Chef Intro

Raghuram PSS

Chef is a configuration management technology developed by **Opscode**
Manages infrastructure on physical or virtual machines.
An open source developed using **Ruby**, which helps in managing complex infrastructure on the fly.
Chef helps you express your infrastructure policy – how your software is delivered and maintained on your servers – as code.
When infrastructure as code, it becomes more maintainable, versionable, testable, and collaborative

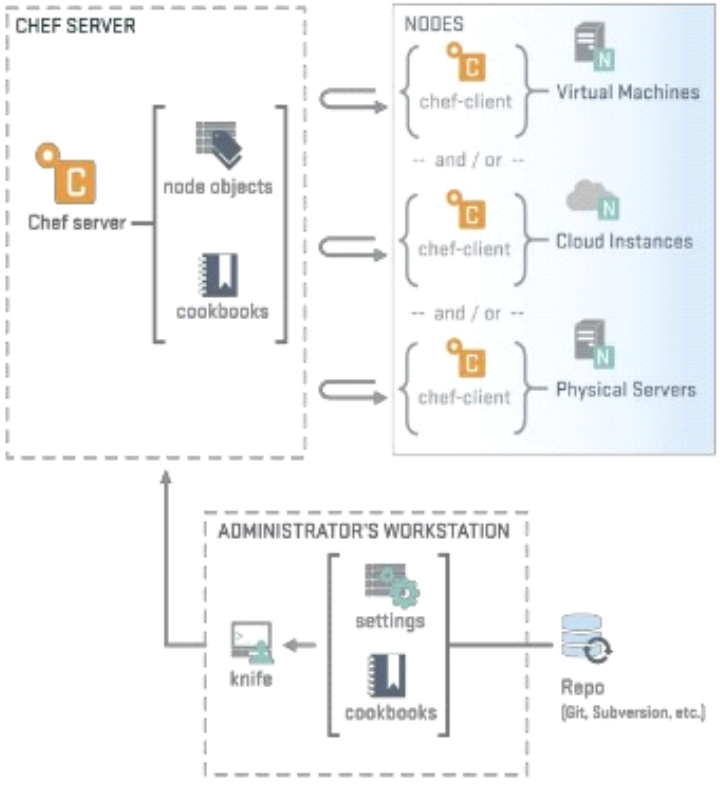
Features:

- Chef uses popular Ruby language to create a domain-specific language.
- Chef does not make assumptions on the current status of a node. It uses its mechanisms to get the current status of machine.
- Chef is ideal for deploying and managing the cloud server, storage, and software.
- Using the knife utility, it can be easily integrated with any of the cloud technologies. It is the best tool for an organizat ion that wishes to distribute its infrastructure on multi-cloud environment.

Chef Architecture

- Chef has three main components for it's overall Chef architecture:
  - Admin Workstation
  - Chef Server
  - Nodes
- The nodes communicate with the Chef server over HTTP(S) using the chef-client script
- The chef-client script is responsible for downloading and applying run-list along with any cookbooks and config data it needs
- The admin workstation also communicates with the Chef server using HTTP(S)
- The workstation is where a system admin will use the CLI utilities to interact with the data stored in the Chef server and modify any data, performs search and interact with nodes through the knife tool
- Chef also presents a web-based GUI for modifying system data

74



Chef building blocks

Resource

It is the basic component of a recipe used to manage the infrastructure with different kind of states. There can be multiple resources in a recipe, which will help in configuring and managing the infrastructure.

Package	Manages the packages on a node
Service	Manages the services on a node
User	Manages the users on the node
Group	Manages groups
Template	Manages the files with embedded Ruby template
cookbook_file	Transfers the files from the files subdirectory in the cookbook to a location on the node
File	Manages the contents of a file on the node
Directory	Manages the directories on the node
Execute	Executes a command on the node
Cron	Edits an existing cron file on the node

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

Resource Syntax:

```
type 'name' do
  attribute 'value'
  action :type_of_action
end
```

Recipe

- It can be defined as a collection of attributes which are used to manage the infrastructure.
- These attributes which are present in the recipe are used to change the existing state or setting a particular infrastructure node.
- They are loaded during Chef client run and comparted with the existing attribute of the node (machine).
- It then gets to the status which is defined in the node resource of the recipe. It is the main workhorse of the cookbook

Cookbook

- A cookbook is a collection of recipes.
- They are the basic building blocks which get uploaded to Chef server.
- When Chef run takes place, it ensures that the recipes present inside it gets a given infrastructure to the desired state as listed in the recipe.

Knife

- knife is the command-line tool that provides an interface between your workstation and the Chef server.
- knife enables you to upload your cookbooks to the Chef server and work with nodes, the servers that you manage.
- Knife helps users to manage
    - Nodes
    - Cookbooks and Recipes roles
    - Stores of JSON data (data bags), including encrypted data
    - Environments
    - Cloud resources, including provisioning
    - The installation of Chef-client on management workstations.

Node

- Each node stores its own private key locally
- This private key is generated as part of bootstrap process that initially installs the Chef-client on the node
- The first time Chef-client runs on the node, it uses Chef-validator to authenticate.

- On each subsequent run, it uses the private key generated for that client by the Chef server.

The Chef server stores cookbooks, the policies that are applied to nodes, and metadata that describes each registered node that is being managed by the chef-client

knife is a command-line tool that provides an interface between a local chef-repo and the Chef server. knife helps users to manage: Nodes. Cookbooks and recipes.

