CHEF by OpsCode

(A Software Configuration Management Tool)

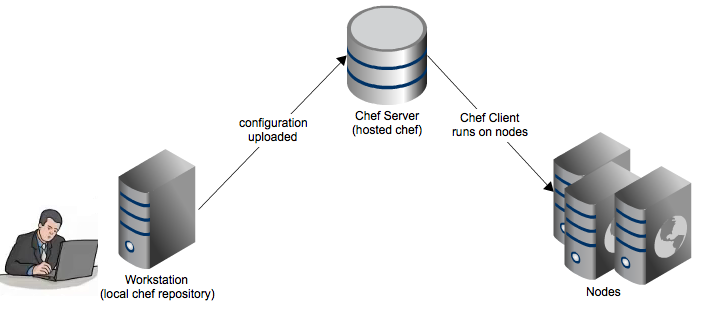
CHEF Overview:

Chef is a framework that makes it easy to manage your infrastructure. Chef was initially written in Ruby, but the latest version is a mixture of Erlang and Ruby. A single chef server can handle upto 10,000 nodes.

Chef can be mainly broken down into three components.

* Server: The chef server holds the configuration data for each and every node registered with it.
* Workstation: A workstation basically holds the local chef repository.
* A node is a client that is registered with the chef server. It has an agent known as chef client installed on it.

## **CHEF Arcitecture**



Why Chef?

As explained previously, chef gives your infrastructure the flexibility, speed, and efficiency you have always wanted. Automation through chef can provide the speed and agility needed by business today to compete. Chef can be used to quickly provide IT solutions and repeatable configurations with minimal human intervention.

Automating your infrastructure with chef could help you to deploy features in minutes rather than days. Chef can manage any number of servers without much complexity, and thus it helps you in managing your infrastructure easily, at less cost, and while avoiding human errors.

Chef helps your enterprise in moving to public clouds and complements the public cloud model by providing integrations with major public cloud providers.

Core Principles of Chef

Chef is a highly configurable and extensible tool with immense power in the hands of administrators to automate their infrastructure. It provides flexibility, agility, and speed to administrators, and they can leverage the tool the way they best deem fit in their scenarios. The main principles on which chef works are

* Idempotence
* Idempotence means that a chef recipe can run multiple times on the same system and the return will be identical. Chef ensures that the configuration changes to the end system (node) are done when the underlying configuration differs from the desired state and no changes are made to the system if they are not needed. Thus, administrators can define the end configurations, and chef will ensure that the nodes have the desired configuration on them
* Thick client, thin server
* Order of execution

Workstation

A workstation is a system that is used to manage chef. There can be multiple workstations for a single chef server. A workstation has the following functionalities.

* + Developing cookbooks and recipes.
  + Managing nodes.
  + Synchronizing the chef repository.
  + Uploading cookbook and other items to the chef server

There are mainly two important components of a workstation.

1. **Knife**: A command line tool used to interact with the chef server. The complete management of the chef server is done using knife. Some of the functions of knife include
   1. Managing nodes
   2. Uploading cook books and recipes
   3. Managing roles and environments
2. **Local chef repository**: Chef repository is a repository where everything related to the chef server/nodes is stored.

Nodes

A **node** can be termed a “virtual” or a “physical” server that is managed by chef. A node can also be on the cloud. A node needs to have an agent, known as **chef client**, installed on it. The agent is used to interact with the chef server.

**Ohai** is a built-in tool that comes with chef and is used to provide node attributes to the chef client so that a node can be configured. There are basically two types of nodes that chef can manage.

* Cloud-based: It is basically a node that is hosted on any of the cloud providers (e.g., Amazon or Windows Azure). There is a chef CLI (command line interface) known as knife which can be used to create instances on the cloud. Once deployed, these nodes can be managed with the help of chef.
* Physical: It can be hardware or a virtual machine that exists in our own environment.

There are mainly two important components of a node.

