Chef and DSC: Module 7, Lesson 5   
Using Chef and DSC to set up an IIS Server

## Overview

In this lab you will use utilize DSC to install and configure the DSC module xWebAdministration on Windows Server 2012 using Chef’s dsc\_resource. You will also see the benefit of dsc\_script and its guard attribute.

## Objectives

In this hands-on lab you will learn how to:

* Launch a virtual Windows Server 2012 R2 machine
* Install the Chef Development Kit the VM
* Install the needed DSC modules
* Create a cookbook for configuring IIS
* Write a recipe to configure these components utilizing dsc\_resource and powershell\_script
* Configure a basic Default.htm to serve on port 80
* Verify that your new webpage is being hosted and can be resolved

## Prerequisites

The following are required to complete this hands-on lab:

* A Microsoft Azure subscription
* A web browser
* An RDP client to connect to the Windows VM, such as Remote Desktop Connection from a Windows machine or Microsoft Remote Desktop for a Mac.
* Windows Management Framework 5.0 or above installed on the VM

**Note:** The Azure portal is continually improved and changed. The steps in this exercise reflect the user interface of the Microsoft Azure portal at the time of writing, but may not match the latest design of portal.

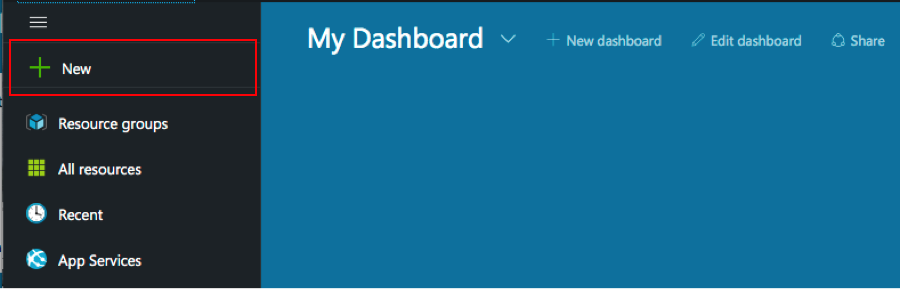
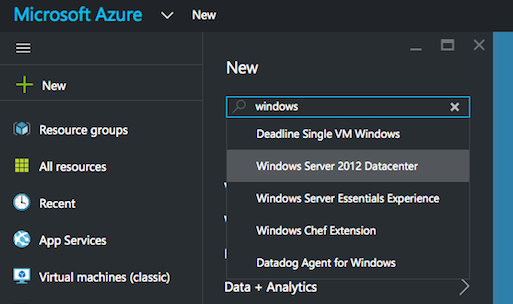
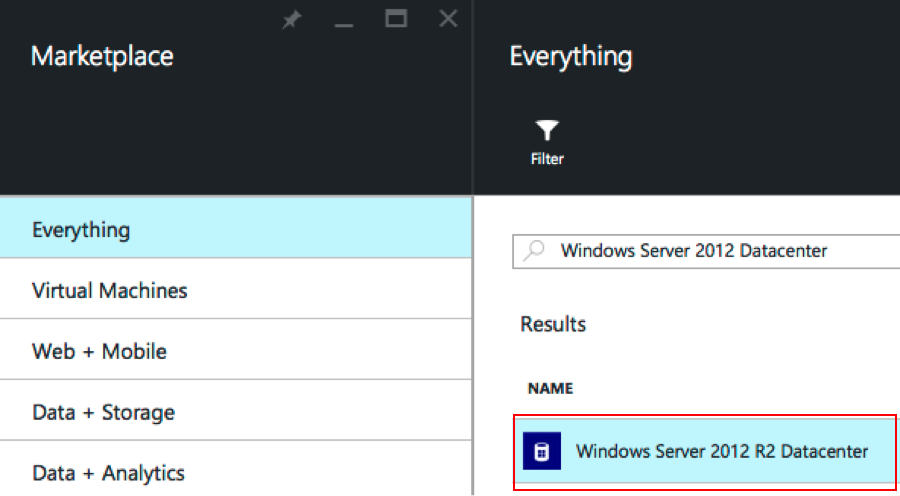
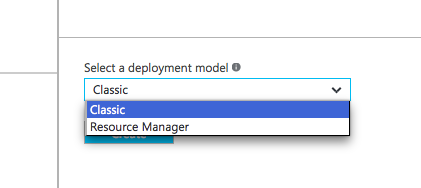
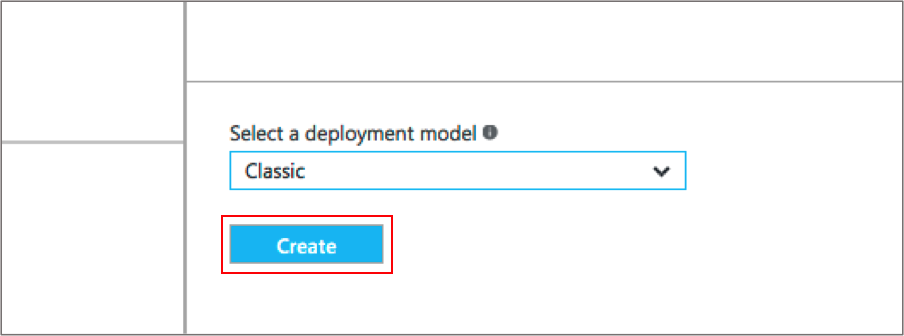
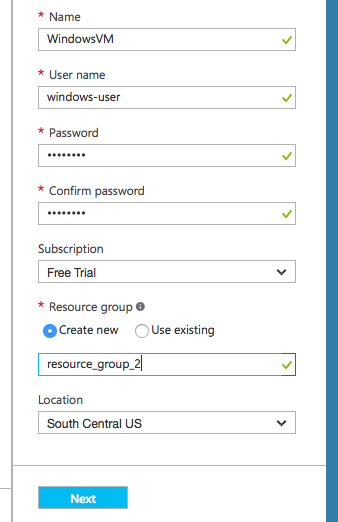
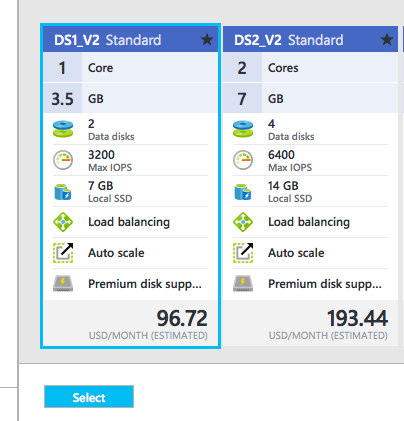
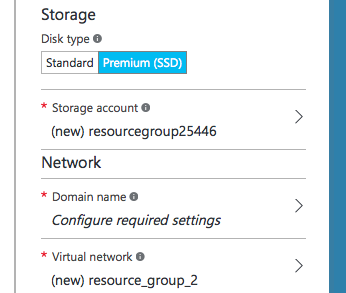
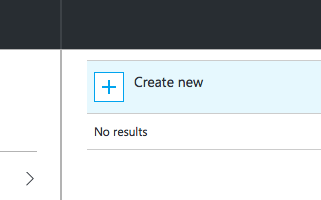
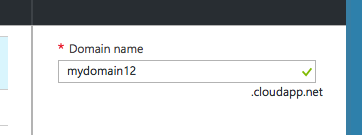
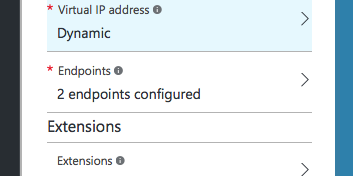
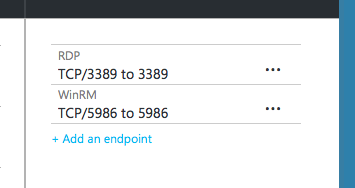
## Exercises

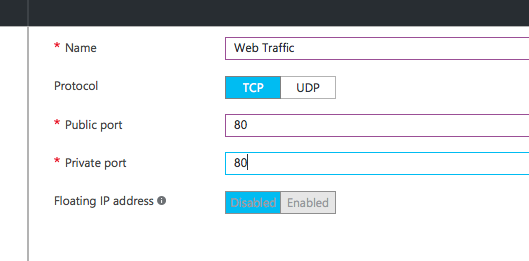
This hands-on lab includes the following exercises:

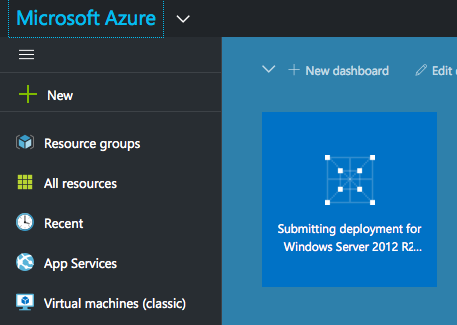
* Exercise 1: Launch a Windows Server 2012 R2 VM
* Exercise 2: Ensure that WMF 5.0 or above is installed
* Exercise 3: Install the Chef Development Kit (ChefDK) on each VM
* Exercise 4: Create a cookbook for setting up System Center and Operations Manager
* Exercise 5: Write a recipe utilizing dsc\_resource and dsc\_script
* Exercise 6: Converge the node and verify that these components are installed

## Exercise 1: Launch a Windows Server 2012 R2 VM

Launch a Windows 2012 R2 Datacenter VM. Ensure that Powershell 5.0 or above is installed.

1. From the dashboard, click ‘New’ to start.  
   
2. Search for windows to find a Windows image  
   
3. Choose the Windows Server 2012 R2 Datacenter image  
   
4. Choose the Classic deployment model  
   
5. Click Create  
   
6. Complete the form  
   
7. Choose the Machine Size (use the least expensive machine size)  
   
8. Configure the Domain Name  
   
9. Create a new Domain Name  
   
10. Create a new Domain Name. This can be arbitrary so if the domain name you have chosen is not available, create a different domain name of your choice. Click ‘OK’  
    
11. Create a new endpoint  
    
12. Click ‘Add an endpoint’ to configure the endpoints  
    
13. Configure the Endpoint
    1. Complete the form
    2. Use TCP as the Protocol
    3. Give the Endpoint a name
    4. Use port 80 as the Public and Private port



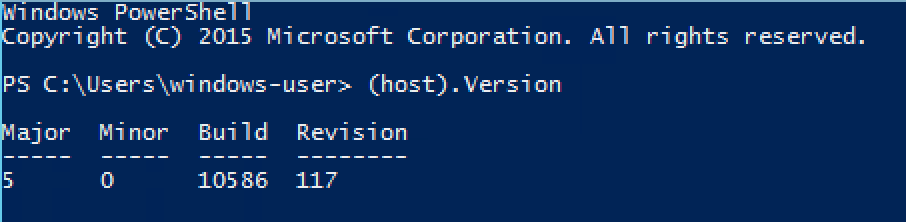
1. Click ‘OK’ several times to clear the various screens and launch your virtual machine.
2. Your instance will launch  
   
3. Find the new VM
   1. From the Dashboard click ‘All Resources’
   2. Use the search function
   3. Find the new VM
   4. Click on the VM to find its IP Address
4. Connect to the new Windows VM using Remote Desktop Connection (from a Windows workstation) or install Microsoft Remote Desktop (if you are using a Mac).
5. After clicking on the new VM from the ‘All Resources’ blade, click ‘Connect’ to download the RDP connection information (and open this connection in your RDP client), or use the IP Address and the username and password that you created to connect manually from the RDP client.

## Exercise 2: Ensure that Windows Management Framework (WMF) 5.0 is installed

WMF 5.0 or above is required in order to utilize Chef’s dsc\_resource. Ensure this is installed, and if not grab the appropriate installer.

Check if WMF 5 is installed

1. Note: this operation is done from the new Windows VM that you just launched
2. Open a Powershell window, executing the following command:
   1. PS> (host).Version



If version 5 or above is not installed:

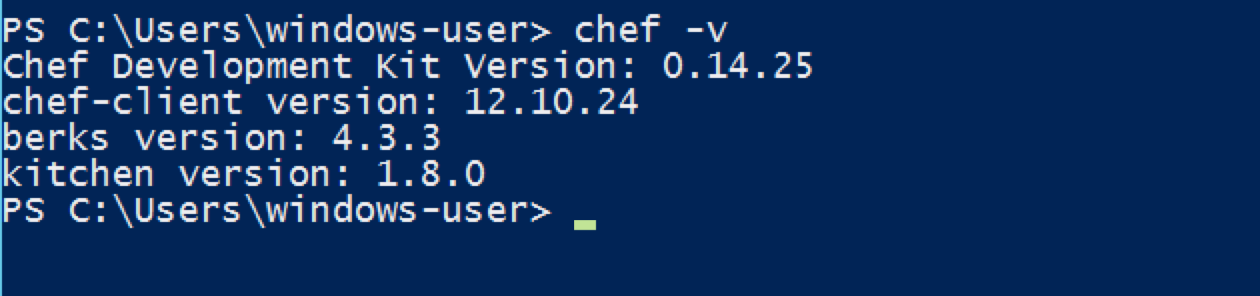
1. Open an internet browser and navigate to <https://msdn.microsoft.com/en-us/powershell/wmf/requirements?f=255&MSPPError=-2147217396>
2. Select the appropriate installer (in this Exercise, Windows 2012 R2) and download the MSI
   1. Note: You may need to adjust your security preferences in order to enable downloads on Server 2012. If using Internet Explorer, open Internet Options, select the “Security” tab, and then click on the “Internet” zone. From there, select a “Custom” security level, and under “Downloads” enable “File Downloads”.
3. Install the MSI, reboot the system.

