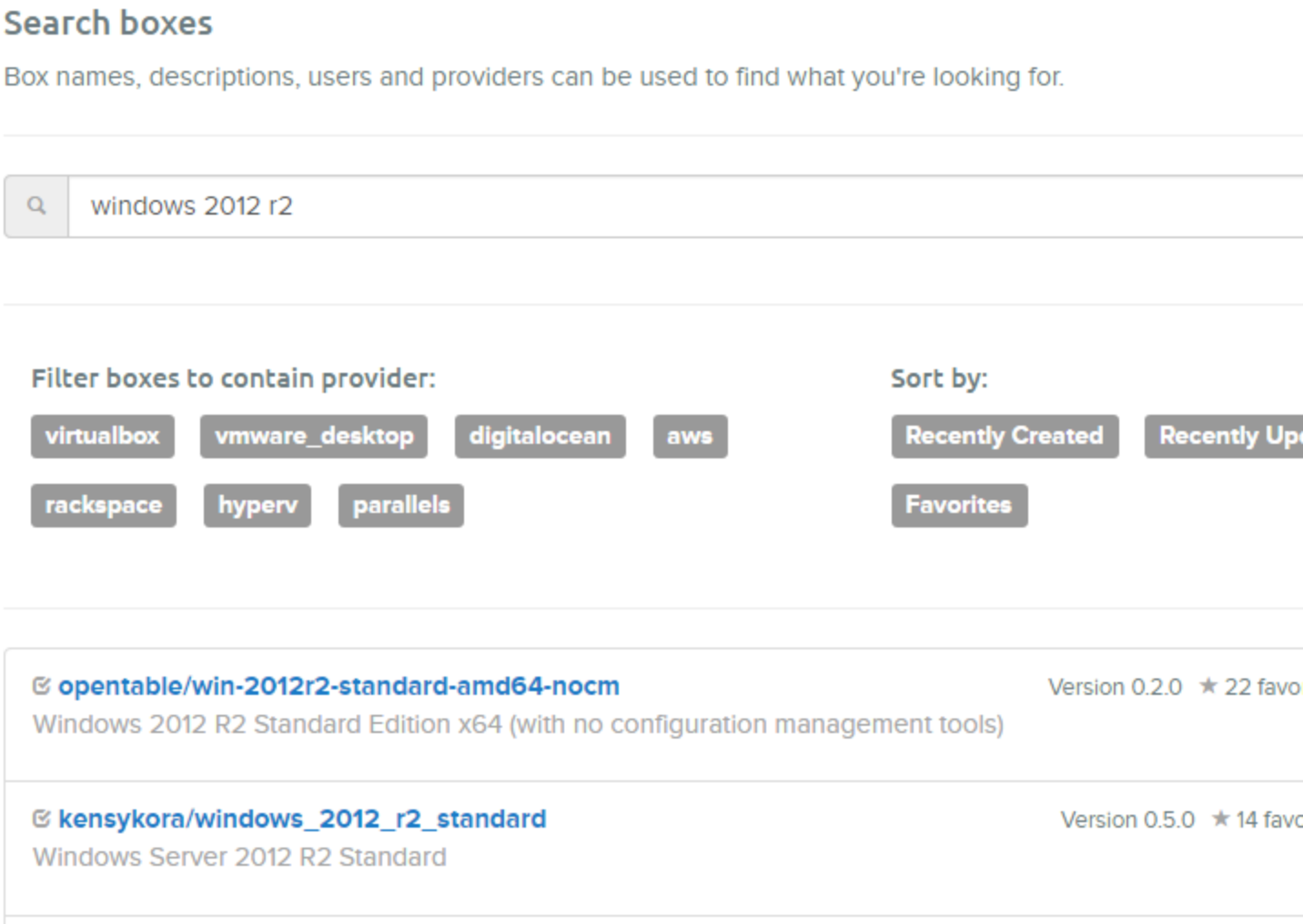
Lab 7 Setup Windows Chef with Vagrant

Nonetheless, the selection of "Boxes" (VM templates) on vagrantcloud.com is pretty limited right now, presumably due to licensing concerns.

