Chef

Configuration Management tool

Introduction 3

Chef Architecture 3

How to Install Chef Development Kit on Linux 4

Install chef server and chef management console 7

Understanding Chef Server, Chef Nodes and Chef Workstations 9

Chef Server 9

Chef Nodes 9

Chef Workstations 9

Understanding the Concepts of Cookbook 10

Recipes 10

Attributes 11

Files 11

Templates 11

Setup connection between workstation and chef-server 11

Create Simple Chef Cookbooks to Manage Infrastructure 12

How to Provision an instance in AWS EC2 14

# Introduction

Chef is a configuration management tool. Configuration management tools are to keep the servers in a desired state. These tools are used to check if the server is in a desired state or not, if not then configuration management tool brings that server in the desired state i.e. correct packages are installed, configuration files contain the expected values, files are having expected, right services are running, and so on.

Along with the Chef, there are other famous configuration management tools as well in this category Ansible, Puppet, and Salt. Chef is most widely used tool in this category. It is ruby based tool. Chef is used to automate common infrastructure tasks and express solutions to common infrastructure problems. Chef is an automation framework which helps to automate your infrastructure. Chef can automate how you build, deploy, and manage your infrastructure. Chef can integrate with cloud-based platforms such as Amazon Elastic Compute Cloud (AWS EC2) to automatically provision and configure new machines.

In short, Chef is an automation platform that configures and manages your infrastructure whether it is on-premises or in the cloud. You can use Chef to speed up application deployment, even creating a continual deployment pipeline. The key to Chef ’s power is that it turns infrastructure into code. Infrastructure as code means your infrastructure becomes versionable, testable, and repeatable as application code enabling you to automate the process of configuring, deploying and scaling servers and applications.

# Chef Architecture

This diagram shows how you develop, test, and deploy your Chef code.



The workstation (chef DK/chef Development Kit in the diagram) is the location from which users interact with Chef. On the workstation users author and test cookbooks using tools such as Test Kitchen and interact with the Chef server using the knife and chef command line tools.

Chef server is used as your foundation to create and manage flexible, dynamic infrastructure to manage thousands of nodes/clients.

Nodes are the machines—physical, virtual, cloud, and so on—that are under management by Chef. The chef-client is installed on each node.

# How to Install Chef Development Kit on Linux

You are having option to install chef-client alone or you can install chef development kit which comes with most of the chef tools. The Chef Development Kit includes all the basic tools you need to get started writing Chef code.

Chef Client (also known as the Chef Omnibus Installer) contains the core components of Chef needed to manage a server or workstation. The installer comprises the entire collection of things necessary to run Chef. Chef Client keeps core application scripts along with the necessary Ruby scripting engine. In production environments, Chef Client is installed on every system intended to be managed by Chef.

The Chef Development Kit is a superset of Chef Client. If you install the Chef Development Kit, there is no need to install the Chef Client. Chef Development Kit comes with all the components of Chef Client, plus additional tools developed by the Chef community, in one package.

To install the chef development kit, visit page <https://downloads.chef.io/chefdk/>

Select Linux distribution for on which you want to install the chef DK and download the package. Please use the below commands based on Linux distribution:

For Ubuntu 14.04

|  |
| --- |
| $ wget https://packages.chef.io/files/stable/chefdk/1.2.22/ubuntu/14.04/chefdk\_1.2.22-1\_amd64.deb |

Output:

|  |
| --- |
| ubuntu@ip-10-0-1-51:~$ wget https://packages.chef.io/files/stable/chefdk/1.2.22/ubuntu/14.04/chefdk\_1.2.22-1\_amd64.deb  --2017-03-08 04:59:25-- https://packages.chef.io/files/stable/chefdk/1.2.22/ubuntu/14.04/chefdk\_1.2.22-1\_amd64.deb  Resolving packages.chef.io (packages.chef.io)... 151.101.52.65  Connecting to packages.chef.io (packages.chef.io)|151.101.52.65|:443... connected.  HTTP request sent, awaiting response... 200 OK  Length: 105773606 (101M) [application/x-debian-package]  Saving to: ‘chefdk\_1.2.22-1\_amd64.deb’  100%[================================================================================================================================================================>] 105,773,606 22.3MB/s in 4.6s  2017-03-08 04:59:30 (21.7 MB/s) - ‘chefdk\_1.2.22-1\_amd64.deb’ saved [105773606/105773606]  ubuntu@ip-10-0-1-51:~$ ls -ltr chef\*  -rw-rw-r-- 1 ubuntu ubuntu 105773606 Feb 2 19:29 chefdk\_1.2.22-1\_amd64.deb  ubuntu@ip-10-0-1-51:~$ |