## Exercise 3: Install the Chef Development Kit (ChefDK) on VM

Now that you’ve created a Windows 2012 R2 VM, let’s install the ChefDK

Installing the ChefDK

1. Note: this operation is done from the new Windows VM that you just launched
2. Find the ChefDK installation Windows msi
   1. Using a web browser from the VM, navigate to <http://downloads.chef.io/chef-dk>
3. Choose ‘Windows’
4. Find the Download button for ‘Windows’
   1. Note: if you do not see the download button, install another browser, such as Firefox
5. Download and install the Windows msi
6. Verify the ChefDK installation
   1. From a PowerShell window, execute the ‘chef -v’ command   
      PS> **chef -v**
   2. Reporting of the version numbers indicates a successful installation



## Exercise 4: Create a cookbook called “iis-dsc” and recipe for setting up IIS and a configuring a basic webserver

Now that you’ve got a functioning 2012 R2 VM with Powershell 5.0, WMF 5.0 and the ChefDK installed, let’s start writing some Chef code!

1. Create a cookbooks directory
   1. Log into the Windows instance using RDP
   2. From the home directory, create a cookbooks directory

PS> mkdir cookbooks

* 1. This is where cookbooks will be stored

1. Ensure that the DSC Module xWebAdministration is installed. This will allow us to configure IIS using DSC. Run the command:

PS> Find-Module xWebAdministration | Install-Module

* 1. This will likely prompt you to install NuGet, which allows us to grab modules from sources like powershellgallery.com. Answer yes to all prompts.
  2. A second option is to run Install-PackageProvider -Name NuGet -MinimumVersion 2.8.5.201 -Force

1. Create a new cookbook
   1. From the home directory, use the chef generate command to create the iis-dsc cookbook. This will create the specific cookbook directory its needed subdirectories and files:

PS> chef generate cookbook cookbooks/iis-dsc

Note: this assumes you are running the command from the home directory, hence the path of cookbooks/iis-dsc

## Exercise 5: Write two recipes, one for setting the proper RefreshMode for the LCM, the default.rb for setting up IIS.

1. Create a new recipe, to be stored in the new cookbook, that contains the powershell\_script Chef resource that sets the LCM RefreshMode to ‘Disabled’. Remember that this is required to use dsc\_resource. If you chose to use dsc\_script to write this configuration, the RefreshMode would need to be set to ‘PUSH’.
   1. From the Windows RDP Session, create a new file named   
      ‘configure\_lcm.rb’ in the ~/cookbooks/iis-dsc/recipes directory
   2. Use any editor. You can install an editor such as Notepad++, Sublime Text or Atom, or for just this exercise, you can use Notepad
   3. This is the recipe for setting the LCM’s RefreshMode to Disabled:
   4. Understanding Chef’s powershell\_script resource.
      1. The package name is ‘Install IIS’ but with PowerShell Script this doesn’t impact the installation at all. It is the ‘code’ attribute that does the actual install.
      2. The ‘guard\_interpreter ‘ determines which interpreter will implement the idempotence (in case the web server is already installed)
      3. The ‘not\_if’ line is the logic for the idempotence, stating that if the LCM is NOT set to Disabled already, then execute this script, otherwise if the LCM IS in the desired state, do not run the script.
2. Create the code for setting up IIS using DSC. Here we create this functionality inside of the default.rb recipe. This recipe is created with the ‘chef generate cookbook’ command.
   1. First, when setting up IIS using DSC resources, we want to run our configure\_lcm recipe to ensure that the RefreshMode is properly set.