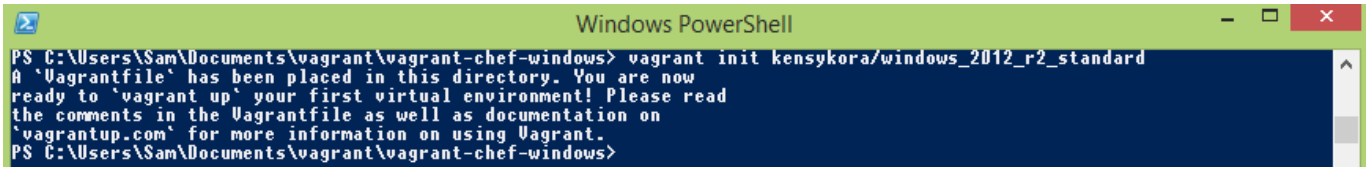


The most popular Windows 2012 R2 box is currently one provided by OpenTable, however it seems to have issues with password expiry, so, we'll go with the *second* most popular, the one by kensykora.

If you open up the link to that box, you'll see a handy command in a textbox, ready for you to copy out.

Copy that command, open a new PowerShell window on your computer, create a new folder in your My Documents called “vagrant-chef-windows”, then execute the command:

```
vagrant init kensykora/windows_2012_r2_standard
```



This creates a *Vagrantfile* in the directory in which you’ve executed the command.

1) Setup Initial Vagrant Configuration

Open the Vagrantfile in your favorite text editor, and replace the contents with the following:

```
VAGRANTFILE_API_VERSION = '2'
Vagrant.configure(VAGRANTFILE_API_VERSION) do |config|
  # Every Vagrant virtual environment requires a box to build off of.
  config.vm.box = 'kensykora/windows_2012_r2_standard'

  # Forward ports
  config.vm.network 'forwarded_port', guest: 80, host: 8080

  config.vm.provider 'virtualbox' do |vb|
    # Don't boot with headless mode
    vb.gui = true
  end

  # Shell Provisioning
  config.vm.provision 'shell' do |shell|
    shell.path = 'install-chef.ps1'
  end
end
```

The configuration file is Ruby based, and does several things.

1. Provisions the VM based on kensykora/windows*2012*r2_standard (downloading it if necessary)
2. Forwards port 80 in the guest machine to port 8080 on your machine (the host)
3. Pops up a Virtualbox window with the guest's console for simplicity's sake
4. Executes install-chef.ps1 in the guest

Take a few moments to pair up the list above with the lines in the configuration file, once you have, you'll wonder "where the hell is it getting install-chef.ps1 from?". At the moment, it isn't.

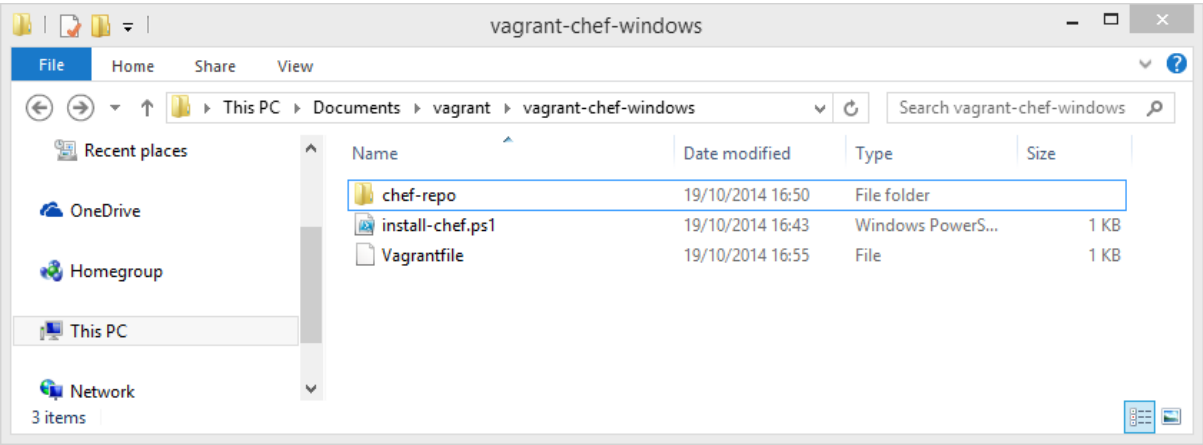
2) Use PowerShell Bootstrapping to Install Chef