1. **Chef client**: An agent that runs on each node. The agent contacts the chef server and pulls the configuration that needs to be done on the node. Its main functions include a. Registering the node with the chef server. b. Downloading the required cookbook in the local cache. c. Compiling the required recipes. d. Configuring the node and bringing it to the expected state.
2. Ohai: Chef Client requires some information about the node whenever it runs. Ohai is a built-in tool that comes with chef and is used to detect certain attributes of that particular node and then provide them to the chef client whenever required. Ohai can also be used as a stand-alone component for discovery purposes. Ohai can provide a variety of details from networking to platform information.

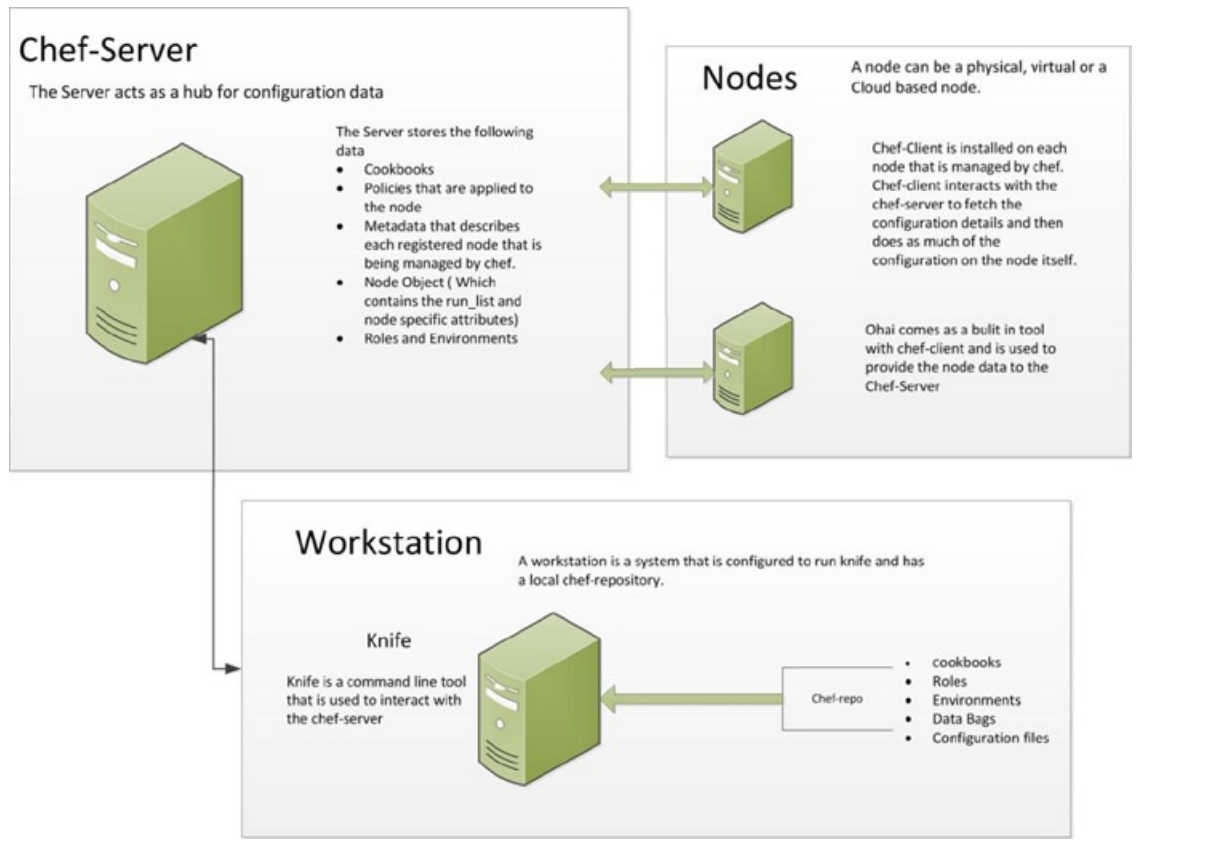
Chef Server

There is a centrally located server which holds all the data related to the chef server; this data includes everything related to the server (i.e., cookbooks, the node object, and metadata for each and every node registered to the chef server). The agent (chef client) runs on each and every node, and it gets the configuration data from the server and then applies the configuration to a particular node. This approach is quite helpful in distributing the effort throughout the organization rather than on a single server.

There are three different types of chef server.

