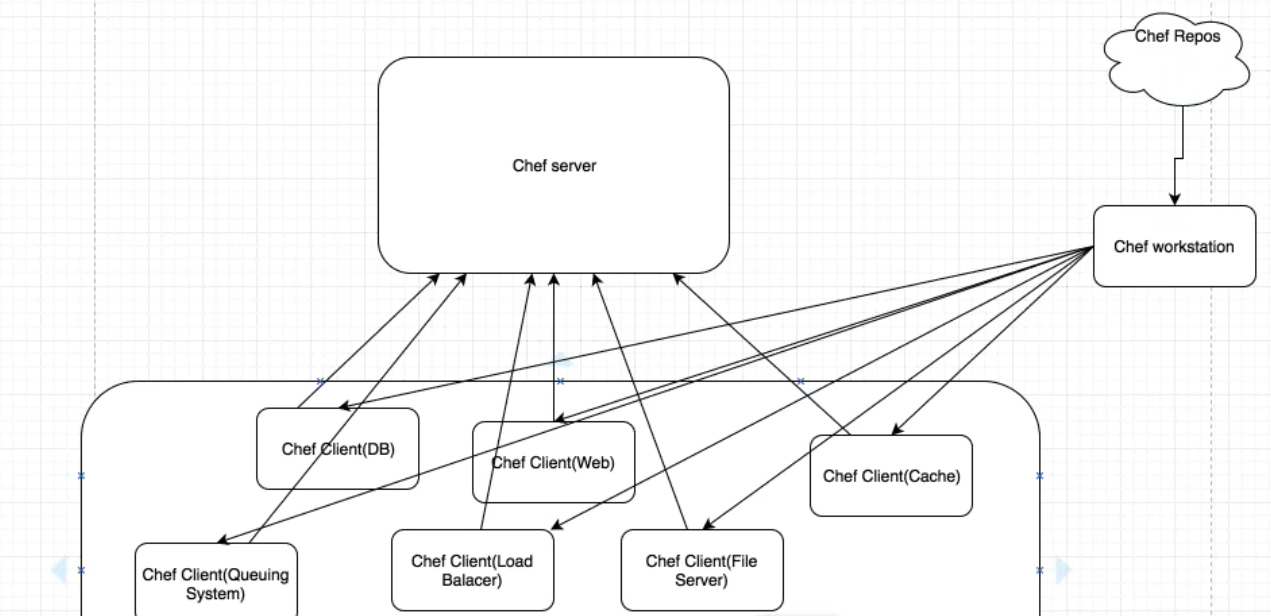
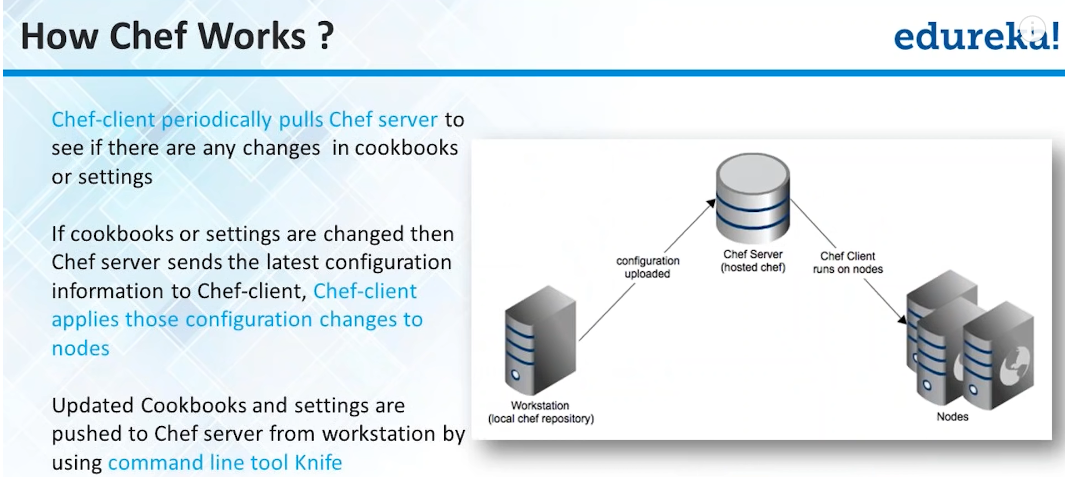
Chef

