Contents

[Installation and setup 3](#_Toc522437003)

[Chef development kit on the workstation. 3](#_Toc522437004)

[Install chef on centos7 (AWS) 3](#_Toc522437005)

[Sample recipe 3](#_Toc522437006)

[Resources 4](#_Toc522437007)

[About recipes: 4](#_Toc522437008)

[cookbooks: 4](#_Toc522437009)

[Version control for cookbooks 5](#_Toc522437010)

[Setting up a web server: 5](#_Toc522437011)

[Chef client run 6](#_Toc522437012)

[Include recipe 6](#_Toc522437013)

[Ohai/Node object 6](#_Toc522437014)

[Templates 7](#_Toc522437015)

[Other resources 8](#_Toc522437016)

[cookbook\_file 8](#_Toc522437017)

[remote\_file 8](#_Toc522437018)

[bash 9](#_Toc522437019)

[execute, directory 9](#_Toc522437020)

[user,group 9](#_Toc522437021)

[notifies, subscribes 9](#_Toc522437022)

[Troubleshooting and debugging 10](#_Toc522437023)

[Test kitchen 11](#_Toc522437024)

[ChefSpec 12](#_Toc522437025)

[Chef server 12](#_Toc522437026)

[Hosted chef 13](#_Toc522437027)

[Knife 13](#_Toc522437028)

[Bootstrap 13](#_Toc522437029)

[Supermarket and cookbook wrapping 14](#_Toc522437030)

[Berkshelf 15](#_Toc522437031)

[Knife node command 15](#_Toc522437032)

[Roles 16](#_Toc522437033)

[Knife search 17](#_Toc522437034)

# Installation and setup

## Chef development kit on the workstation.



## Install chef on centos7 (AWS)

# Run these commands on the cloud instance after connecting with ssh

centos@ip$ curl https://omnitruck.chef.io/install.sh | sudo bash -s -- -P chefdk -c stable -v 0.18.30

centos@ip$ chef --version

centos@ip$ sudo yum install nano -y

centos@ip$ sudo yum install vim -y

centos@ip$ exit

## Sample recipe

# This is an example of the file you should have written on your vagrant or cloud instance

# The title of the recipe is 'hello.rb'

# This ruby file will be run with the command 'sudo chef-client --local-mode hello.rb'

# The result is that a file, hello.txt, will be created in the root directory

# with the content "Hello, world!"

#

# The file, named '/hello.txt' is created

# with the content 'Hello, world!'

#

# @see https://docs.chef.io/resource\_file.html

#

file 'hello.txt' do

content 'Hello, world!'

owner ‘root’

group ‘root’

mode ‘0644’

end

## Resources



**Package resource:**

The package named 'httpd' is installed.

package 'httpd' do

action :install

end

**Service resource:**

service 'ntp' do

action [ :enable, :start ]

end

**File resource**

file '/etc/php.ini.default' do

action :delete

end

Resources reference: <https://docs.chef.io/resource_reference.html>

## About recipes:

<https://docs.chef.io/recipes.html>

# cookbooks:

<https://docs.chef.io/cookbooks.html>



# 4-3 Cookbook Components

# commands that involve editing files assume nano as the text editor

# this command list is provided to accompany the demos in the Chef Fundamentals Udemy course

# run these commands from your vagrant or cloud CentOS 7 instance

cd ~

chef --help

chef generate --help

chef generate cookbook --help

mkdir cookbooks

chef generate cookbook coobooks/workstation

tree

cat cookbooks/workstation/metadata.rb

cat cookbooks/workstation/README.md

cat cookbooks/workstation/recipes/default.rb

mv setup.rb cookbooks/workstation/recipes/

## Version control for cookbooks

# 4-4 Revision with Git

# this command list is provided to accompany the demos in the Chef Fundamentals Udemy course

# commands that involve editing files assume nano as the text editor

# run these commands from your vagrant or cloud CentOS 7 instance

cd ~

tree

which git

nano cookbooks/workstation/recipes/setup.rb

sudo chef-client -z cookbooks/workstation/recipes/setup.rb

cd cookbooks/workstation/

pwd

tree

git init

ls -a

git status

git add .

git status

git commit -m "initial workstation cookbook commit"

git status

nano recipes/setup.rb

git status

git add recipes/setup.rb

git commit -m "added ntp to setup.rb"

git status

## Setting up a web server:



# 4-5 Lab: Deploy Apache Webserver

# this command list is provided to accompany the demos in the Chef Fundamentals Udemy course

# commands that involve editing files assume nano as the text editor

# run these commands from your vagrant or cloud CentOS 7 instance

cd ~

chef generate cookbook cookbooks/apache

tree

chef generate --help

chef generate recipe cookbooks/apache/server

tree

nano cookbooks/apache/recipes/server.rb

chef exec ruby -c cookbooks/apache/recipes/server.rb

sudo chef-client -z cookbooks/apache/recipes/server.rb

curl localhost

Check syntax of the ruby file:

>chef exec ruby –c server.rb

## Chef client run



Chef-client overview: <https://docs.chef.io/chef_client_overview.html>

# 5-2 Applying Recipes and Cookbooks

# this command list is provided to accompany the demos in the Chef Fundamentals Udemy course

# commands that involve editing files assume nano as the text editor

# run these commands from your vagrant or cloud CentOS 7 instance

cd ~

pwd

ls cookbooks

tree cookbooks/apache

sudo chef-client -z --runlist "apache::server"

sudo chef-client -z "workstation::setup"

sudo chef-client -z --runlist "workstation::setup"

sudo chef-client -z -r "recipe[apache::server]"

sudo chef-client -z -r "recipe[apache::server],recipe[workstation::setup]"

## Include recipe

Info: <https://docs.chef.io/recipes.html#include-recipes>

# 5-3 The include\_recipe Method

# this command list is provided to accompany the demos in the Chef Fundamentals Udemy course

# commands that involve editing files assume nano as the text editor

# run these commands from your vagrant or cloud CentOS 7 instance

cd ~

tree cookbooks/workstation

nano cookbooks/workstation/recipes/default.rb

cat cookbooks/workstation/recipes/default.rb

# ~/cookbooks/workstation/recipes/default.rb

include\_recipe 'workstation::setup'

sudo chef-client -z -r "recipe[workstation::default]"

sudo chef-client -z -r "recipe[workstation]"

nano cookbooks/apache/recipes/default.rb

sudo chef-client -z -r "recipe[apache]"

sudo chef-client -zr "recipe[apache],recipe[workstation]"

cd cookbooks/apache

git status

git add .

git commit -m "default.rb includes server.rb"

git status

# Ohai/Node object

Info: <https://docs.chef.io/ohai.html>



The node object: <https://docs.chef.io/nodes.html#node-objects>

# 6-3 Node Attributes

# this command list is provided to accompany the demos in the Chef Fundamentals Udemy course

# commands that involve editing files assume nano as the text editor

# run these commands from your vagrant or cloud CentOS 7 instance

cd ~

cat /etc/motd

ohai

ohai ipaddress

ohai hostname

ohai memory

ohai memory/total

ohai cpu/0/mhz

nano cookbooks/workstation/recipes/setup.rb

sudo chef-client -zr "recipe[workstation]"

cat /etc/motd

# ~/cookbooks/workstation/recipes/setup.rb

package 'nano'

# or package 'vim-enhanced'

package 'ntp'

package 'git' do

action :install

end

file '/etc/motd' do

content "This server is the property of Chef

HOSTNAME: #{node['hostname']}

IPADDRESS: #{node['ipaddress']}

CPU: #{node['cpu']['0']['mhz']}

MEMORY: #{node['memory']['total']}

"

action :create

end

## Templates

Info: <https://docs.chef.io/templates.html>



Create template motd:

chef generate template --help cookbooks/workstation/ motd

In the recipe setup.rb we will replace file resource by template:

Code of the template file

centos@centos01:~>cat cookbooks/workstation/templates/motd.erb

This server is the property of <%= @name %>

HOST: <%= node['hostname'] %>

IP: <%= node['ipaddress'] %>

CPU: <%= node['cpu']['0']['mhz'] %> Mhz

MEMORY: <%= node['memory']['total'] %>

template '/etc/motd' do

source 'motd.erb'

variables(

:name => 'Dimitry Karalov'

)

action :create

end

## Other resources

### cookbook\_file

Copy static file from files storage

<https://docs.chef.io/resource_cookbook_file.html>

>chef generate file cookbooks/apache error.html

>tree

>vi cookbooks/apache/files/default/error.html

>vi cookbooks/apache/recipes/server.rb

>cat cookbooks/apache/recipes/server.rb

package 'httpd' do

action :install

end

template '/var/www/html/index.html' do

source 'index.html.erb'

variables(

:name => 'Dima'

)

action :create

end

**cookbook\_file '/var/www/html/error.html' do**

**source 'error.html'**

**end**

service 'httpd' do

action [:enable, :start]

end

### remote\_file

Copy file from remote source (URL)