Similarly, For Red Hat Enterprise Linux 7

|  |
| --- |
| $ wget https://packages.chef.io/files/stable/chefdk/1.2.22/el/7/chefdk-1.2.22-1.el7.x86\_64.rpm |

Then install the downloaded package:

For Ubuntu 14.04

|  |
| --- |
| $ sudo dpkg -i chefdk\_1.2.22-1\_amd64.deb |

Output:

|  |
| --- |
| ubuntu@ip-10-0-1-51:~$ sudo dpkg -i chefdk\_1.2.22-1\_amd64.deb  Selecting previously unselected package chefdk.  (Reading database ... 57407 files and directories currently installed.)  Preparing to unpack chefdk\_1.2.22-1\_amd64.deb ...  Unpacking chefdk (1.2.22-1) ...  Setting up chefdk (1.2.22-1) ...  Thank you for installing Chef Development Kit!  ubuntu@ip-10-0-1-51:~$ |

Similarly, For Red Hat Enterprise Linux 7

|  |
| --- |
| $ sudo rpm -Uvh chefdk-1.2.22-1.el7.x86\_64.rpm |

The Chef Development Kit installer will automatically install Chef and Ruby in the /opt/ chefdk/embedded directory on your local machine. Neither of these directories are commonly present in the default environment’s PATH.

The Chef Development Kit includes a chef shell-init command to modify the current shell environment to use these paths. Assuming you are using the Linux default bash shell, run the following command to permanently enable this PATH setting:

|  |
| --- |
| $ echo 'eval "$(chef shell-init bash)"' >> ~/.bash\_profile  $ source $HOME/.bash\_profile |

**Verify Installation**

|  |
| --- |
| ubuntu@ip-10-0-1-51:/opt/chefdk/embedded$ chef-client --version  Chef: 12.18.31  ubuntu@ip-10-0-1-51:/opt/chefdk/embedded$ chef --version  Chef Development Kit Version: 1.2.22  chef-client version: 12.18.31  delivery version: master (0b746cafed65a9ea1a79de3cc546e7922de9187c)  berks version: 2017-03-08T05:15:01.539641 16651] 2017-03-08T05:15:01.539781 16651] 2017-03-08T05:15:01.539893 16651] 2017-03-08T05:15:01.539976 16651] 2017-03-08T05:15:01.570638 16651] 2017-03-08T05:15:01.570772 16651] 5.6.0  kitchen version: 1.15.0  ubuntu@ip-10-0-1-51:/opt/chefdk/embedded$  ubuntu@ip-10-0-1-51:/opt/chefdk/embedded$ which ruby  /opt/chefdk/embedded/bin/ruby  ubuntu@ip-10-0-1-51:/opt/chefdk/embedded$ ls /opt/chefdk/bin  berks chef-apply chef-shell chef-vault dco foodcritic kitchen ohai push-apply pushy-service-manager  chef chef-client chef-solo cookstyle delivery inspec knife print\_execution\_environment pushy-client rubocop  ubuntu@ip-10-0-1-51:/opt/chefdk/embedded$ |

# Install chef server and chef management console

We have already seen use of chef server in the chef architecture section. The Chef server acts as a hub of information. Cookbooks and policy settings are uploaded to the Chef server by chef developers from workstations.

Nodes get the latest data from chef-server and make the changes in the state of the node based on the data it got from chef-server. Chef server can be accessed through a web interface known as Chef management console. It can be used to manage data bags, attributes, run-lists, roles, environments, and cookbooks, and also to configure role-based access for users and groups.