# Chef Architecture





#### Chef Workstation

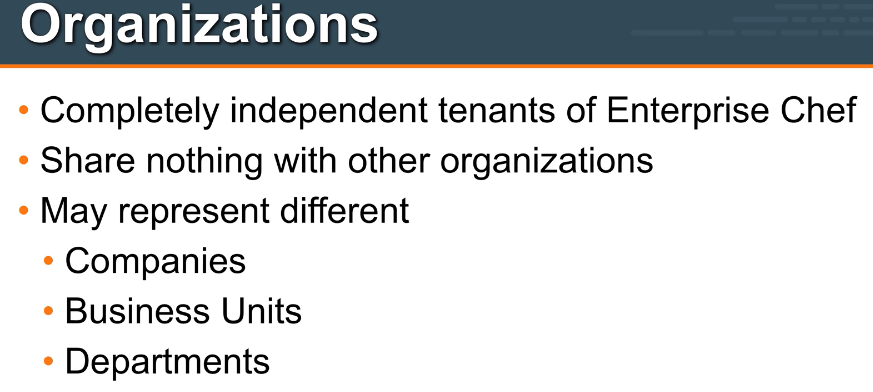
This is responsible to get the client registered with chef server. So basically workstation does the initial talking with clients. Workstation talks to chef repos (which are hosted on the internet) and get the chef client downloaded on all the machines and do the registration with chef server.

#### Chef advantage over puppet

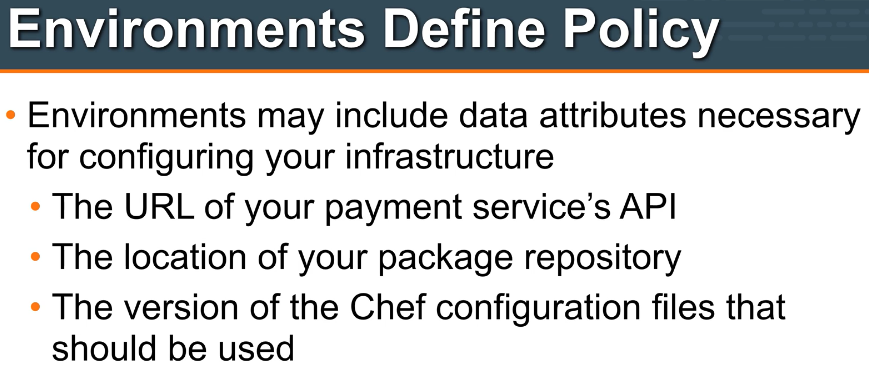
In puppet, we have to do the registration manually on all the clients.

# Chef components

Organization – This represent a separate group like different company or different domain or departments etc. Organisation is to make sure that no organisation infrastructure is visible to other.



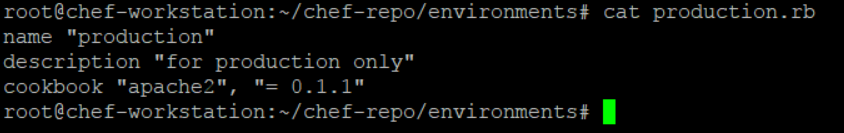
Environment – This is second big entity which describes the different env like dev, prepro, prod. Environment may include data attributes which are required to configure infrastructure.



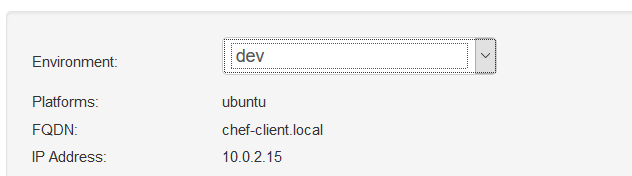
Whenever environments come, think about dev, preprod, prod etc.Environment gives us the ability to use the different versions of the same cookbook across different env like dev, preprod and prod.

