Chapter 1: Authentication

Installing Vagrant

Download Course Materials

Handouts (1)

m310-vagrant-env.zip

Throughout this course we'll be using Vagrant as our environment for homework exercises.

Vagrant enables us to have a consistent deployment environment for all students. This makes validating the correctness of a homework solution a lot easier. It will also aid you when asking and answering questions on the discussion forum as you know all of your peers have a near identical environment.

Vagrant requires an hypervisor or virtual environment provider to operate. For this purpose we use VirtualBox.

Below you'll find instructions on how to install Vagrant and VirtualBox on Windows, OSX and Linux.

General Notes

- The installation of the Vagrant environment takes, in a fast network (49Mbps download, 194Mbps upload), around 2:33 minutes to complete. Account for a considerably longer time on slower networks!
- The video instructions might not reflect the exact commands required for your workstation operating system.
- The Vagrant environment files are made available as a downloadable zip file.
- Windows OS 7 and 32-bit systems are not supported.

Installing Vagrant on OSX or Linux

1. Install VirtualBox

Download and installation should be straightforward. Go to the VirtualBox downloads page, download the setup binary for your appropriate OS, and run the installer.

If you run into issues installing VirtualBox on a recent version of MacOS, you may want to look at this knowledge article.

2. Install Vagrant

After VirtualBox has been installed you can go ahead and download and install Vagrant.

Installing Vagrant on Windows

1. Install VirtualBox

Download and installation should be pretty straightforward. Go to the VirtualBox downloads page, download the setup binary for Windows hosts, and run the installer.