powershell\_script 'Configure LCM for dsc\_resource' do

code <<-EOH

[DscLocalConfigurationManager()]

Configuration ConfigLCM

{

Node "localhost"

{

Settings

{

ConfigurationMode = "ApplyOnly"

RebootNodeIfNeeded = $false

RefreshMode = 'Disabled'

}

}

}

ConfigLCM -OutputPath "#{Chef::Config[:file\_cache\_path]}\\DSC\_LCM"

Set-DscLocalConfigurationManager –Path "#{Chef::Config[:file\_cache\_path]}\\DSC\_LCM"

EOH

only\_if '(Get-DscLocalConfigurationManager).RefreshMode -notlike "Disabled"'

end

We do this with the include\_recipe method, which calls the selected recipe in order at wherever the method is called. Remember that we made sure this powershell\_script was idempotent by utilizing a guard. The first line of your recipe should be:

* + 1. Include\_recipe “iis-dsc::configure\_lcm”
  1. We can now begin scaffolding out the logic that serves a basic Default.htm file. First, we install IIS.
     1. Calling dsc\_resource, we set the resource attribute to :windowsfeature, meaning that we are adding a feature. This resource’s :name property is ‘web-server’, which is bundled from the xWebAdministration module installed prior.
  2. The ‘not\_if’ line is the logic for the idempotence, stating that if the web server is NOT installed, then do this installation, otherwise if the web server IS installed, skip the installation.

dsc\_resource 'Install IIS' do

resource :windowsfeature

property :name, 'web-server'

end

* + 1. Congrats, IIS is set up! It will now serve the default configuration to the web, which we would like to modify.

1. Instruct Chef to write a web page for IIS to serve up
   1. First, let’s disable the default page being served to the web. This site is installed by default in the C:\inetpub\wwwroot\ directory. We can disable it using our newly configured xwebsite dsc\_resource that was installed from the xWebAdministration Module.

dsc\_resource 'Shutdown Default Website' do

resource :xwebsite

property :name, 'Default Web Site'

property :State, 'Stopped'

property :PhysicalPath, 'C:\inetpub\wwwroot'

end

Once the :xwebsite resource has been called, we can make a call to the :State property and assign it to ‘Stopped’. We describe the location of the site with the :PhysicalPath property.

1. Now that the default website has been shut down, we can prepare to make our own at the same path. This requires us to set up some working directories, and we’ll manage those resources using Chef’s dsc\_resource. We’ll set up this working directory first.
   1. First, let’s assign a working directory to the variable site\_dir by giving our site a name:
      1. site\_name = “My Awesome Website”
      2. site\_dir = "#{ENV['SYSTEMDRIVE']}\\inetpub\\wwwroot\\#{site\_name}"
      3. This ensures that the working directory for our website is located on the main hard drive, and it will reference the site\_name variable when you’d like to change it. Let’s now configure the path to this file using DSC’s file resource:

dsc\_resource "#{site\_name} Directory' do

resource :file

module\_name 'PSDesiredStateConfiguration'

property :DestinationPath, site\_dir

property :Type, 'Directory'

end

* 1. Great! Now we can reference this as the location of our Application Pool. This is very easy to configure with the :xWebAppPool resource.

dsc\_resource "#{site\_name} App Pool' do

resource :xWebAppPool

property :Name, site\_name

end

* 1. Now, associate this new Application Pool with our new site’s path:

dsc\_resource “#{site\_name} Web Site” dp

resource :xWebSite

property :Name, site\_name

property :ApplicationPool, site\_name

property :PhysicalPath, site\_dir

end

* 1. Finally, our app is ready to serve a new site!

1. After shutting down the default site, we can set up our own at the same path. There are a couple of ways to do this, using either Chef’s file resource or a template resource. The template resource is more robust and real-world applicable, but for this exercise we can create a basic site using the file resource and some environmental variables.

The file name is ‘c:\inetpub\wwwroot\Default.htm’. This is the path on the VM where we want to write the HTML file. This location was chosen because it is the default path and filename where IIS will look for the HTML file to serve when a request comes in on port 80.

file "#{site\_dir}\\Default.htm" do

content "<html><body><h1>Hello World!</h1><h2>Welcome to#{site\_name}</h2></body></html>"

rights :read, 'Everyone'

action :create

notifies :restart, 'service[w3svc]'

end

We make a call to Chef’s standard file resource, placing a file at the c:\inetpub\wwwroot\ directory called Default.htm. Note here the reference to the site name with Ruby string interpolation, as in “#{site\_name}” and “#{site\_dir}”. To reference variables inside of a string using Ruby we must wrap the string in double quotes as well. Also, backspaces are designated as special characters inside of Ruby stings, meaning that to reference a path when using double quotes we must specify that path with 2 backslashes. Hence the path of “#{site\_dir}\\Default.htm”.