<https://docs.chef.io/resource_remote_file.html>

remote\_file '/var/www/html/cat.jpg' do

source 'https://www.petmd.com/sites/default/files/what-does-it-mean-when-cat-wags-tail.jpg'

end

### bash

with bash code condition

bash “inline script” do

user “root”

code “mkdir –p /var/www/mysites && chown –R apache /var/www/mysites”

not\_if ‘[ -d /var/www/mysites]’

end

with ruby code condition

not\_if do

File.directory?(‘/var/www/mysites’)

end

### execute, directory

execute shell scripts/commands using default system shell

<https://docs.chef.io/resource_execute.html>

# execute 'run a script' do

# command <<-EOH

# mkdir -p /var/www/mysites/ /

# chown -R apache /var/www/mysites/

# EOH

# not\_if do

# File.directory?('/var/www/mysites/')

# end

# end

# the directory resource that should be used

# when handling directory creation

directory '/var/www/mysites' do

owner 'apache'

recursive true

end

### user,group

create user, create group

<https://docs.chef.io/resource_user.html>

<https://docs.chef.io/resource_group.html>

user 'user1' do

comment 'user1'

uid '123'

home '/home/user1'

shell '/bin/bash'

end

group 'admins' do

members 'user1'

end

### notifies, subscribes

will run action on resource in case of other resource changed

<https://docs.chef.io/resource_common.html#notifications>

template '/var/www/html/index.html' do

source 'index.html.erb'

**notifies :restart, 'service[httpd]', :immediately**

end

service 'httpd' do

action [:enable,:start]

**subscribes :restart, 'template[/var/www/html/index.html', :immediately**

end

# Troubleshooting and debugging

<https://docs.chef.io/about_chefdk.html>

# 9-3 Your Toolkit

# this command list is provided to accompany the demos in the Chef Fundamentals Udemy course

# commands that involve editing files assume nano as the text editor

# run these commands from your vagrant or cloud CentOS 7 instance after logging in with 'vagrant ssh'

chef -v

chef generate --help

chef --help

rspec

tree cookbooks/apache

chef exec rspec --help

kitchen --help

berks --help

knife --help

knife client list

rubocop cookbooks/apache

chef exec rubocop cookbooks/apache

foodcritic --help

chef exec foodcritic --help

| **Tool** | **Description** |
| --- | --- |
| Berkshelf | A dependency manager for cookbooks. |
| chef | A workflow tool for Chef. |
| chef-client | The agent that runs Chef. |
| chef-vault | Use to encrypt data bag items using the public keys of a list of nodes. This allows only those nodes to decrypt the encrypted values. |
| ChefSpec | A unit testing framework that tests resources locally. |
| Cookstyle | A Rubocop-based style-checking tool for writing clean cookbooks. |
| Delivery CLI | A command-line tool for continuous delivery workflow. Is used to setup and execute phase jobs on a Chef Automate server. |
| Fauxhai | A gem for mocking Ohai data in ChefSpec tests. |
| Foodcritic | A lint tool for static analysis of recipe code. |
| Test Kitchen | An integration testing framework tool that tests cookbooks across platforms. |
| kitchen-dokken | A test-kitchen plugin that provides a driver, transport, and provisioner for rapid cookbook testing and container development using Docker and Chef. |
| kitchen-vagrant | A Kitchen driver for Vagrant. |
| knife-spork | A workflow plugin for knife that helps groups of people work together in the same chef-repo and Chef server. |
| Ruby | The reference language for Chef. |

The chef command: <https://docs.chef.io/ctl_chef.html>

Knife plugins: <https://docs.chef.io/plugin_knife.html>

Ohai plugins: <https://docs.chef.io/ohai_custom.html>

Debugging: <https://docs.chef.io/debug.html>

Ruby basics: <https://docs.chef.io/ruby.html>

## Test kitchen

About test kitchen: <https://docs.chef.io/kitchen.html>

Kitchen.ci: <http://kitchen.ci/>

chef generate --help

cd apache/

ls -a

cat .kitchen.yml

nano .kitchen.yml

kitchen list

kitchen create

kitchen list

kitchen login

kitchen converge

kitchen login

kitchen verify

nano test/recipes/default\_test.rb

kitchen verify

kitchen destroy

# ~/cookbooks/apache/test/recipes/default\_test.rb

# # encoding: utf-8

# Inspec test for recipe apache::default

# The Inspec reference, with examples and extensive documentation, can be

# found at https://docs.chef.io/inspec\_reference.html

describe port(80) do

it { should be\_listening }

its('protocols') { should include 'tcp6'}

end

describe command('curl localhost') do

its(:stdout) { should match(/Hello, world!/) }

end

## ChefSpec

<https://docs.chef.io/chefspec.html>

<https://github.com/sethvargo/chefspec>

cd ~/cookbooks/apache

pwd

tree

chef generate recipe test

tree

chef exec rspec

nano spec/unit/recipes/default\_spec.rb

nano spec/spec\_helper.rb

chef exec rspec spec/unit/recipes/default\_spec.rb

# ~/cookbooks/apache/spec/unit/recipes/default\_spec.rb

#

# Cookbook Name:: apache

# Spec:: default

#

# Copyright (c) 2015 The Authors, All Rights Reserved.

require 'spec\_helper'

