# Using the Vagrant VM

*Fall 2019*

## Contents



## 1 Using the Vagrant VM

Vagrant is a tool for automatically creating and configuring Virtual Machines (VMs) using simple scripts. Vagrant operates on a `Vagrantfile`, which contains instructions for how to download and run a particular VM image.

We have prepared a basic VM image for use in this class: it is simply an Ubuntu 18.04 installation with some common tools and libraries useful for this course. Since you will have root access, you can configure the VM as necessary to support your work—so long as we can still build your code on our VM.

This document is designed to familiarize you with using Vagrant and the VM to develop your work for this class.

**Note**: This is a new component of the course. Please feel free to ask questions or report any issues you encounter. We appreciate your feedback!

## 2 Installing Vagrant

Vagrant does not run VMs by itself: it requires a "provider" on the host machine to start and manage the VM. For our development VM, we will be using VirtualBox, a very popular open-source

VM provider that works well on most platforms. If you would like to use another provider (such as libvirt, Hyper-V, or Parallels), see Section 5.4 for details.

Before you can run the VM, you will need to install two components on your machine:

1. VirtualBox: `https://virtualbox.org`
2. Vagrant: `https://vagrantup.com`

If you use Linux, the versions of these tools provided in your package manager should be sufficient.

## 3 Installing the VM

1. Download the Vagrantfile for our VM, available here:
   `https://cs.brown.edu/courses/csci1680/f19/content/Vagrantfile`
   You should place this file in its own directory, such as `~/cs168/vagrant`. If you are using Windows, make sure you save the file with no extension (not as a text file).
2. Open up the Vagrant file and skim over the options. This file lets you configure various settings about how the VM runs, including directories shared between the host and the VM, network interfaces, and more. You do not need to change anything now, but feel free to edit this file later.
3. In a terminal, enter this directory and run:
   `vagrant up`
   This should cause Vagrant to download, install, and start the VM. This may take a few minutes, depending on your network.
4. Once the VM has started, run `vagrant ssh` to get a shell!

If you encounter any issues with this process, please feel free to see the course staff for help.

The VM will remain running until you shut it down. To stop the VM, run `vagrant halt`. You can also save snapshots of the VM's state, roll it back to its original version, and more—see `vagrant help` for details.

## 4 Working in the VM

You can work in the VM just like any other Linux system—you can edit files, install packages, etc. The default user on the VM is `vagrant`, with password `vagrant`. You should not need this information often, as password-free sudo access is enabled.

The directory `/vagrant` in the VM is automatically mounted to the directory on the host where the Vagrantfile is located. We recommend that you clone your project repositories in this directory to share them with the VM—this way, you can edit your work on either the VM or your host OS.

# 5 Tips and Tricks

## 5.1 X Forwarding

**Note**: If you use Windows, it may be easier to install a GUI in the VM than configuring your system for X11 forwarding. See Section 5.3 for more information.

Vagrant does not enable X forwarding for SSH connections by default. To run GUI applications, connect to the VM with:

```
vagrant ssh -- -X
```

This passes the -X argument when Vagrant connects to the VM. Once the connection completes, you should be able to run GUI applications as usual.

To streamline this process, you can add an entry to your `~/.ssh/config` to always use X11 forwarding. To start, you can get Vagrant to output a template ssh configuration with `vagrant ssh-config`.

## 5.2 Using Wireshark

One of the most important features of the development VM (compared to the department machines) is that you can run Wireshark to debug your traffic.

To run wireshark, SSH to the VM *with X11 forwarding* (see above) and run `wireshark`.

To start a capture, select **Capture** > **Options** from the menus to select the capture interface. If the program you are testing is connecting to localhost, you should capture on the Loopback interface (`lo`), which is used for local traffic.

When capturing on the Loopback interface, you will likely see a lot of X11 protocol packets. These packets are actually caused by Wireshark itself—they represent X11 protocol messages to update Wireshark's GUI.

Since capturing the X11 traffic is very useful for us (and will use a lot of memory), you can tell Wireshark to ignore these packets using a *capture filter*. When starting a capture, you can enter the capture filter "`not tcp port 6010`", which will tell Wireshark to ignore this traffic.

## 5.3 Using a GUI

The default configuration for the VM is to access it in "headless mode" using SSH. If you prefer to use the VM with a full desktop environment, you can install one easily:

First, install the necessary packages:

```
sudo apt-get install --no-install-recommends \
    ubuntu-desktop gnome-terminal virtualbox-guest-dkms
```

After the installation finishes, shut down the VM (`vagrant halt`) and edit the Vagrantfile to set `vb.gui` to true. The next time you start the VM with `vagrant up`, it will open a VirtualBox

window and start the VM in GUI mode. Note that the VM may take a few minutes to boot the first time you start in this mode.

### 5.4 Using your own VM provider

If you would prefer to use another VM provider (such as libvirt, Hyper-V, or parallels, you are welcome to do so, but support from the course staff will be limited.

The Vagrantfile we have provided is specific to Virtualbox. However, you can create your own VM from a starter image that is compatible with any provider. To do this, use the following Vagrantfile: `https://cs.brown.edu/courses/csci1680/f18/content/Vagrantfile-generic`

This Vagrantfile downloads a basic Ubuntu image that is not specific to any VM provider, and then installs all of the required packages (in fact, we used a similar script to create the standard VM). You can start this VM in the same way as the instructions above—however, the initial setup process will take longer in order to install the required packages.

## 6 Getting Help

If you have questions on using or configuring the VM, please do not hesitate the course staff. In addition, if you run into any issues, please let us know so that we can update our documentation. This is a new component of the course—we appreciate your feedback!

---

Please let us know if you find any mistakes, inconsistencies, or confusing language in this or any other CS168 document by filling out the anonymous feedback form:

`https://piazza.com/brown/fall2019/csci1680`.