The content attribute here will fill the content of the file with whatever we specify inside of a string argument, which is the basic html for our website. For permissions on this file, we give “Everyone” read access. Finally, if Chef does write/modify this file, we make use of notifications with the “notifies :restart, ‘service[w3svc]’”. This means we will only restart the w3svc service if this file is modified, saving us from restarting the service on every chef-client run.

1. Start the IIS web service
   1. Use the ‘service’ resource to start IIS
   2. ‘w3svc’ is the service name for IIS
   3. Brackets denote that two separate actions are being implemented
   4. The two actions ‘enable’ the service to start upon reboot as well ‘start’ the service now

service 'w3svc' do

action [:enable, :start]

end

1. Completed default.rb file:

include\_recipe "iis-dsc::configure\_lcm"

site\_name = "MyWebSite"

dsc\_resource 'Install IIS' do

resource :windowsfeature

property :name, 'web-server'

end

dsc\_resource 'Shutdown Default Website' do

resource :xwebsite

property :name, 'Default Web Site'

property :State, 'Stopped'

property :PhysicalPath, 'C:\inetpub\wwwroot'

end

site\_dir = "#{ENV['SYSTEMDRIVE']}\\inetpub\\wwwroot\\#{site\_name}"

dsc\_resource "#{site\_name} Directory" do

resource :file

module\_name 'PSDesiredStateConfiguration'

property :DestinationPath, site\_dir

property :Type, 'Directory'

end

dsc\_resource "#{site\_name} App Pool" do

resource :xWebAppPool

property :Name, site\_name

end

dsc\_resource "#{site\_name} Web Site" do

resource :xWebSite

property :Name, site\_name

property :ApplicationPool, site\_name

property :PhysicalPath, site\_dir

end

file "#{site\_dir}\\Default.htm" do

content "<html><body><h1>Hello World!</h1><h2>Welcome to #{site\_name}</h2></body></html>"

rights :read, 'Everyone'

action :create

notifies :restart, 'service[w3svc]'

end

service 'w3svc' do

action [:enable, :start]

end

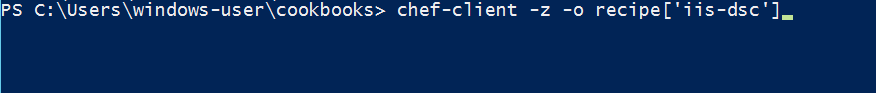
## Exercise 6: Converge the node instance verify that the basic webserver is functional.

1. Converge the Node
   1. Instruct the node to execute the Chef recipe
   2. From the Powershell command line, in the home directory, invoke chef-client
   3. First, move to the home directory

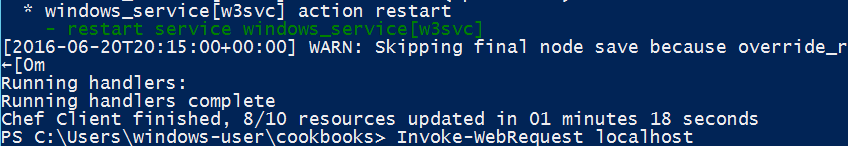
PS> cd ~

* 1. Next, execute the command as follows. Note the –z option instructs chef-client to run in local mode and not use a Chef Server, and the –o option points to the recipe we want to run and the cookbook in which that recipe exists. Without specifying the recipe to run, the default.rb recipe is selected.