Create a new file in your *vagrant-chef-windows* directory called *install-chef.ps1 *and populate it with the following:

```
$progressPreference = 'silentlyContinue';
$chefInstaller = 'C:vagrantchef-windows-11.16.2-1.windows.msi';
$chefInstallerUri = "https://opscode-omnibus-
packages.s3.amazonaws.com/windows/2008r2/x86_64/chef-windows-11.16.2-
1.windows.msi";

if(!(test-path $chefInstaller)){
  Write-Host "$(Get-Date) Downloading Chef...";
  Invoke-WebRequest -Uri $chefInstallerUri -outfile $chefInstaller;
}

if(!(Test-Path "C:chef")){
  Write-Host " $(Get-Date) Installing Chef";
  Start-Process -Wait -FilePath 'C:\Windows\system32\msiexec.exe' -
ArgumentList @('-i',$chefInstaller,'/quiet','/log','C:\tmp\chef-client-
install.log')
  Write-Host " $(Get-Date) Installation Complete"
}else{
  Write-Host " $(Get-Date) Chef is already installed!";
}
```

Ideally, we wouldn't need to do this as Chef would already be installed in the Box we got from VagrantCloud.com, however, at the time of writing there are no Windows 2012 R2 boxes with Chef pre-installed.

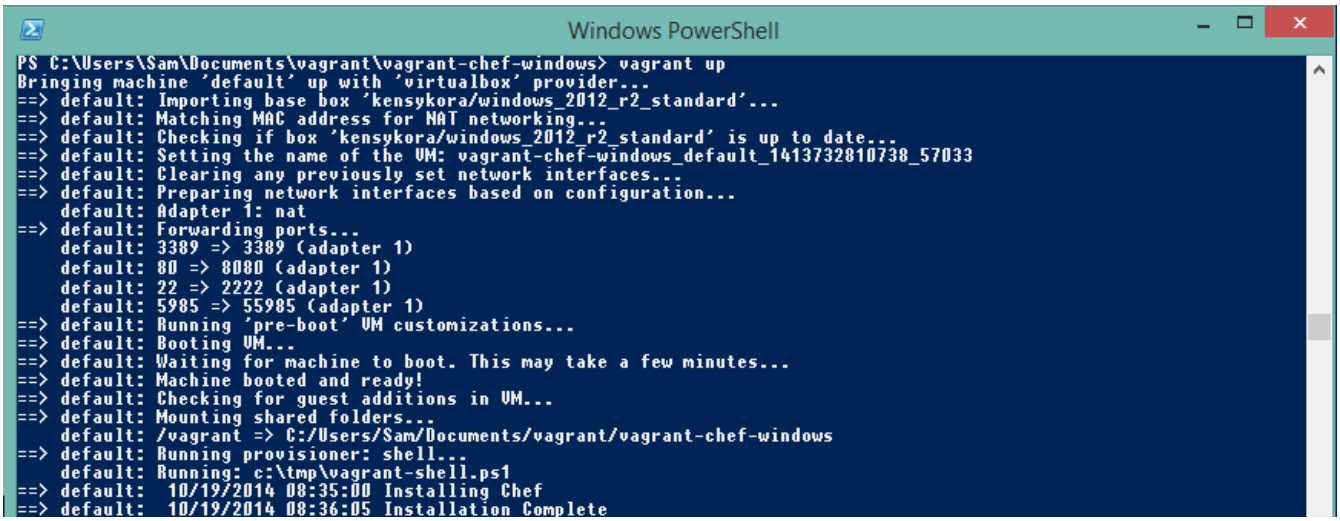
Your folder should now look like this:



3) Power On – Vagrant Up

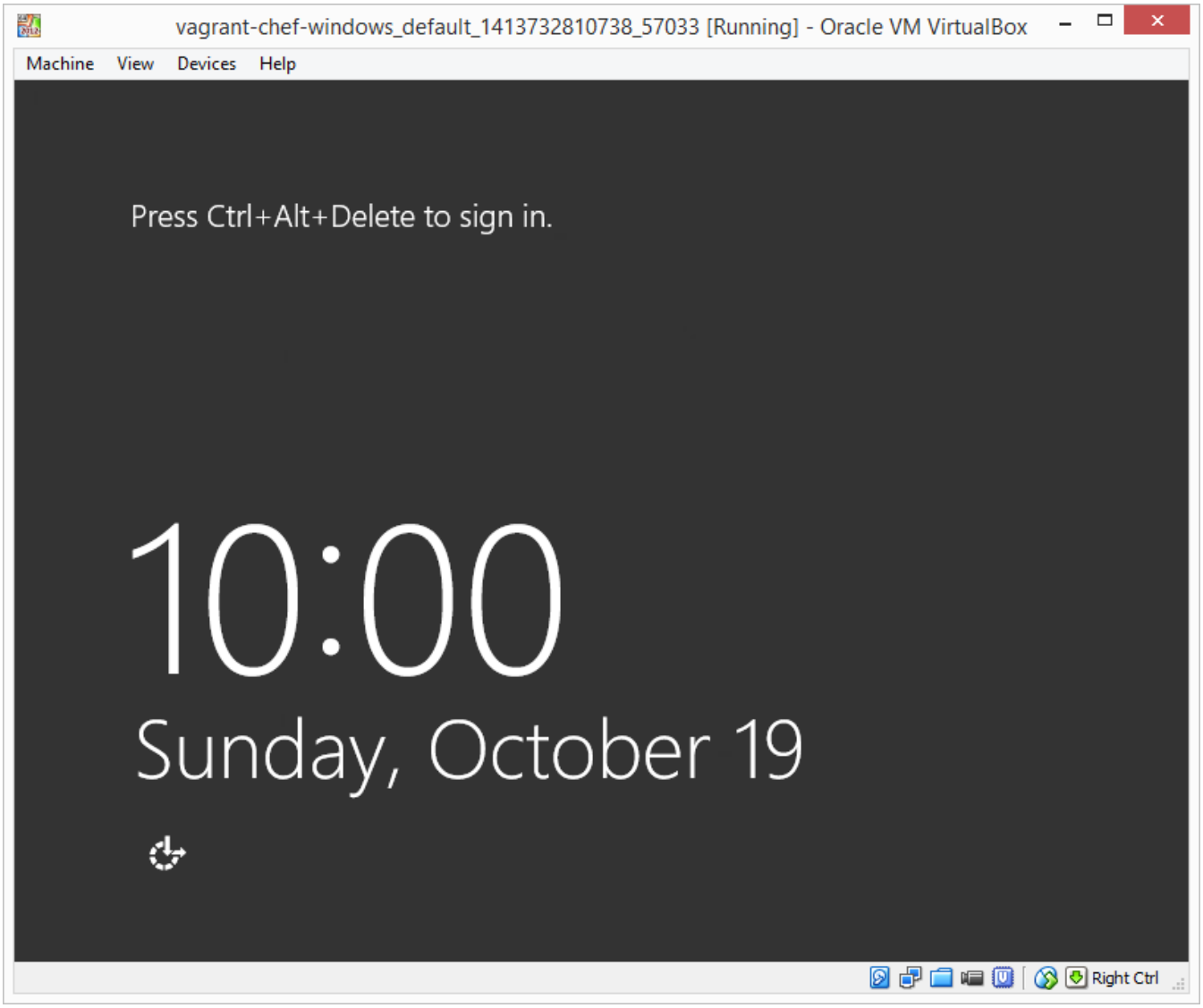
Now, ensure you're in your *vagrant-chef-windows* folder in the PowerShell console, then execute:

```
vagrant up
```