To install chef-server on a Linux machine you should follow below:

1. Get the package from <https://downloads.chef.io/chef-server/> for your Linux distribution. You can also download through command line if you are not having GUI.

|  |
| --- |
| $ curl -LSO https://packages.chef.io/files/stable/chef-server/12.13.0/el/7/chef-server-core-12.13.0-1.el7.x86\_64.rpm |

1. Install the package

|  |
| --- |
| $ sudo rpm -Uvh chef-server-core-12.13.0-1.el7.x86\_64.rpm |

1. Start the services

|  |
| --- |
| $ sudo chef-server-ctl reconfigure |

1. Create administrator

|  |
| --- |
| $ chef-server-ctl user-create USER\_NAME FIRST\_NAME LAST\_NAME EMAIL 'PASSWORD' --filename FILE\_NAME |

Example:

|  |
| --- |
| $ sudo chef-server-ctl user-create chef\_admin\_user Nikhil Gupta coolnicks.nikhil@gmail.com 'my\_password' --filename chef\_key |

It will save will save the RSA private key to the file specified using --filename

1. Create organization

|  |
| --- |
| $ chef-server-ctl org-create short\_name 'full\_organization\_name' --association\_user user\_name --filename ORGANIZATION-validator.pem |

Example:

|  |
| --- |
| $ sudo chef-server-ctl org-create my\_org 'My Organization' --association\_user chef\_admin\_user --filename my\_org-validator.pem |

1. Install chef management console

You can use install sub-command with chef-server to install any required package from <https://packages.chef.io/>

|  |
| --- |
| $ sudo chef-server-ctl install chef-manage  $ sudo chef-server-ctl reconfigure  $ sudo chef-manage-ctl reconfigure --accept-license |

You can access chef management console using https://<host\_name/ip>



# Understanding Chef Server, Chef Nodes and Chef Workstations

## Chef Server

This is the central location that stores configuration recipes, cookbooks, and node and workstation definitions. It is the central machine that every other machine in the organization will use for deployment configuration. It handles connections and permissions from nodes. It keeps the data in organized way so that it can easily be pulled by clients. Web interface can also be installed on the chef server (which is known as chef management console) to manage or configure some details.

## Chef Nodes

Chef nodes are the deployment targets that are configured by Chef. Each node represents a separate, contained machine environment that can be on physical hardware, Virtual machine or Container. Chef client is installed on each node so nodes and chef server can communicate with each other. Nodes can be configured with chef as soon as it is online and connected to chef server.

Node’s only requirement is that it should have access to the network and can communicate with the chef server. Another requirement is that the user which will be used on these nodes through chef should have permission to install the packages, make config changes.

Each node communicates with the central server using an application called chef-client. This handles pulling data off of the server and executing the configuration steps necessary to get the node into its final state. The chef-client program and the chef server communicate through the use of RSA key-based authentication.

## Chef Workstations

Chef workstations are where Chef configuration details are created or edited. The configuration files are then pushed to the Chef server, where they will be available to deploy to any nodes. There can be multiple workstations. Work stations are used to write chef cookbooks, recipes and these can be tested to workstation. Once you are fine with the configuration files, then you can push those to chef server.

A workstation has the tools necessary to create and modify configuration details for any of the available nodes and can communicate with the chef server to make these available. An important tool to manage chef on a workstation is called knife. Knife acts as a gateway in which you can configure anything that would be stored on the server. It can manage nodes and configurations and can generally be used to access the server in a "chef-specific" way. You can connect to the server and make the changes using ssh but this is not the chef way to do the things.

# Understanding the Concepts of Cookbook

Cookbooks are the configuration units that allow us to configure and perform specific tasks within Chef on our remote nodes. Cookbooks brings a node in desired state. Chef uses cookbooks to perform the tasks to bring a node in the desired state. Cookbooks are usually used to handle one specific service, application, or functionality for example to install and configure database, install a web server and configure it, host a website etc.

