**Organizations**

* Completely independent tenants of Enterprise Chef
* Provide multi-tenancy in Enterprise Chef
* Share nothing with other organizations
* May represent different
  + Companies
  + Business Units
  + Departments
* Each Organization may have multiple Users
* Manage an Organization’s Users via the Enterprise Server interface

**Environments**



Environments reflect your patterns and workflow, and can be used to model the life-stages of your application

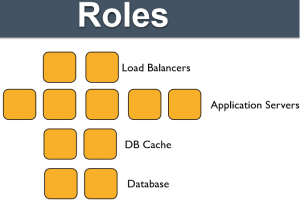
* Development
* Test
* Staging
* Production
* etc.

Every Organization starts with a single environment!

Environments may include data attributes necessary for configuring your infrastructure, e.g.

* The URL of your payment service’s API
* The location of your package repository
* The version of the Chef configuration files that should be used

**Roles**

[](https://mychefscribble.files.wordpress.com/2015/07/roles-blog.png)

Roles may include an ordered list of Chef configuration files that should be applied

* This list is called a Run List
* Order is always important in the Run List

Roles may include data attributes necessary for configuring your infrastructure, for example:

* The port that the application server listens on
* A list of applications that should be deployed

**Node**

Nodes represent the servers in your infrastructure

* Could be physical servers or virtual servers
* May represent hardware that you own or compute instances in a public or private cloud
* Could also be network hardware – switches, routers, etc

Each Node will

* Belong to one Organization
* Belong to one Environment
* Have zero or more Roles

The chef-client application runs on each node, which

* Gathers the current system configuration of the node
* Downloads the desired system configuration policies from the Chef server for that node
* Configures the node such that it adheres to those policies

**Resources**

A Resource represents a piece of the system and its desired state

* A package that should be installed
* A service that should be running
* A file that should be generated
* A cron job that should be configured
* A user that should be managed
* and more

Resources are the fundamental building blocks of Chef configuration

Resources are gathered into Recipes

Recipes ensure the system is in the desired state

**Recipes**

* Configuration files that describe resources and their desired state
* Recipes can:
  + Install and configure software components
  + Manage files
  + Deploy applications
  + Execute other recipes
  + and more

**Cookbooks**

* Recipes are stored in Cookbooks
* Cookbooks contain recipes, templates, files, custom resources, etc
* Code re-use and modularity
* A cookbook is like a “package” for Chef recipes.
  + It contains all the recipes, files, templates, libraries, etc. required to configure a portion of your infrastructure
* Typically they map 1:1 to a piece of software or functionality.

**Run List**

* The Run List is an ordered collection of policies that the Node should follow
* Chef-client obtains the Run List from the Chef Server
* Chef-client ensures the Node complies with the policy in the Run List

**Knife**

Knife provides an API interface between a local Chef repository and the Chef Server, and lets you manage:

* Nodes
* Cookbooks and recipes
* Roles
* Stores of JSON data (data bags), including encrypted data
* Environments
* Cloud resources, including provisioning
* The installation of Chef on management workstations
* Searching of indexed data on the Chef Server

**Node Objects**

* When you are writing Recipes, the Node object is always available to you.
* Every node must have a unique name within an organization
* Chef defaults to the Fully Qualified Domain Name of the server, i.e. in the format server.domain.com
* Nodes are made up of Attributes •
  + Many are discovered automatically (platform, ip address, number of CPUs)
  + Many other objects in Chef can also add Node attributes (Cookbooks, Roles and Environments, Recipes, Attribute Files)
  + Nodes are stored and indexed on the Chef Server

**What are Attributes?**

* Attributes represent information about your node
* The information can be autodetected from the node (e.g.# of CPUs, amount of RAM) & populated by Ohai
* You can also set attributes on your node using cookbook recipes & attribute files, roles, environments, etc
* Attributes keep the program code separate from data.
* All attributes are set on the “node object”, and are indexed for search on the server

**Attribute Sources** Attributes can be set at various levels (in increasing order of precedence)

* Automagically on the node itself (by Ohai)
* In roles
* In environments
* In cookbook recipes
* In cookbook attribute files