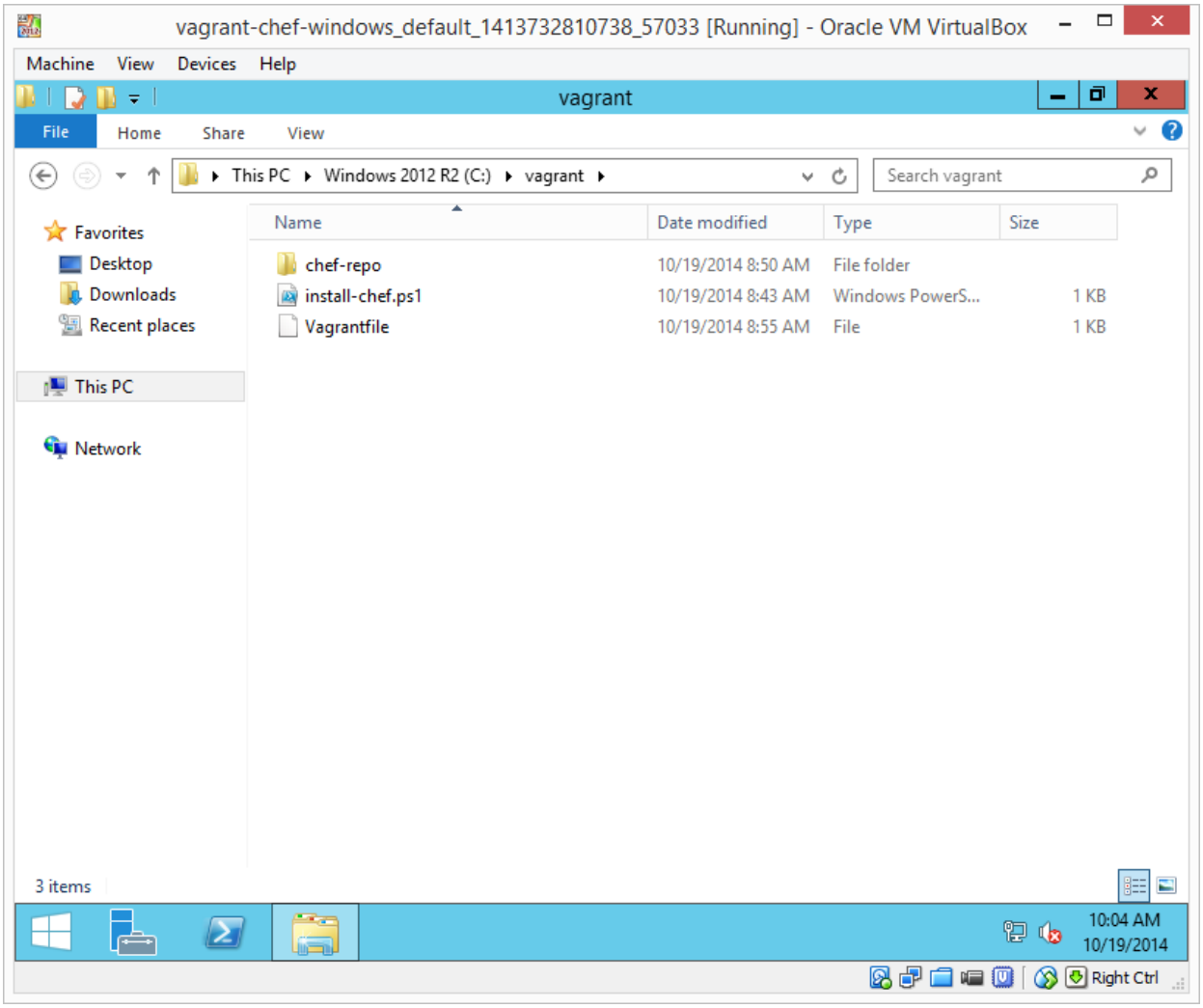


It will scurry off, download the kensykora 2012 R2 box (not shown as I already had it), power up a new VM and execute your ps1. Once complete, you should have a VirtualBox console pop up and allow you to sign in (right ctrl + del = Ctrl + Alt + Delete).

Username: Vagrant
Password: vagrant



If you login, you'll see *C:\chef* exists, and if you browse into *C:vagrant*, you'll see that the entirety of your *vagrant-chef-windows* folder is available within the VM!



This is **important** because almost all file paths you'll set in your Vagrantfile configuration will be relative to this directory.

4) Setup Vagrant Chef Provisioning Configuration

Now it's time to actually *use* Chef. But we're not going to just open up a PS console inside the VM and run chef-client. Oh no, we're going to use Vagrant's chef-client provisioning functionality!

That means that every time we deploy a new VM, our PS1 file will install Chef, then Vagrant will run chef-client for us, with the configuration we've defined in the Vagrantfile.

Add the following lines to the end of your Vagrantfile (but before the final "end").

```
# Chef Provisioning
config.vm.provision 'chef_client' do |chef|
  chef.chef_server_url = 'https://api.opscode.com/organizations/orgname'
  chef.node_name = 'node20141019'
```

```
  chef.validation_client_name = 'orgname-validator'
  chef.validation_key_path = "chef-repo\.chef\orgname-validator.pem"
  chef.add_recipe 'learn_chef_iis'
end
```

You will, of course, need to replace *orgname* with your organisation name on the highlighted lines, and amend the node_name if you like.