Cookbooks are created on the workstation and then uploaded to a Chef server. Cookbooks have recipes which are assigned to nodes as part of the node's "run-list". A run-list is a sequential list of recipes and roles that are run on a node by chef-client in order to bring the node into desired state. Cookbooks have a directory structure which contains different files. We will discuss some important files and directories.

## Recipes

A recipe is the main part of the cookbook. A cookbook can contain more than one recipe. Cookbook can also depend on outside recipes. State of different resources is defined in recipes. For example, X package should be installed or Y service should be running. A recipe can be run on a client/node to bring that node in a state which is defined in the recipe. If a resource is in desired state on the client, then recipe will go and check the next resource. If any resource is not in desired state i.e. package is not installed, service is not running then recipe will bring that resource in desired state. There are many resources. Generally used resources are:

* **package**: Used to manage packages on a node
* **service**: Used to manage services on a node
* **user**: Manage users on the node
* **group**: Manage groups
* **template**: Manage files with embedded ruby templates
* **cookbook\_file:** Transfer files from the files subdirectory in the cookbook to a location on the node
* **file**: Manage contents of a file on node
* **directory**: Manage directories on node
* **execute**: Execute a command on the node
* **cron**: Edit an existing cron file on the node

## Attributes

Attributes in key-value pairs which you can use in your cookbook. An attribute is a specific detail about a node, such as an IP address, a host name, a list of loaded kernel modules, the version(s) of available programming languages that are available, and so on. An attribute may be unique to a specific node or it can be identical across every node in the organization. An attribute can be defined in a attribute file and then used to override the default settings on a node. When a cookbook is loaded during a chef-client run, these attributes are compared to the attributes that are already present on the node.

## Files

The files subdirectory within the cookbook contains any static files that we will be placing on the nodes that use the cookbook. For instance, any simple configuration files that we are not likely to modify can be placed, in their entirety, in the files subdirectory. A recipe can then declare a resource that moves the files from that directory into their final location on the node.

## Templates

Templates are similar to files, but they are not static. Template files end with the .erb extension, meaning that they contain embedded Ruby. These are mainly used to substitute attribute values into the file to create the final file version that will be placed on the node.

For example, if we have an attribute that defines the default port for a service, the template file can call to insert the attribute at the point in the file where the port is declared. Using this technique, you can easily create configuration files, while keeping the actual variables that you wish to change elsewhere.

# Setup connection between workstation and chef-server

To access chef-server from chef workstation, you will have to do the following setup. Please install chefDK on workstation. Download chef-repo from github, create .chef directory and put knife.rb, my\_org-validator.pem and user pem file at this location. Then

|  |
| --- |
| $ git clone <https://github.com/opscode/chef-repo.git>  $ mkdir -p ~/chef-repo/.chef  $ cd chef-repo/ .chef/  $ knife ssl fetch  WARNING: Certificates from chef.domain.com will be fetched and placed in your trusted\_cert  directory (/Users/ngupta9/chef-repo/.chef/trusted\_certs).  Knife has no means to verify these are the correct certificates. You should  verify the authenticity of these certificates after downloading.  Adding certificate for ip-10-0-1-37\_us-west-2\_compute\_internal in /Users/ngupta9/chef-repo/.chef/trusted\_certs/ip-10-0-1-37\_us-west-2\_compute\_internal.crt |

# Create Simple Chef Cookbooks to Manage Infrastructure

To demonstrate some of the work flow involved in working with cookbooks, we will create a cookbook of our own. This will be a very simple cookbook that installs and configures the Nginx web server on our node.

