In today’s post I’m going to show you how to create a local Chef development environment that can be used to provision a simple cookbook to a Docker Centos7 instance. This environment is based on a VMWare Workstation vm created from a Centos 7 ISO image.

To install Docker, log on to the newly created vm as root and run the following commands:

$ sudo yum update

$ cat >/etc/yum.repos.d/docker.repo <<-EOF

[dockerrepo]

name=Docker Repository

baseurl=https://yum.dockerproject.org/repo/main/centos/7

enabled=1

gpgcheck=1

gpgkey=https://yum.dockerproject.org/gpg

EOF

$ yum install docker-engine

$ chkconfig docker on

$ service docker start

Create a linux group named *docker* and add your user to it, this will enable you to run Docker commands without using sudo

$ usermod -aG docker afryer

Install wget and curl packages

$ yum install wget

$ yum install curl-devel

Download the latest ChefDK and install

$ wget https://opscode-omnibus-packages.s3.amazonaws.com/el/7/x86\_64/chefdk-0.10.0-1.el7.x86\_64.rpm

$ rpm -Uvh chefdk-0.10.0-1.el7.x86\_64.rpm

Install the kitchen-docker ruby gem, switch to the user you will use for developing the Chef Cookbook and execute the following:

$ chef gem install kitchen-docker

**Creating the Application Cookbook**

Let’s create a simple application cookbook for a simple webserver configuration that uses the Apache 2 cookbook from the Chef Supermarket to install Apache.

Switch to the user that you will use for creating the application cookbook and create the folder ~/chef-repo. This will be used as the local Chef repository for the source files for the application cookbook.

~$ mkdir-p ~/chef-repo

The chef executable is a command-line utility that comes as part of the ChefDk we will use this to generate the Chef Source code for the Application Cookbook.

Create the c2b2\_website application cookbook by executing the following commands:

$ cd ~/chef-repo

$ chef generate app c2b2\_website

This generates the following folder structure which includes a top level Test Kitchen instance for testing the cookbook.

/c2b2\_website

   /.git

   /cookbooks

      /c2b2\_website

        /recipes

**default**.rb

        /spec

           /unit

              /recipes

                 default\_spec.rb

           spec\_helper.rb

        Berksfile

        chefignore

        metadata.rb

   /test

     /integration

        /**default**

           /server\_spec

              default\_spec.rb

           /helpers

              /server\_spec

                 spec\_helper.rb

   .gitignore

    README.md

  .kitchen.yml

Let’s create a new recipe for the application cookbook named *intsallapache*, this will reference the appropriate recipes in the Apache2 cookbook to install Apache.

First we need to set up the dependency to this cookbook in the metadata.rb file for the c2b2\_websitecookbook, add the following to the file~/chef-repo/c2b2\_website/cookbook/c2b2\_website/metadata.rb:

***depends'apache2'***

Test Kitchen uses Berkshelf for cookbook dependency management, so to run the integration tests we need to create the configuration file in the folder ~/chef-repo/c2b2\_website/Berksfile and reference the Apache2 cookbook and the c2b2\_website cookbook.

Create the file~/chef-repo/c2b2\_website/Berksfile and add the following content:

source 'https://supermarket.chef.io'

cookbook 'apache2', '~> 3.1.0'

Dir['/home/username/chef-repo/c2b2\_website/cookbooks/\*\*'].eachdo |path|

  cookbookFile.basename(path), path: path

**end**

The Apache2 cookbook installs Apache as a Linux service, as Docker by default does not run services we need to disable the creation of the service and create a recipe to start Apache.

Fortunately, the default recipe will only create the service if the only\_if condition defined in the service resource block succeeds. The only\_if condition runs the httpd binary with the -t switch which performs a syntax check on the Apache configuration,where the binary name is defined in the node attribute node[‘apache’][‘binary’].

We can use this to our advantage by creating an attribute in our cookbook to override the default value set in the Apache2 cookbook, and set the value to ‘httpd -‘.  This will cause the condition to fail, hence preventing Apache from being installed as a service.