* + Enterprise chef
  + Open source chef
  + Chef solo

Chef components in detail

****

## Chef Server Setup

### Pre-requisites:

#### System Requirements

Some of the important system requirements that need to be fulfilled before we install the chef server are

* Users: Chef server requires a local user and group to be created. It will create them automatically if proper privileges are given, but if we don’t have a restricted access to the environment then we need to create them manually.
* FQDN: The server should have a complete and fully qualified domain name (FQDN), and it should be resolvable. If we are working in a production environment we should go for a DNS (Domain Name System) entry.
* Git: Git must be installed on the server so that it is able to maintain the revisions of internal services.
* NTP: As the chef server is sensitive to click drift, the server should be connected to an NTP server.
* Apache Qpid: The daemon should be disabled on CentOS and Red Hat systems.
* Make sure your firewall is configured properly.
* Provides inbound access (including firewall) on port 443 (HTTPS)
* SELinux disabled or set to permissive mode
* You may also need to open port 22 (SSH)

#### Hardware Prerequisite

* + 4GB RAM
  + Cores—4 with 2.0GHz Intel/AMD CPUs
  + Disk space—5GB in /opt and 5GB free in /var
  + hostname should be set and can be accessed from your workstation and nodes
    - sudo hostnamectl set-hostname <hostname>

## Steps to Installation

### Hardware Checking:

Note: All the commands should execute as super user (root)

* Checking available : df -h
* cd /usr/sbin
* is connected to NTP :

./ntpq

* Has Apache qpid Disabled

rpm -qa | grep qpid

* Provides inbound access (including firewall) on port 443 (HTTPS).

ps -ef | grep 443

* has SELinux disabled or set to permissive mode.

cd /usr/sbin

./getenforce

./setenforce Permissive

* Enable port#22

vi /etc/ssh/sshd\_config  (Remove hash #port 22)

* Restart sshd:

service sshd restart

### Download below three rpm files

1. chef-server-core-12.8.0-1.el7.x86\_64.rpm (<https://downloads.chef.io/chef-server/redhat/>)
2. chef-manage-2.4.1-1.el7.x86\_64.rpm (<https://downloads.chef.io/chef-manage/>)
3. opscode-reporting-2.4.1-1.el7.x86\_64.rpm ([https://downloads.chef.io/reporting/](https://downloads.chef.io/chef-manage/))

Step1

* Execute below command to install chef server

rpm -ivh /<path to sw>/chef-server-core-12.8.0-1.el7.x86\_64.rpm

* Write the Chef server configuration file to ensure it includes your server's public hostname

*cd /etc/opscode*

*vi chef-server.rb*

*server\_name = "<hostname>”*

*api\_fqdn server\_name*

*bookshelf['vip'] = server\_name*

*nginx['url'] = "https://#{server\_name}"*

*nginx['server\_name'] = server\_name*

*nginx['ssl\_certificate'] = "/var/opt/opscode/nginx/ca/#{server\_name}.crt"*

*nginx['ssl\_certificate\_key'] = "/var/opt/opscode/nginx/ca/#{server\_name}.key"*

* Reconfigure the chef server

chef-server-ctl reconfigure

Step2

* Execute below commands to install chef manage
* chef-server-ctl install chef-manage --path /scratch/chef-manage-2.4.1-1.el7.x86\_64.rpm
* chef-server-ctl reconfigure
* chef-manage-ctl reconfigure

Note: read the licence and press "q" and say yes

#### Step3

* Execute below commands to install opscode reporting

* chef-server-ctl install opscode-reporting --path /scratch/opscode-reporting-2.4.1-1.el7.x86\_64.rpm
* chef-server-ctl reconfigure
* opscode-reporting-ctl reconfigure

 If Site is not opened on the browser then add the ip address and hostname of the VM in below file on your system.