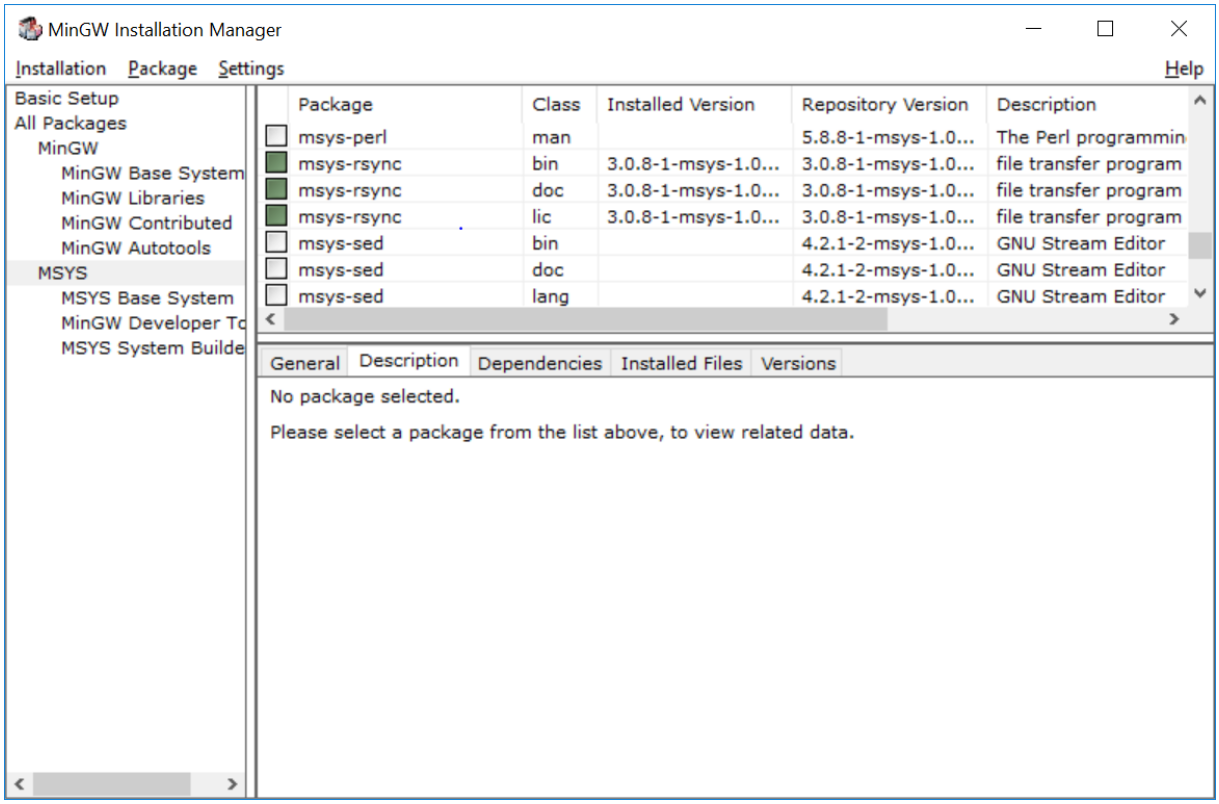
2. Install MinGW

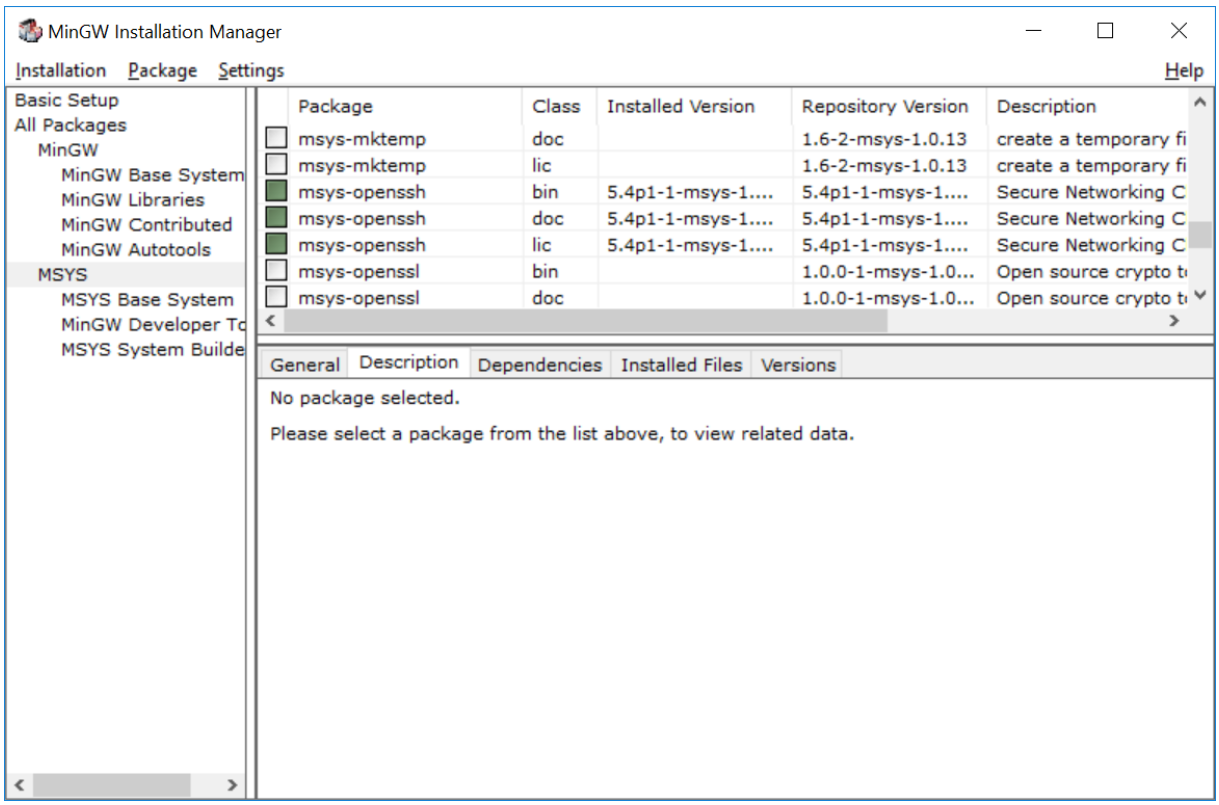
Vagrant requires two important command-line tools: `rsync` and `ssh`. These command-line tools allow us to connect to our virtual machine and to sync files between our host and virtual machines. MinGW is a set of GNU utilities for software development on Windows. We'll use MinGW to install both `rysnc` and `ssh` onto our Windows machine.

You should download and run the MinGW Installer (mingw-get-setup.exe).