describe 'apache::default' do

context 'When all attributes are default, on an unspecified platform' do

let(:chef\_run) do

runner = ChefSpec::ServerRunner.new

runner.converge(described\_recipe)

end

it 'installs the correct package' do

expect(chef\_run).to install\_package('apache2')

end

it 'creates an default html file' do

expect(chef\_run).to create\_template('/var/www/html/index.html')

end

it 'starts the service' do

expect(chef\_run).to start\_service('apache2')

end

it 'enables the service' do

expect(chef\_run).to enable\_service('apache2')

end

end

end

# ~/cookbooks/apache/spec/spec\_helper.rb

require 'chefspec'

require 'chefspec/berkshelf'

ChefSpec::Coverage.start!

# Chef server

<https://docs.chef.io/server_components.html>

<https://docs.chef.io/install_server.html>

## Hosted chef

<https://manage.chef.io/login>

<https://docs.chef.io/manage.html>



## Knife

<https://docs.chef.io/knife_cookbook.html#upload>



Upload local cookbooks to chef server

# 10-4 Uploading Cookbooks

# this command list is provided to accompany the demos in the Chef Fundamentals Udemy course

# commands that involve editing files assume nano as the text editor

# run these commands on your local machine after configuring the ~/chef-repo directory

# and downloading the /cookbooks to your local machine

cd ~/chef-repo

ls

ls cookbooks/

ls -a

ls .chef/

cat .chef/knife.rb

knife --help

knife client list

knife ssl check

knife cookbook --help

knife cookbook list

knife cookbook upload workstation

knife cookbook upload apache

knife cookbook list

## Bootstrap

<https://docs.chef.io/install_bootstrap.html>



Bootstrap example with run list:

>knife bootstrap localhost --ssh-port <ssh port if not default> --ssh-user centos --sudo --identity-file /PATH/TO/SSHKEY -N load-balancer --run-list "recipe[myhaproxy]"

## Supermarket and cookbook wrapping

# 10-6 Lab: Bootstrap a Webserver

# this command list is provided to accompany the demos in the Chef Fundamentals Udemy course

# commands that involve editing files assume nano as the text editor

# run these commands on your local machine after copying new Vagrantfile to ~/chef-repo and running 'vagrant up'

cd ~/chef-repo

vagrant status

vagrant ssh-config

knife bootstrap localhost --ssh-port PORT --ssh-user vagrant --sudo --identity-file /PATH/TO/KEY -N web1

knife node list

knife node show web1

knife node run\_list add web1 "recipe[workstation],recipe[apache]"

knife node show web1

vagrant ssh web1

# run these commands on the web1 vagrant instance

vagrant@web1$ sudo chef-client

vagrant@web1$ curl localhost

#For AWS:

>knife bootstrap centos01 –ssh-port 22 –ssh-user centos –sudo –identify-file ~/.ssh/desktop –N web1

Supermarket: <https://docs.chef.io/supermarket.html>

haproxy cookbook from supermarket: <https://supermarket.chef.io/cookbooks/haproxy>



Wrapping cookbooks: <https://blog.chef.io/2013/12/03/doing-wrapper-cookbooks-right/>

Attribute precedence: <https://learn.chef.io/skills/beyond-essentials-4>



chef generate cookbook cookbooks/myhaproxy

# chef-repo/cookbooks/myhaproxy/metadata.rb

name 'myhaproxy'

maintainer 'The Authors'

maintainer\_email 'you@example.com'

license 'all\_rights'

description 'Installs/Configures myhaproxy'

long\_description 'Installs/Configures myhaproxy'

version '0.1.0'

**depends 'haproxy', '= 2.0.0'**

Update files:

## Berkshelf

# chef-repo/cookbooks/myhaproxy/recipes/default.rb

#

# Cookbook Name:: myhaproxy

# Recipe:: default

#

# Copyright (c) 2016 The Authors, All Rights Reserved.

servers = ["centos01"]

ips=["172.31.46.153"]