C:\Windows\System32\drivers\etc\hosts

Stop the firewall

service iptables stop

### Creating Admin User

chef-server-ctl user-create ADMIN\_USER\_NAME ADMIN\_FIRST\_NAME ADMIN\_LAST\_NAME ADMIN\_EMAIL ADMIN\_PASSWORD --filename ADMIN\_USER\_NAME.pem

e.g: chef-server-ctl user-create chefadmin Chef Admin [chefadmin@aaa.com](mailto:chefadmin@aaa.com) Welcome1 --filename chefadmin.pem

Creating Organizaiton

chef-server-ctl org-create ORG\_SHORT\_NAME "ORG\_LONG\_NAME" --association\_user ADMIN\_USER\_NAME

### Testing Installation of chef server

* Execute below command to test the installation

chef-server-ctl test

## File System locations

Chef server uses the following file locations for installing chef.

* + **/opt/Chef-server**: This directory is used for Installation.
  + **/etc/Chef-server**: This directory is used for storing the key files and the API configuration.
  + **/var/opt/Chef-server**: All the services in chef are here.
  + **/var/log/Chef-server**: This directory is used for storing the logs.

## Workstation Setup

* First and foremost install the ChefDK on your workstation by the downloading the software from the URL : <https://downloads.chef.io/chefdk>
* Once install ChefDK then download the **“Starter Kit”** from the Cher Server 🡪 Administration tab 🡪Click on Org name🡪Select Start kit and download
* Once downloaded the **Starter Kit** then extract it and execute below commands from the extracted folder i.e., **“chef-repo”**
* knife ssl fetch
* knife ssl check
  + If you get ***connection established successfully*** message, then you are done with workstation setup.

## Manage Nodes

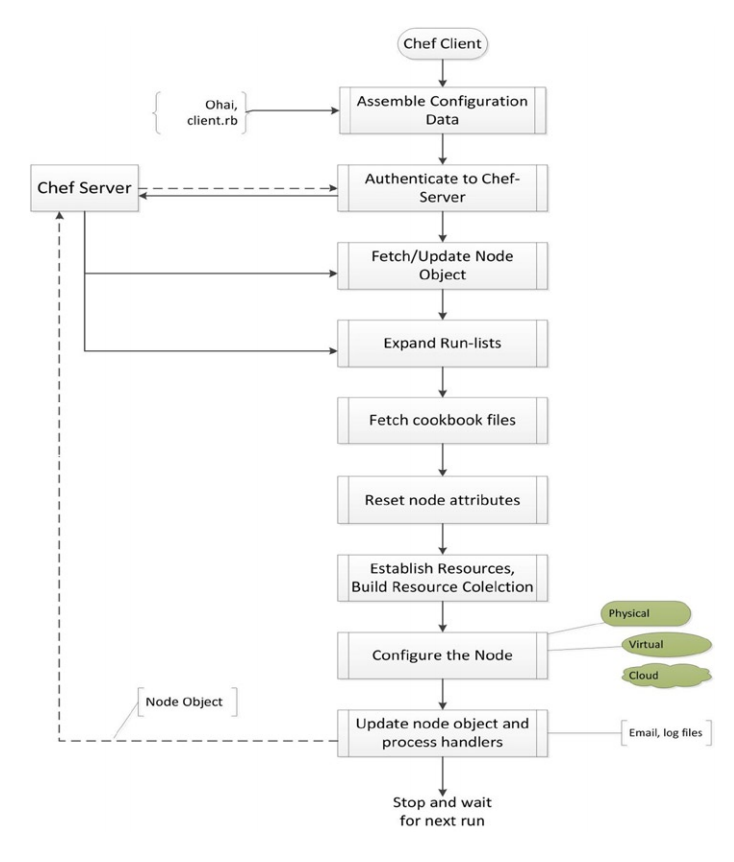
Nodes can be managed in chef using various approaches. They can be managed using knife or using the management console that chef provides.