Chef Workstation Installation

Raghuram PSS

The normal Chef workflow starts from your workstation.
You use the **Chef Development Kit**, or **Chef DK**, to write and verify your configuration policy by writing Chef code.
The following process installs…

- Chef Development Kit
- Chef-client
- Embedded version of Ruby
- RubyGems
- OpenSSL
- Tools
    - Kitchen
    - Berkshelf
    - ChefSpec
    - Ohai - System profiler

Workstation responsibilities:

- Developing and testing cookbooks and recipes
- Testing Chef code
- Keeping the chef-repo synchronized with version source control
- Configuring organizational policy, including defining roles and environments, and ensuring that critical data is stored in data bags
- Interacting with nodes, as (or when) required, such as performing a bootstrap operation

Chef Development Kit - Widows	
Go to https://downloads.chef.io/chefdk	File "chefdk-2.0.28-1-x86.msi" gets downloaded
Click on the .msi file to install Chef Development Kit	
Run command prompt as administrator	
Knife --version # will show the knife version	Check the installation
Chef Development Kit - Ubuntu	
Create an instance (Instance type: t2.micro)	
apt-get update	
curl https://omnitruck.chef.io/install.sh | sudo bash -s -- -P chefdk -c stable -v 0.16.28	Downloads and installs Chef DK
The above command installs these components… • Chef Development Kit Version: 0.16.28 • chef-client version: 12.12.15 • delivery version: master (921828facad8a8bbbd767368bfc72f19bd30e7bd) • berks version: 4.3.5 • kitchen version: 1.10.2	
Upload chef-repo	scp -i chef-server.pem chef-starter.zip ubuntu@13.126.210.113:~/data
Set up your working directory	su mkdir chef-repo cd chef-repo
Create the MOTD file	nano hello.rb
	file '/tmp/motd' do content 'hello world' end
Run the recipe file	chef-apply hello.rb

	# chef-client --local-mode hello.rb
See the hello.rb created in the given dir..	cd /tmp ls
Update the file manually and run the chef-apply hello.rb again	
To delete the file through recipe file	nano rm-hello.rb
	file '/tmp/motd' do action :delete End

Chef Knife

knife runs from a management workstation and sits in-between a Chef server and an organization's infrastructure.

knife is a command-line tool that provides an interface between a local chef-repo and the Chef server. knife helps users to manage:

- Nodes
- Cookbooks and recipes
- Roles, Environments, and Data Bags
- Resources within various cloud environments
- The installation of the chef-client onto nodes
- Searching of indexed data on the Chef server

Chef Server

Organizations	• Completely independent tenants of Enterprise Chef
	• Share nothing with other organizations
	• May represent different…
	  • Companies
	  • Business units
	  • departments

Environments	• Environments model the life-stages of your applications
	• Every Organization starts with a single environment (_default)
	• Environments to reflect your pattern and workflow …
	  • Development
	  • Test
	  • Staging
	  • Production
	  • Etc.,
	• Environments may include data attributes necessary for configuring your infrastructure
	  • The URL of your payment service's API (e.g. Can configure Sandbox PayPal environment for Dev, test and Staging environments and PayPal production for prod. Environment)
	  • The location of your package repository
	  • The version of the Chef configuration files that should be used

Roles	• The way to classify the servers in your infrastructure
	• Roles represent the types of servers in your infrastructure
	  • Load balancer
	  • Application server
	  • Database cache
	  • Database
	  • Monitoring
	• **Roles Define Policy**
	  • Roles may include a list of Chef configuration files that should be applied
	  ○ We call this list a Run List
	  • Roles may include data attributes necessary for configuring your infrastructure
	  ○ The port that the application server listens on
	  ○ A list of applications that should be deployed

Nodes	• Nodes make up the infrastructure
	• Nodes represent server in your infrastructure
	• Nodes may represent Physical servers or virtual servers
	• Nodes may represent hardware that you own or may represent compute instances in a public or private cloud.
	• Each node will
	  • Belong to one Organization
	  • Belong to one Environment
	  • Have zero or more Roles (e.g. application server or DB server or both, or zero roles)

	Nodes Adhere to a Policy
	• An application, the chef-client, runs on each node
	• Chef-client will
	  • Gather current system configuration
	  • Download the desired system configuration from the Chef server
	  • Configure the node such that it adheres to the policy

Hostnames	Ensure that all systems have properly configured hostnames. The hostname for the Chef server must be a FQDN, including the domain suffix, and must be resolvable. See Hostnames, FQDNs for more information
FQDNs	Ensure that all systems have a resolvable FQDN
NTP	Ensure that every server is connected to NTP; the Chef server is sensitive to clock drift
Mail Relay	The Chef server uses email to send notifications for various events; a local mail transfer agent should be installed and available to the Chef server
Cron	Periodic maintenance tasks are performed using cron
Git	git must be installed so that various internal services can confirm revisions
libfreetype and libpng	These libraries are required
Apache Qpid	This daemon must be disabled on CentOS and Red Hat systems
Required users	If the environment in which the Chef server will run has restrictions on the creation of local user and group accounts, ensure that the correct users and groups exist before reconfiguring
Firewalls and ports	If host-based firewalls (iptables, ufw, etc.) are being used, ensure that ports 80 and 443 are open. These ports are used by the nginx service
Hostname	The hostname for the Chef server must be a FQDN, including the domain suffix, and must be resolvable. See Hostnames, FQDNs for more information

For a standalone deployment…

- 4 total cores (physical or virtual)
- 8 GB of RAM or more
- 5 GB of free disk space in /opt
- 5 GB of free disk space in /var

The RAM requirement can be lowered down to a minimum of 4 GB of RAM if the number of Chef client runs (CCRs) per minute are low (i.e. less than 33 CCRs/min)