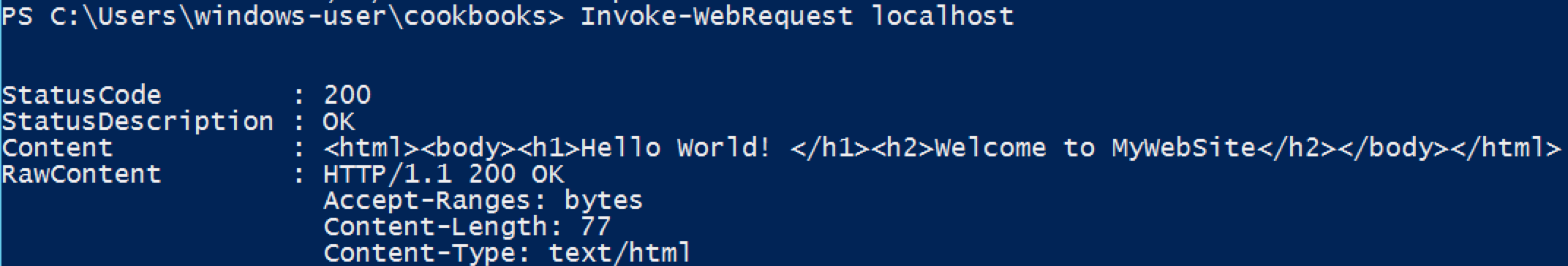
PS> chef-client -z -o recipe[ iis-dsc ]



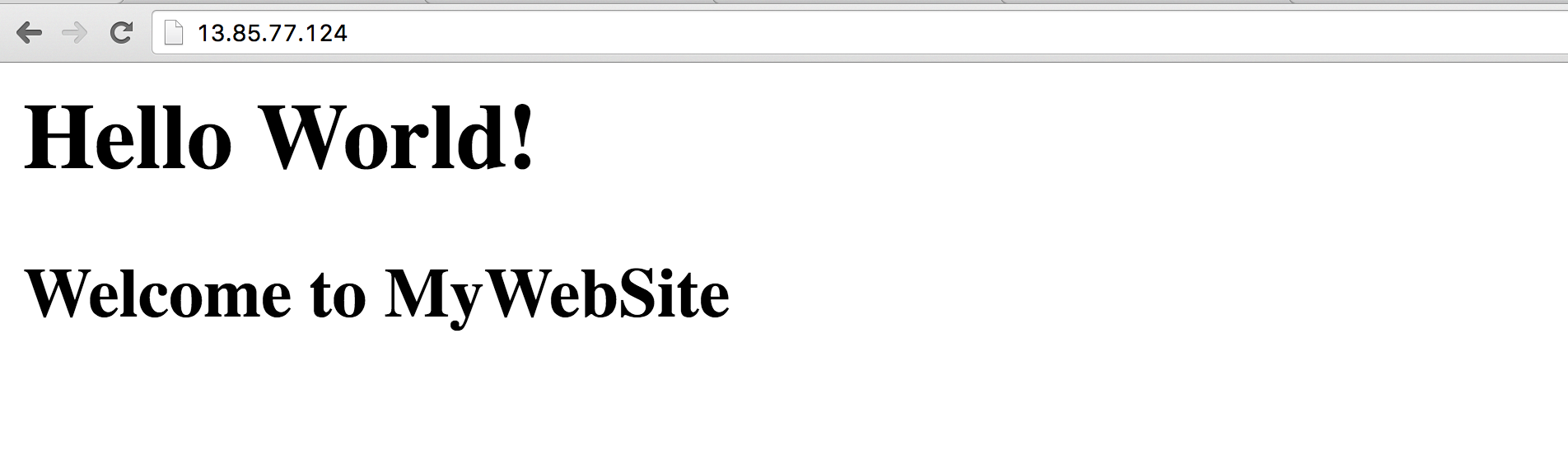
* 1. Watch the node install, configure and start the IIS service.
  2. Note the following:
     1. ‘Chef Client finished’ message
     2. Total number of resources updated
     3. Time taken by the run



1. Verify Web Server Functionality via the command line to eliminate potential VM networking issues
   1. **PS> Invoke-WebRequest localhost** can verify (without being limited by network issues):
      1. the web server has been installed
      2. the w3svc (IIS) service is running
      3. the web page is being served correctly



1. Verify Web Server Functionality visually
   1. Find the IP Address of the virtual machine
   2. Enter the IP Address into a web browser and see the ‘Hello World’ page displayed



**\*\*\*Don’t forget to delete your VM’s in order to avoid wasting your Azure credit\*\*\***

## Summary

In this hands-on lab, you learned how to:

* Launch a virtual Windows Server 2012 R2 Datacenter machine
* Install the Chef Development Kit and WMF 5 on the VM
* Create a web server cookbook that utilizes chef’s dsc\_resource and powershell\_script
* Write a recipe to install, configure and start a web server using DSC
* Verify that the web server is running from the command line and from a web browser