Your Vagrantfile should now look like this:

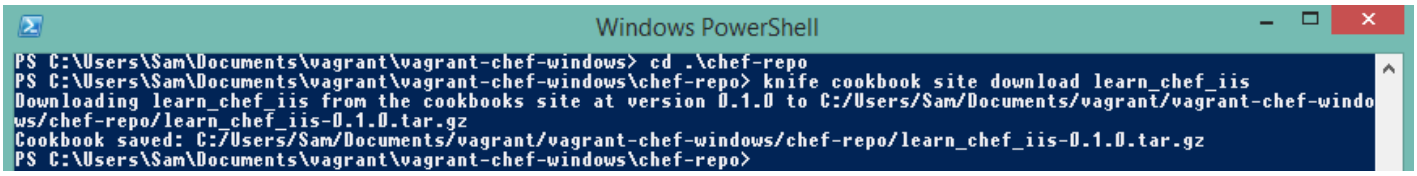
```
# Vagrantfile API/syntax version. Don't touch unless you know what you're doing!
VAGRANTFILE_API_VERSION = "2"

Vagrant.configure(VAGRANTFILE_API_VERSION) do |config|
    # Every Vagrant virtual environment requires a box to build off of.
    config.vm.box = "kensykora/windows_2012_r2_standard"

    # Forward ports
    config.vm.network "forwarded_port", guest: 80, host: 8080

    config.vm.provider "virtualbox" do |vb|
        # Don't boot with headless mode
        vb.gui = true
    end

    # Shell Provisioning
    config.vm.provision "shell" do |shell|
        shell.path = "install-chef.ps1"
    end

    # Chef Provisioning
    config.vm.provision "chef_client" do |chef|
     chef.chef_server_url = "https://api.opscode.com/organizations/samartin"
     chef.node_name = "node20141019"
     chef.validation_client_name = "samartin-validator"
     chef.validation_key_path = "chef-repo\\.chef\\samartin-validator.pem"
     chef.add_recipe "learn_chef_iis"
    end

end
```

This code uses the Chef Client we've already installed and the *orgname-validator.pem* which came with our Starter Kit in order to add this guest as a node to our managed Chef environment.

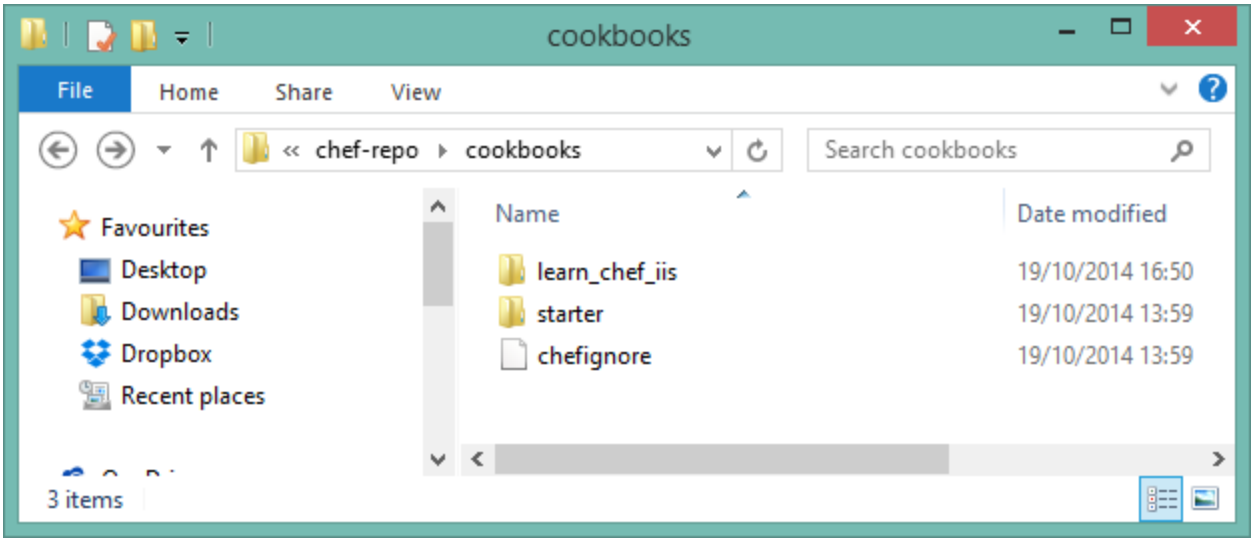
5) Upload the Cookbook

But wait, we haven't got the cookbook *learn_chef_iis* (a simple Windows/IIS example used by the learnchef.com/windows walkthroughs)! CD into your *chef-repo* directory and execute:

```
knife cookbook site download learn_chef_iis
```

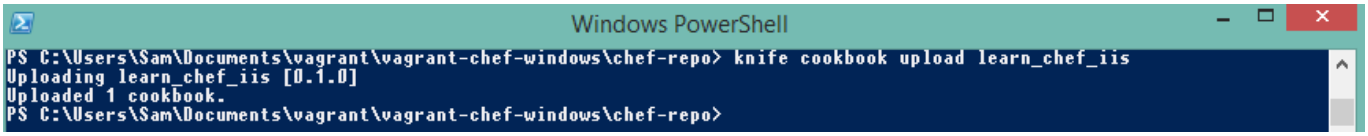


Now extract the resulting tar.gz into your cookbooks subdir.