* + We can use knife to manage the nodes. Managing covers creating, editing, tagging, listing, and so on.
  + We can use knife plug-ins to provision nodes.
  + We can also manage nodes using the management console

### Chef Client

Chef client is an agent installed on the node, and with its help, the nodes communicate with the chef server. Whenever chef client is run on a node, it follows certain steps

1. Get configuration details
2. Authenticate with chef server
3. Rebuild on the node object
4. Expand the run list
5. Download the cookbooks on the node
6. Reset node attributes
7. Identify resources
8. Configure the node
9. Update the node object



### Cookbook

A cookbook is the fundamental unit of configuration and policy distribution. A cookbook defines a scenario and contains everything that is required to support that scenario:

* Recipes that specify the resources to use and the order in which they are to be applied
* Attribute values
* File distributions
* Templates
* Extensions to Chef, such as custom resources and libraries

### Recipe:

A recipe is the most fundamental configuration element within the organization. A recipe:

* Is authored using Ruby, which is a programming language designed to read and behave in a predictable manner
* Is mostly a collection of resources, defined using patterns (resource names, attribute-value pairs, and actions); helper code is added around this using Ruby, when needed
* Must define everything that is required to configure part of a system
* Must be stored in a cookbook
* May be included in a recipe
* May use the results of a search query and read the contents of a data bag (including an encrypted data bag)
* May have a dependency on one (or more) recipes
* May tag a node to facilitate the creation of arbitrary groupings
* Must be added to a run-list before it can be used by the chef-client
* Is always executed in the same order as listed in a run-list

### Resource:

A resource is a statement of configuration policy that:

* Describes the desired state for a configuration item
* Declares the steps needed to bring that item to the desired state
* Specifies a resource type—such as package, template, or service
* Lists additional details (also known as resource properties), as necessary
* Are grouped into recipes, which describe working configurations

### Attribute:

An attribute is a specific detail about a node. Attributes are used by the chef-client to understand:

* The current state of the node
* What the state of the node was at the end of the previous chef-client run
* What the state of the node should be at the end of the current chef-client run

Attributes are defined by:

* The state of the node itself
* Cookbooks (in attribute files and/or recipes)
* Roles
* Environments

#### Attribute Precedence level

Attributes are always applied by the chef-client in the following order:

1. A default attribute located in a cookbook attribute file
2. A default attribute located in a recipe
3. A default attribute located in an environment
4. A default attribute located in a role
5. A force\_default attribute located in a cookbook attribute file
6. A force\_default attribute located in a recipe
7. A normal attribute located in a cookbook attribute file
8. A normal attribute located in a recipe
9. An override attribute located in a cookbook attribute file
10. An override attribute located in a recipe
11. An override attribute located in a role
12. An override attribute located in an environment
13. A force\_override attribute located in a cookbook attribute file
14. A force\_override attribute located in a recipe
15. An automatic attribute identified by Ohai at the start of the chef-client run

where the last attribute in the list is the one that is applied to the node.

### Working with Knife

Knife executes its functions from a workstation and is used to interact with the chef server and also with your infrastructure. The interaction with the workstation and the chef server is done using the REST API that is used by a chef client. The workstation configuration includes knife configuration, but if we want to change or modify anything, we can directly do it in the knife configuration file.

### Boot strapping the node

A bootstrap is a process by which we install chef client on a target node. This command takes the IP (Internet provider) or the FQDN (fully qualified domain name) of the node as an input and installs the chef client package on it. Syntax as follows

**Linux:**

knife bootstrap client\_node\_host\_name.domain.com --ssh-user username --ssh-password 'XXXXXX' --sudo --use-sudo-password --node-name chef-client-node-name --run-list 'recipe[hello\_chef\_server]'

**Windows:**

knife bootstrap windows winrm address\_of\_the \_node --winrm-user domainname/username --winrm-password XXXXX --node-name nodename

Cookbook

A cookbook is a basic unit of configuration and policy definition in chef. A cookbook essentially defines a complete scenario.

A cookbook contains all the components that are required to support the installation and configuration of an application or component, including

Default Cookbook Structure :

├───attributes

├───definitions

├───files

│ └───default

├───libraries

├───providers

├───recipes

├───resources

└───templates

└───default

* **Update your node**

Linux:

knife ssh client\_node\_host\_name.domain.com 'sudo chef-client' --manual-list --ssh-user username --ssh-password 'XXXXXXX’

Windows:

knife winrm den02chm.us.oracle.com chef-client --manual-list --winrm-user domainname/username --winrm-password XXXXX

* **Add recipes to run list**
  + knife node run\_list add client\_node\_host\_name.domain.com 'recipe[hello\_chef\_server],recipe[COOKBOOK::RECIPE],COOKBOOK::RECIPE,role[NAME]'