, and select "Managed (.NET Core for Unix)"



Once done keep a break point on the test controller.

And browse the following url: <http://172.20.128.2/api/test/get>

And control will be available at breakpoint location.