And finally, upload it to your managed Chef environment.

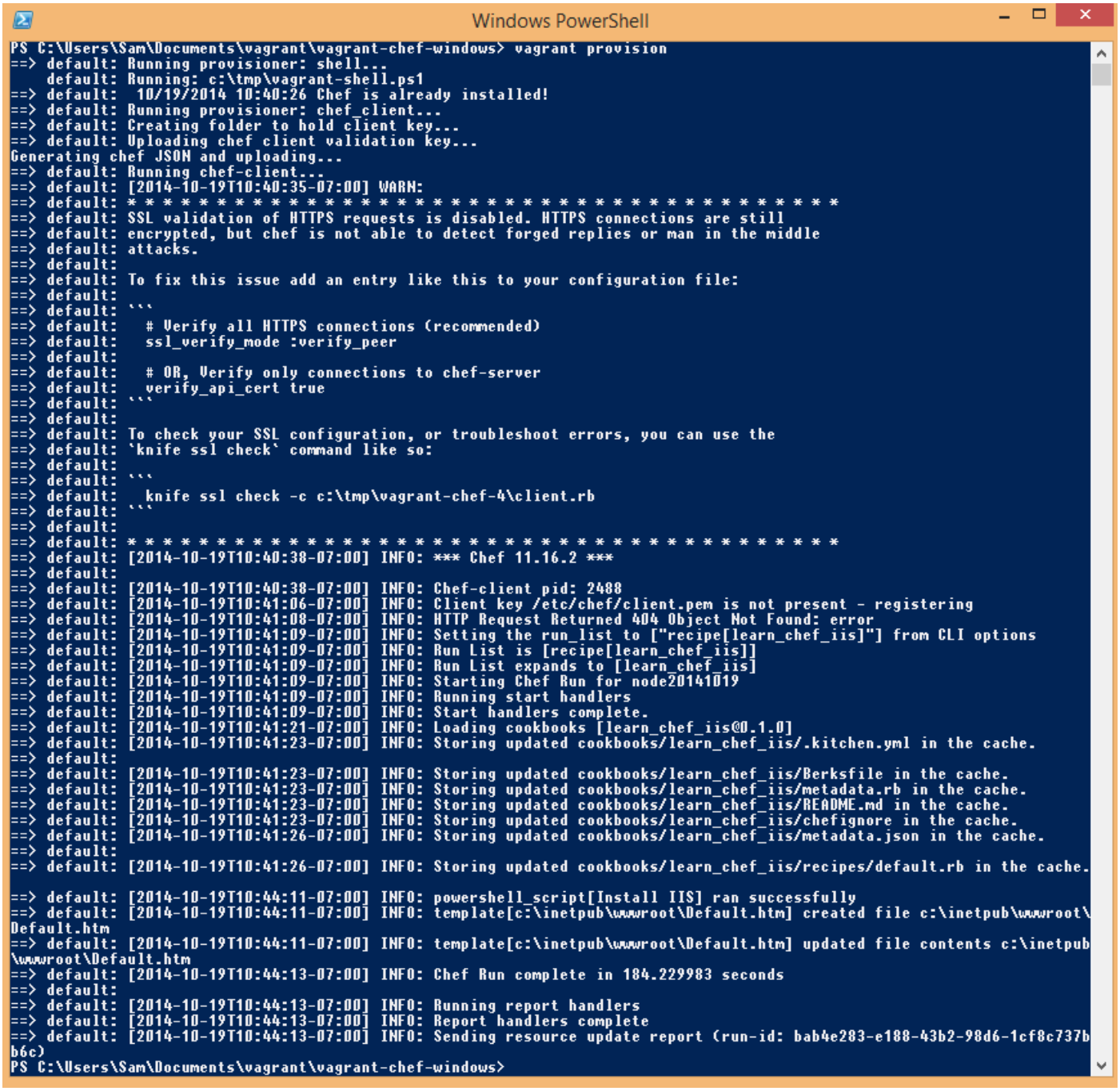
```
knife upload learn_chef_iis
```



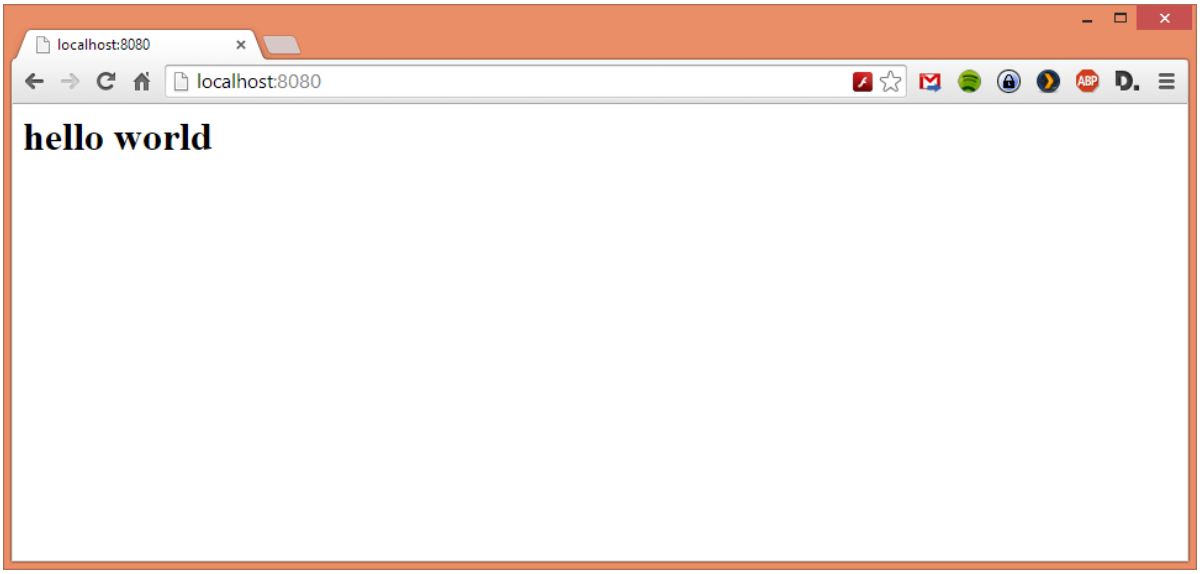
6) Vagrant Provision

Excellent! The cookbook's ready to go. Now CD up a level into your vagrant directory and run:

```
vagrant provision
```



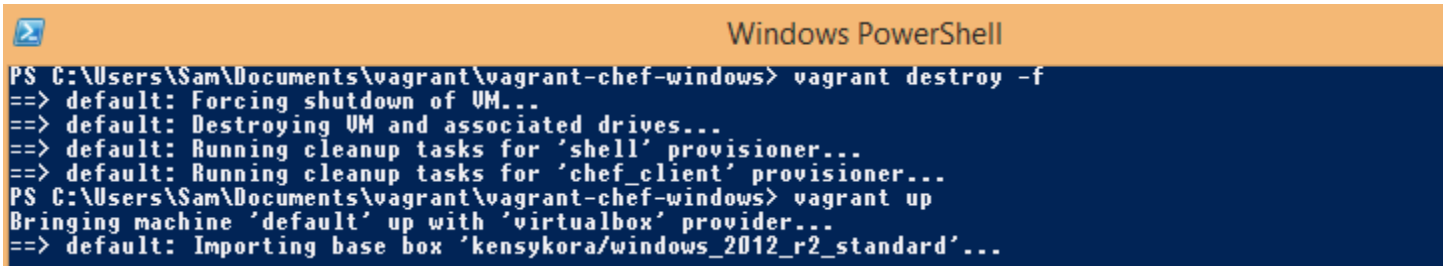
Vagrant has now kicked off a chef-client run with the learn*chef*iis cookbook as its runlist. Once it's finished (and in combination with the forwarded port we setup earlier) you should now be able to open your favourite browser on your host machine and go to http://localhost:8080 and see…



7) Redeploy from Scratch

Now for the moment of truth. Delete the node from the managed Chef environment, destroy the VM and redeploy a fresh one based on the configuration we've provided!


```
vagrant destroy -f vagrant up
```



Wait a little while for Vagrant and Chef to finish doing their thing and you should be able to go back to localhost:8080 again and see exactly the same thing on a fresh VM!