Attributes can be set in the cookbook’s attributes file **./cookbooks/<cookbook>/attributes/default.rb**. The Format is [attribute-blog-1](https://mychefscribble.files.wordpress.com/2015/07/attribute-blog-1.png) In Recipes: [](https://mychefscribble.files.wordpress.com/2015/07/attribute-blog-2.png)

**Data Bags**

A data bag is a container for items that represent information about your infrastructure that is not tied to a single node

Examples

* Users
* Groups
* Application Release Information

**chef cheat sheet**

|  |  |  |
| --- | --- | --- |
| **Keyword/Operation** | **Description/Command** | **Comments** |
| cookbook | A cookbook is the fundamental unit of configuration and policy distribution.Each cookbook defines a scenario, such as everything needed to install and configure MySQL. |  |
| role | A role is a way to define certain patterns and processes that exist across nodes in an organization as belonging to a single job function.Each role consists of zero (or more) attributes and a run-list. |  |
| data bag | A data bag is a global variable that is stored as JSON data and is accessible from a Chef server.A data bag is indexed for searching and can be loaded by a recipe or accessed during a search |  |
| Upload cookbook | knife cookbook upload <cookbook-name> eg:knife cookbook upload cookbook-httpd |  |
| Add/Upload role | knife role from file <FILE> eg: knife role from file http-env.rbknife role from file ./roles/\*.rb |  |
| View or Show role | knife role show <role name> eg:knife role show http-env |  |
| Update role | knife role edit <role name> eg:knife role edit http-env |  |
| Add data bag | knife data bag from file <FILE> eg:knife data bag from file http-password.json |  |
| Update data bag | knife data bag edit <databag name> eg:knife data bag edit http-password |  |
| View/Show data bag | knife data bag show <databag name> eg:knife data bag show http-password |  |

**CONCEPTS  
OPEN VS PAID**  
Chef has two different flavors that it comes in, paid or free. Paid is called Chef Enterprise and comes with some extra features and support (ex: hooks into M$ Active Directory, managing your enterprise like tenants/projects like in OpenStack, . Chef Open Source is the free version that includes only up to 25 nodes of the [Enterprise options](https://www.chef.io/pricing/). You also in turn could host Chef in the cloud (for a fee or up to 5 nodes for free) or install it inhouse. One nice thing for the Windows guys out there is you can run Chef with Windows, Macs and Linux machines.

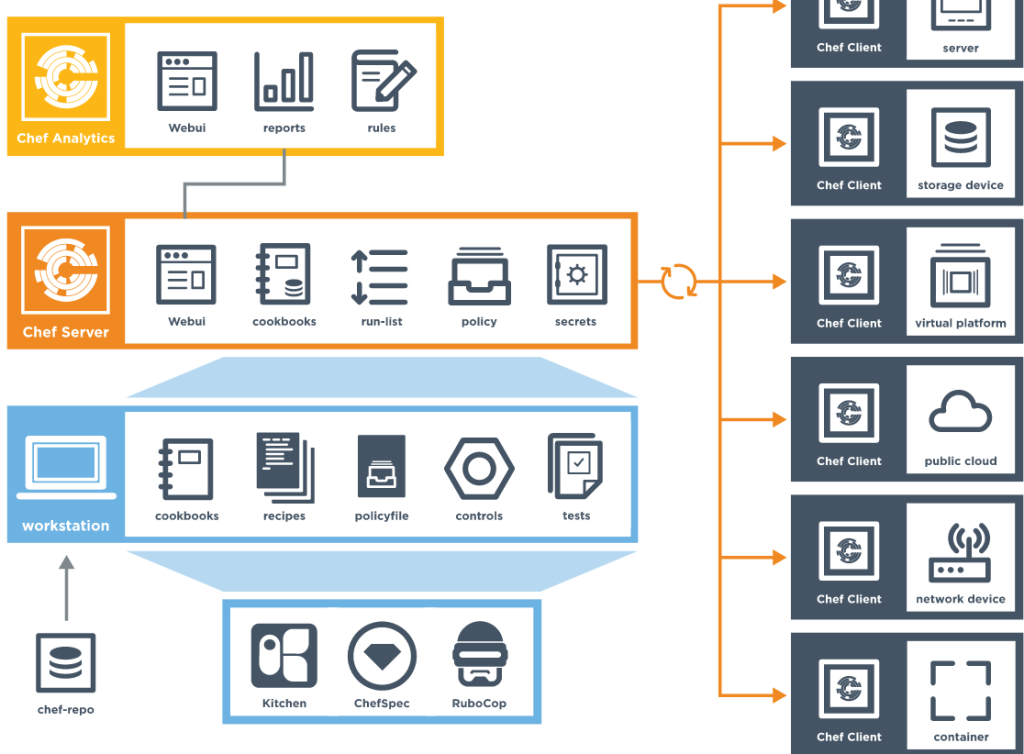
**HISTORY LESSON**  
Chef came out in Jan 2009, whereas Puppet is older started in 2005. Similar to Chef, Puppet is also written in Ruby. Ansible is the newer of the 3, having started in Feb 2012. Ansible uses Python to run its playbooks and uses YAML. Now do you need to know Ruby to use Chef? ***No, you don’t need to know Ruby***. But if you want to be a full-blown Puppet Engineer, it would probably be a good idea to do so.

