**Set up Test Kitchen for windows in Microsoft Azure**

Step1: Install the **kitchen-azurerm** driver on your workstation.

**chef gem install kitchen-azurerm**

When we generate a cookbook Chef creates a file named .kitchen.yml in the root directory of our cookbook. .kitchen.yml defines what's needed to run Test Kitchen, including which virtualization provider to use, how to run Chef, and what platforms to run our code on.

Step2: cd to your cookbook directory and change .kitchen.yml accordingly.

---

driver:

name: azurerm

driver\_config:

subscription\_id: '4801fa9d-our subscription\_id-b265ff49ce21'

location: 'West Europe'

machine\_size: 'Standard\_D1'

provisioner:

name: chef\_zero

platforms:

- name: windows2012-r2

driver\_config:

image\_urn: MicrosoftWindowsServer:WindowsServer:2012-R2-Datacenter:latest

transport:

name: winrm

username: username

suites:

- name: default

run\_list:

- recipe[ourcookbook::default]

attributes:

* **driver** specifies the software that manages the machine. We're using the Azure Test Kitchen driver.
* **driver\_config** specifies additional configuration options for the Azure Test Kitchen driver, including our subscription ID, the region where we'll run our test instances, and the VM size.
* **provisioner** specifies how to run Chef. We use chef\_zero because it enables us to mimic a Chef server environment on our local machine. This allows us to work with node attributes and other Chef server features.
* **transport** specifies how to execute commands remotely on the test instance. WinRM is the default transport on Windows; SSH is the default on all other operating systems.
* **verifier** specifies which application to use when running automated tests.
* **platforms** specifies the target operating systems. We're targeting Windows Server 2012 R2. (We can manage more than one instance at a time to enable us to test our cookbooks on multiple platforms.)
* **suites** specifies what we want to apply to the virtual environment. We can have more than one suite. We define just one, named default. This is where we provide the run-list, which defines which recipes to run and in the order to run them. Our run-list contains one recipe.

Step3: Create the Test Kitchen instance. We'll provision a virtual machine to serve as your test environment. This is the kitchen create step in our workflow



**Kitchen create:-** Create the instance.

* Run the following to check the instance created.

kitchen list.

**Kitchen converge:-** To apply the cookbook to the Windows Server virtual machine.

* Test Kitchen runs chef-client on the instance. When the chef-client run completes successfully, Test Kitchen exits with exit code 0. Run the following to check the exit code.

echo $?

* If we receive a result other than 0, fix the errors that were reported. Then run kitchen converge to apply the changes and again check the exit code.

**[Kitchen login](https://docs.chef.io/ctl_kitchen.html" \l "kitchen-login):-** It is the most common way to access your test instance. This command creates a WinRM connection into your instance and enables you to explore your test instance and verify its configuration.

**Verify:-** To verify instances. We typically write automated tests that verify whether your instance is configured as we expect. Having automated tests enables us to quickly verify that our configuration works as we add features to your cookbook

**Kitchen destroy:-** We're all done with our virtual machine, so now run the kitchen destroy command to delete it.