Create a directory and generate a cookbook template

|  |
| --- |
| ubuntu@ip-10-0-1-51:~$ mkdir -p ~/chef-repo  ubuntu@ip-10-0-1-51:~$ cd ~/chef-repo  ubuntu@ip-10-0-1-51:~/chef-repo$ chef generate cookbook nginx  Generating cookbook nginx  - Ensuring correct cookbook file content  - Committing cookbook files to git  - Ensuring delivery configuration  - Ensuring correct delivery build cookbook content  - Adding delivery configuration to feature branch  - Adding build cookbook to feature branch  - Merging delivery content feature branch to master  Your cookbook is ready. Type `cd nginx` to enter it.  There are several commands you can run to get started locally developing and testing your cookbook.  Type `delivery local --help` to see a full list.  Why not start by writing a test? Tests for the default recipe are stored at:  test/smoke/default/default\_test.rb  If you'd prefer to dive right in, the default recipe can be found at:  recipes/default.rb  ubuntu@ip-10-0-1-51:~/chef-repo$  ubuntu@ip-10-0-1-51:~/chef-repo$ cd nginx/  ubuntu@ip-10-0-1-51:~/chef-repo/nginx$ ls -ltr  total 28  -rw-rw-r-- 1 ubuntu ubuntu 740 Mar 17 14:34 metadata.rb  -rw-rw-r-- 1 ubuntu ubuntu 53 Mar 17 14:34 README.md  -rw-rw-r-- 1 ubuntu ubuntu 1133 Mar 17 14:34 chefignore  -rw-rw-r-- 1 ubuntu ubuntu 47 Mar 17 14:34 Berksfile  drwxrwxr-x 3 ubuntu ubuntu 4096 Mar 17 14:34 test  drwxrwxr-x 3 ubuntu ubuntu 4096 Mar 17 14:34 spec  drwxrwxr-x 2 ubuntu ubuntu 4096 Mar 17 14:34 recipes  ubuntu@ip-10-0-1-51:~/chef-repo/nginx$ |

Then write the code in recipe to install Nginx package, start the service once package is installed and then create static html file to deploy on the node.

|  |
| --- |
| ubuntu@ip-10-0-1-51:~/chef-repo/nginx/recipes$ cat default.rb  #  # Cookbook:: nginx  # Recipe:: default  #  # Copyright:: 2017, The Authors, All Rights Reserved.  execute "apt-get update" do  command "apt-get update"  end  package 'nginx' do  action :install  end  service 'nginx' do  action [ :enable, :start ]  end  cookbook\_file "/usr/share/nginx/www/index.html" do  source "index.html"  mode "0644"  end  ubuntu@ip-10-0-1-51:~/chef-repo/nginx/recipes$ |

Then you can run this cookbook in local mode or you can upload the cookbook to chef-server and then update the run list on node.

# How to Provision an instance in AWS EC2

You can create an AWS EC2 instance with the help of chef. For this you will have to install a gem

|  |
| --- |
| $ gem install knife-ec2  Fetching: knife-ec2-0.15.0.gem (100%)  Successfully installed knife-ec2-0.15.0  1 gem installed |

Now, you can create an EC2 instance. You will have to provide AMI name, instance type, storage type, key pair, region and zone.

You can run the below command to create an instance.

|  |
| --- |
| $ knife ec2 server create -I ami-1ed88f69 -f t2.small -S my\_key -i ~/.ssh/my\_key.pem --ssh-user ubuntu --region eu-west-1 -Z eu-west-1a -r "role[ubuntu]" |

Where,

* -I is the AMI ID you selected earlier
* -f is the [Amazon EC2 instance type](http://aws.amazon.com/ec2/instance-types/) (see Model)
* -S is the name you gave to the SSH key pair generated in the AWS management console
* -i points to the private key file of that SSH key pair as downloaded when the key pair was created in the AWS management console
* --ssh-user the official Ubuntu EC2 AMIs use ubuntu as the default user
* --region eu-west-1 If you want your instances to be deployed in any specific Amazon AWS region, add this parameter and the desired region
* -Z eu-west-1a is the availability zone within your region
* -r "role[ubuntu]" is the run\_list, which you want to associate with the newly created node. You can put any roles and recipes you like here

You can check the list of EC2 instances managed with chef using below

|  |
| --- |
| $ knife ec2 server list --region eu-west-1 |

Please make sure you use the correct --region parameter

If you want to get rid of an instance (terminate instance and delete the corresponding Chef node), it’s as easy as:

|  |
| --- |
| $ knife ec2 server delete i-XXXXXXXX --region eu-west-1  $ knife node delete i-XXXXXXXX |

Where, i-XXXXXXXX is the ID of the instance as found in the AWS management console or a knife ec2 server list call.