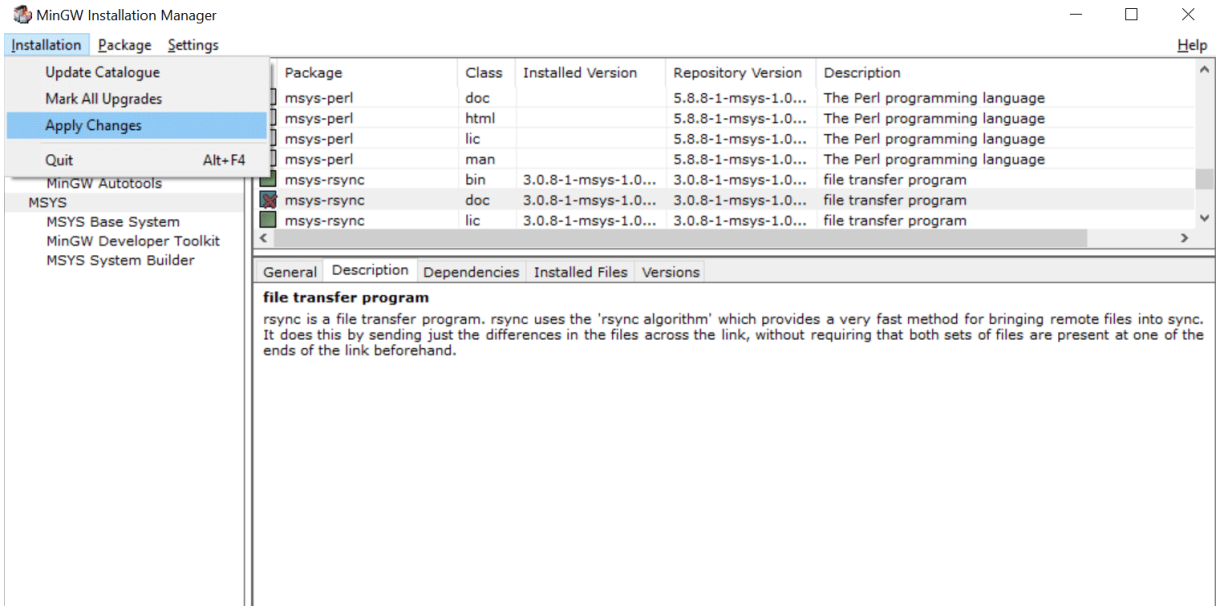
3. Install rsync and ssh

The last step of the installation of MinGW is to specify the packages you want to install. From here you can mark `rsync` and `ssh` for installation like so:



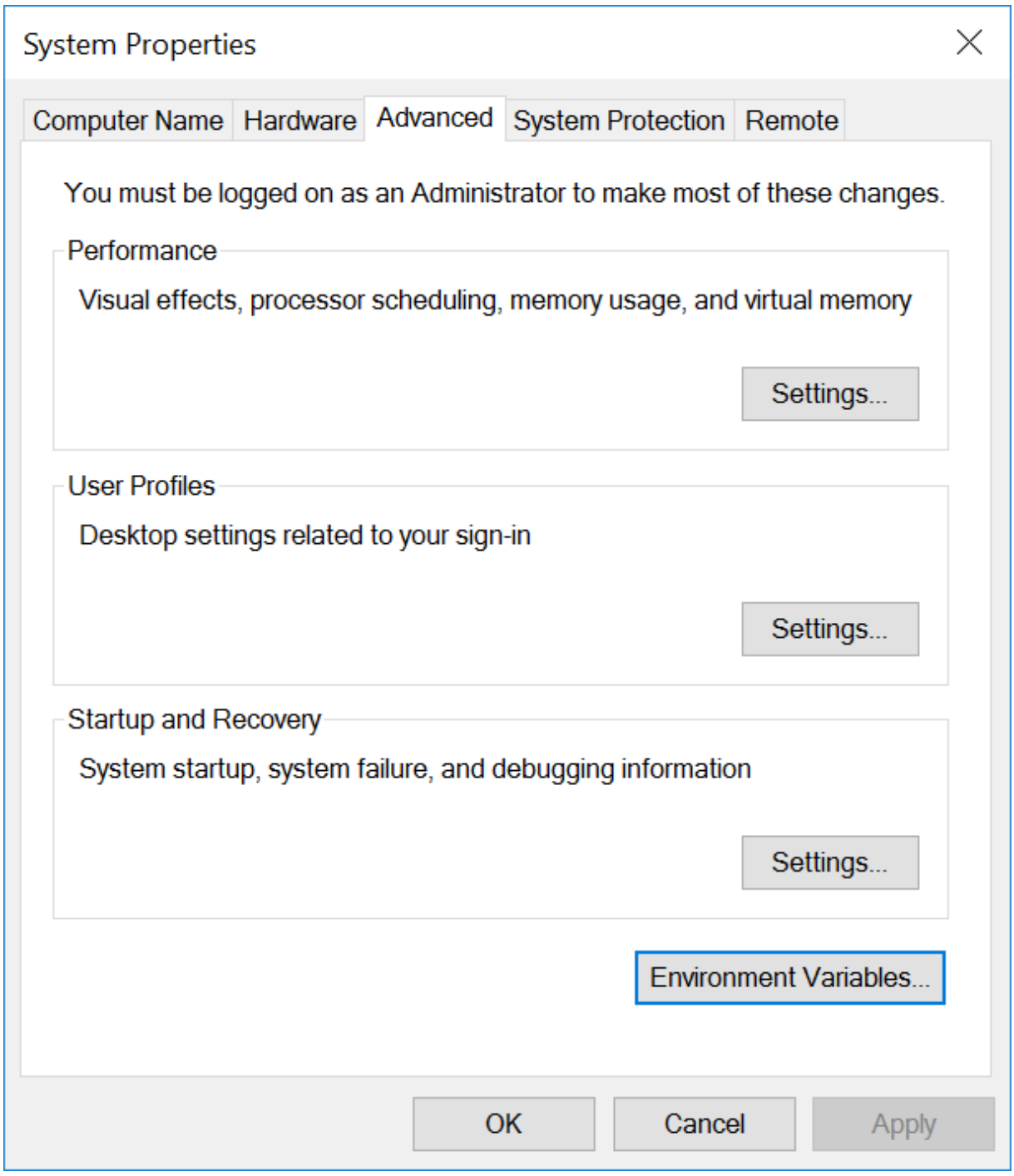


After marking the packages for installation you can go ahead and apply the changes:

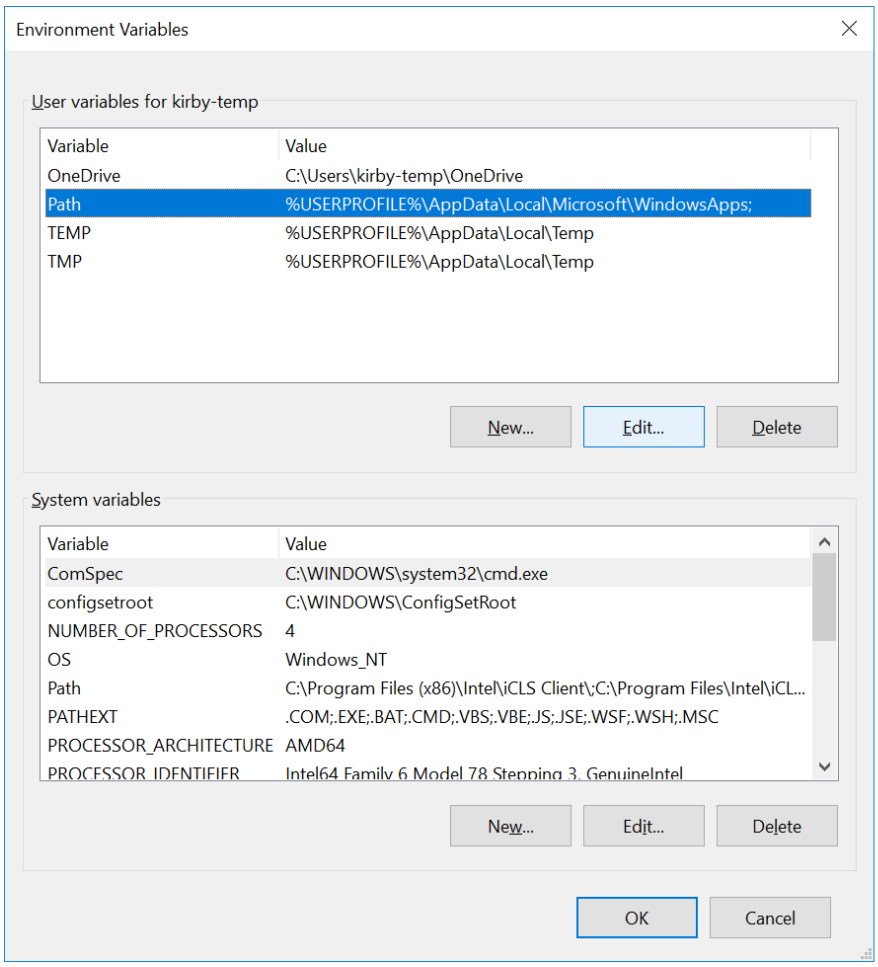


Assuming you've kept the default installation path `rsync` and `ssh` should be installed to the `C:\MinGW\msys\1.0\bin` directory. In order for these tools to be accessible via the command-line (and Vagrant) you'll need to add this directory to your PATH environment variable.