#servers.each\_with\_index {|name, index| ips[index] = Resolv.getaddress(name) }

node.default['haproxy']['members'] = [

{

"hostname" => servers[0],

"ipaddress" => ips[0],

"port" => 80,

"ssl\_port" => 80

}]

node.default['haproxy']['admin']['port'] = 8080 #default was 22001, didn’t work

include\_recipe "haproxy::manual"

About berkshelf: <https://docs.chef.io/berkshelf.html>



11-4 Berkshelf

# this command list is provided to accompany the demos in the Chef Fundamentals Udemy course

# commands that involve editing files assume nano as the text editor

# run these commands on your local machine after copying new Vagrantfile to ~/chef-repo and running 'vagrant up'

cd ~/chef-repo

cd cookbooks/myhaproxy

tree

cat Berksfile

berks install

cd ~/.berkshelf/cookbooks

ls

cd ~/chef-repo/cookbooks/myhaproxy

cat Berksfile.lock

berks upload

nano recipes/default.rb

berks upload

knife cookbook list

berks upload --force

## Knife node command

<https://docs.chef.io/knife_node.html>

cd ~/chef-repo

knife node --help

knife node list

knife cookbook list

knife cookbook show myhaproxy

knife cookbook show myhaproxy 0.2.0

knife node show --help

knife node show web1

knife node show web1 -a ipaddress

knife node show web1 -a hostname

knife node show load-balancer -a haproxy

nano cookbooks/myhaproxy/recipes/default.rb

knife node show load-balancer -a haproxy.members

knife node show load-balancer -a cpu.0.mhz

knife node show load-balancer -a memory.total

knife node show load-balancer -a cookbooks

# Roles

About roles: <https://docs.chef.io/roles.html>

Managing roles: <https://docs.chef.io/server_manage_roles.html>

 File web.rb:

# 13-3 Methods for Creating Roles

# this command list is provided to accompany the demos in the Chef Fundamentals Udemy course

# commands that involve editing files assume nano as the text editor

# run these commands on your local machine after copying new Vagrantfile to ~/chef-repo and running 'vagrant up'

cd ~/chef-repo

knife role --help

knife role list

ls roles/

cat roles/starter.rb

nano roles/web.rb

knife role from file roles/web.rb

knife role list

knife role show web

knife node show web1

knife node run\_list set web1 "role[web]"

knife node show web1

knife node run\_list set web2 "role[web]"

knife node show web2

vagrant ssh web1

knife node show web1

vagrant ssh web2

knife node show web2

# run these commands on the web1, web2 instances

$ sudo chef-client

$ exit

# run these commands on the web2 vagrant instance

vagrant@web2$ sudo chef-client

vagrant@web2$ exit

name 'web'

description 'Web Server Role'

run\_list 'recipe[workstation]','recipe[apache]'

Knife ssh

<https://docs.chef.io/knife_ssh.html>

# 13-5 Converge Using knife ssh

# this command list is provided to accompany the demos in the Chef Fundamentals Udemy course

# commands that involve editing files assume nano as the text editor

# run these commands on your local machine after copying new Vagrantfile to ~/chef-repo and running 'vagrant up'

cd ~/chef-repo

knife winrm --help

knife ssh --help

vagrant ssh-config

knife ssh localhost 'sudo chef-client' --manual-list --ssh-port WEB1\_PORT --ssh-user vagrant --identity-file /PATH/TO/WEB1\_KEY/

knife ssh localhost 'sudo chef-client' -m -p WEB1\_PORT -x vagrant -i /PATH/TO/WEB1\_KEY/

knife ssh "\*:\*" -x chef -P chef "sudo chef-client"

knife ssh "\*:\*" -x chef -i /PATH/TO/AWS\_KEY "sudo chef-client"

knife ssh "role:web" -x chef -P chef "sudo chef-client"

# Knife search

Indexes: <https://docs.chef.io/chef_search.html>



# 14-3 Running Searches with knife

# this command list is provided to accompany the demos in the Chef Fundamentals Udemy course

# commands that involve editing files assume nano as the text editor

# run these commands on your local machine after copying new Vagrantfile to ~/chef-repo and running 'vagrant up'

cd ~/chef-repo

knife node show web1

knife node show web1 -a node

knife node show web1 -a ipaddress

knife node show web1 -a memory.total

knife search --help

knife search node "\*:\*"

knife search node "\*:\*" -a ipaddress

knife search node "name:web1"

knife search node "name:\*"

knife search node "name:web"

knife search node "name:web" -a ipaddress

knife search role "\*:\*"

knife search role "\*:\*" -a name

knife search node "role:web AND recipes:apache"

knife search node "role:web" -a ipaddress

Other resources