**HOW DOES CHEF WORK?**Chef Server – the brains of your env. Cookbooks pulled by clients here. This machine HAS to be 64-bit, clients don’t need to be.

Chef Client – as the name implies, a device (router,switch, container, vm, etc) that pulls down a Cookbook to receive it’s automated instructions (usually via a cron job or some kind of scheduled repeated task) .  
Chef Workstation – where you update Cookbooks and send them over to the Chef Server. You could manage the server via a GUI, or CLI (called Knife). Note you’ll also need to install the Client tools on this workstation as well. The Chef Repo lives here that will in turn get Cookbooks pulled to Clients.  
Cookbook and Recipes – a Recipe — or a a single task(s) can make up a Cookbook which contain multiples Recipes.

Your Workstation will communicate with the Server securely via chef-client and chef-validator keys.  These are the SSH keys that we have to generate in our routers and switches. You will want to go to each Client and install these keys so that the Server<->Client relationship won’t give you issues. More details on installing can be found [here](https://docs.chef.io/chef_private_keys.html).

We will also want to install Knife on our workstation. For this, type in knife configure on the terminal. For details on this process go [here](https://docs.chef.io/install_dk.html).

[](http://www.nycnetworkers.com/wp-content/uploads/2015/11/chefoverview.png)

**FIRST TIME INSTALLATION OF CLIENT**  
The first time you add a node to Chef, you’ll need to install on the node the Chef client and then add this node to the Chef Server. This process is called “Bootstrapping”. On the workstation machine we can use **knife bootstrap 192.168.1.50 -x wzambrano -P ilovedogs123 –sudo** for example. Find more syntax examples [here](https://docs.chef.io/install_bootstrap.html)and [here](https://docs.chef.io/knife_bootstrap.html). Of course if you have DNS configured you can use the hostname instead of the IP in the knife syntax above.

To verify you the client has been added to the server, type in**knife client list 192.168.1.50** to see its stats. You can also see ALL clients on the server by typing **knife client list**. You can also go to the Chef Server GUI and see it under Nodes.

On the client you can see Chef settings by typing **sudo chef-client**. By default the Cookbook list is empty.

**COOKBOOK CONFIG**  
Now that we’ve added in a few server nodes, lets see what we can do to make a Cookbook and recipes to it. There is a directory called cookbooks and this is where they will live. Once inside of it, type in **knife cookbook create TEST\_COOKBOOK** for example. Chef Docs has a good guide on this found [here](https://docs.chef.io/cookbook_repo.html). Once you do that, a bunch of new directories get created like attribute, resource,files, recipes, etc. View the Chef Docs page to see what each one does.

There are also community created cookbooks found on Chef’s forum dubbed[the Supermarket](https://supermarket.chef.io/cookbooks). “Chef Corp Approved” cookbooks[live here](https://github.com/chef-cookbooks).