Create an attribute file named default using the ‘chef generate attribute’ command:

$ cd ~/chef-repo/c2b2\_website

$ chef generate attribute cookbooks/c2b2\_website default

add the following content to the file

default['apache']['binary'] = '/usr/sbin/httpd - '

Create a recipe named startapache using the ‘chef generate recipe’ command:

$ cd ~/chef-repo/c2b2\_website

$ chef generate recipe cookbooks/c2b2\_website startapache

Add the following content to start Apache:

**execute** 'start\_apache' **do**

  command'httpd -k start'

**user** 'root'

**group** 'root'

**action** :run

**end**

Create the recipe ***installapache***:

$ cd ~/chef-repo/c2b2\_website

$ chef generate recipe cookbooks/c2b2\_website installapache

add the following content to include the default recipe from the Apache2cookbook - which installs a basic Apache configuration and the recipe from this cookbook to start Apache:

**include\_recipe** 'apache2::default'

include\_recipe 'c2b2\_website::startapache'

**Configure Test Kitchen (Docker)**

Now let’s configure Test Kitchen to use provision a Docker image based on Centos 7, update the file .kitchen.yml file with the contents below (see [github](https://github.com/spheromak/kitchen-docker/blob/master/README.md:" \t "_blank) for more info on this)

---

driver:

  name: docker

  binary:docker

  use\_sudo:false

provisioner:

  name: chef\_solo

 environments\_path: environments

 coobooks\_path:

    -cookbooks

  ohai:

   disabled\_plugins: ["passwd"]

platforms:

  - name:centos-7

   driver\_config:

     privileged: true

      memory:1512m

      volume:

        -/sys/fs/cgroup:/sys/fs/cgroup:ro

     provision\_command:

        - echo"root:password" | chpasswd

        - sed-i 's/Defaults   requiretty/#Defaults   requiretty/g' /etc/sudoers

suites:

  - name:**default**

    run\_list:

      -recipe[c2b2\_website::installapache]

**Test Kitchen**

Test Kitchen can be run in different modes of operation:

|  |  |
| --- | --- |
| **Command** | **Description** |
| kitchen create | Creates one or more instances configured in the .kitchen.yml file |
| kitchen converge | Converging the vm(s) with the configured Chef policy (cookbooks, roles, environment and data bags) |
| kitchen destroy | Destroys the instance and deletes all information for the instance |
| kitchen list | List on or more instances and their state (Created, converged, Verified) |
| kitchen login |  |
| kitchen verify | Run the test suite(s) configured in the.kitchen.yml file on one or more instances |
| kitchen test | Test (destroys, creates, converges, verifies and destroys) one or more instances |

If we now run ‘kitchen converge’, a Centos7 Docker instance is created and Chef Solo used to converge the instance, running the c2b2\_website cookbook, installing and starting Apache.   
  
We can check that Apache is installed and running by logging to the Docker instance and executing ps -ef | grep httpd to check that the process is running:

$ kitchen login

$$$$$$ Running legacy login for 'Docker' Driver

[kitchen@6c610c7ab9ce ~]$ ps -ef | grep httpd

root      512     1  0 17:32 ?        00:00:00 httpd -k **start**

apache    513   512  0 17:32 ?        00:00:00 httpd -k **start**

apache    514   512  0 17:32 ?        00:00:00 httpd -k **start**

apache    515   512  0 17:32 ?        00:00:00 httpd -k **start**

apache    516   512  0 17:32 ?        00:00:00 httpd -k **start**

apache    517   512  0 17:32 ?        00:00:00 httpd -k **start**

apache    518   512  0 17:32 ?        00:00:00 httpd -k **start**

apache    519   512  0 17:32 ?        00:00:00 httpd -k **start**

apache    520   512  0 17:32 ?        00:00:00 httpd -k **start**

apache    521   512  0 17:32 ?        00:00:00 httpd -k **start**

apache     522  512  0 17:32 ?        00:00:00 httpd -k **start**

apache    523   512  0 17:32 ?        00:00:00 httpd -k **start**

apache    524   512  0 17:32 ?        00:00:00 httpd -k **start**

There you have it, a way to develop and provision Chef cookbooks to Docker instance running in this case Centos 7.  In the next blog I’ll take this a step further and add integration tests to Test Kitchen to validate the Apache installation.