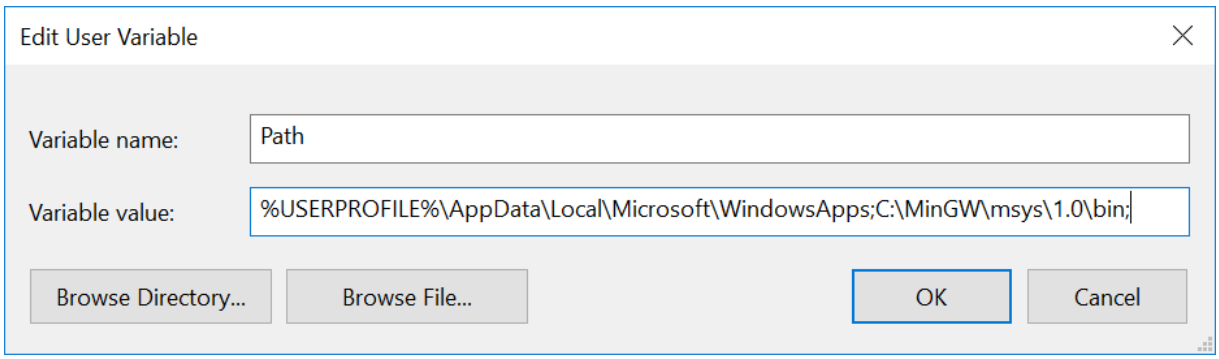
First go to Advanced system settings under System Properties and click "Environment Variables...":



From there select the "Path" variable and click "Edit...".



Directories in the PATH environment variable are deliminated by semicolons. You can now append the `C:\MinGW\msys\1.0\bin` directory to the end of the "Variable value."



You can confirm that `rsync` and `ssh` are both successfully installed and accessible via your PATH by launching the command prompt (`cmd`) and running `rsync` and `ssh` respectively. Both commands should output usage information. If you receive an error about not being recognized as an internal or external command then you'll need revisit the steps above.

4. Install Vagrant

After all of the above prerequisites have been installed you can go ahead and download and install Vagrant. This will require a restart of your machine.

5. Spin up the M310 Environments

   We can now bring up our Vagrant machines. This course provides you with two VMs: `database` and `infrastructure`. You'll spend most of your time using the `database` VM, but for certain labs you'll use both the `database` and `infrastructure` VMs.

   Go ahead and download the attached handout which contains the `Vagrantfile` and the associated provisioning scripts. After extracting the zip file you can run the following commands to setup the VMs.

   ```
   $ cd m310-vagrant-env
   $ vagrant plugin install vagrant-vbguest
   $ vagrant up
   ```

   After `vagrant up` exits successfully you'll have two VMs provisioned and up and running. You can confirm this by running `vagrant status`.

   You'll notice that in spinning up these VMs a new **shared** folder was created. This folder is automatically synced with both of your VMs. You'll use this as a mechanism to bring different files and scripts into your virtual environments.

   Let's go ahead confirm that each VM is up and running and that the **shared** folder is being synced properly. Create a temporary file on your host machine's **shared** folder, connect to each of the VMs, and confirm that the file was properly synced.

   ```
   $ echo hello >> shared/test.txt
   $ vagrant ssh database
   $ cat ~/shared/test.txt
   $ exit
   $ vagrant ssh infrastructure
   $ cat ~/shared/test.txt
   ```

   Once you've confirmed that both machines are up and running and that files are seamlessly synced between your host machine and the VMs you're all set to complete the homework!

Troubleshooting

- Occasionally Vagrant will fail to sync files in the **shared** directory. If this happens you should perform the following steps from the host machine:

  ```
  ```

```
$ vagrant plugin install vagrant-vbguest
$ vagrant reload <machine>
$ vagrant provision <machine>
```

You can then test if this worked in the same fashion explained in step 5.

- On some systems after `rsync` is installed it will require permissions to actually sync data between the host and the VM. This means that your first `vagrant up` might fail. If this is the case simply running `vagrant up` a second time should resolve the issue.
- If you have another set of Unix utilities for Windows, those may conflict with MinGW. This will result in an `rsync` error that says "Error: dup() in/out/err failed". In these circumstances you should move the `C:\MinGW\msys\1.0\bin` directory to the beginning of your PATH.

Proceed to next section