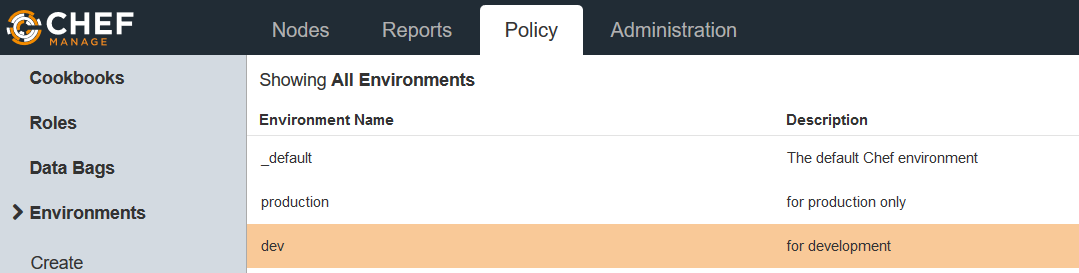
Environment is directly created under chef-repo (as it has to work on all the cookbooks)

Sample environment looks like below

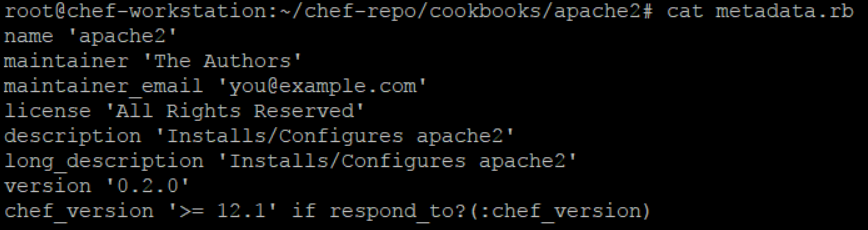


At admin console it looks like below





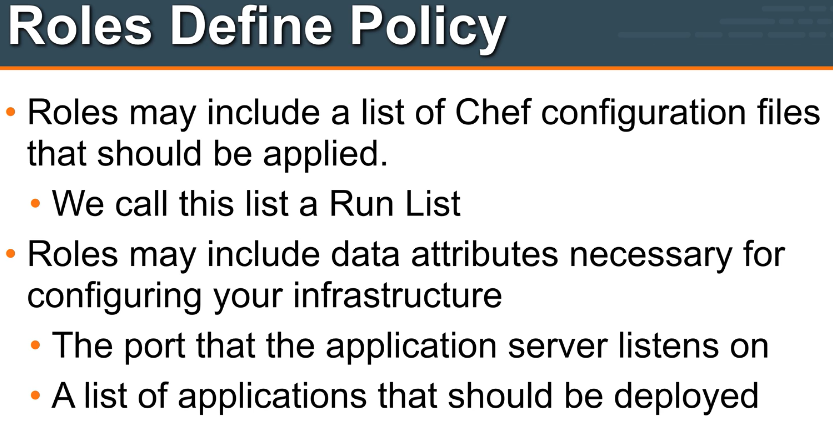
**Important – Environment take the version of a cookbook from its metadata file. Sample below**



For example, the below apache cookbook version is changed to 0.2.0 and when it is uploaded to server, the environment will consider as one piece of code under 0.2.0 version of this cookbook. If we specify this version inside the environment .rb file then chef will load this particular version for that cookbook.

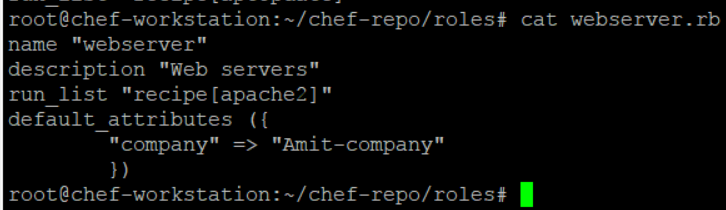
Check environment commands in the chef commands section.

Roles – Roles defines the type of servers in the infra. Like DB server, cache server, web, application servers etc. Roles may include the chef config files that should be applied. This is called Run List



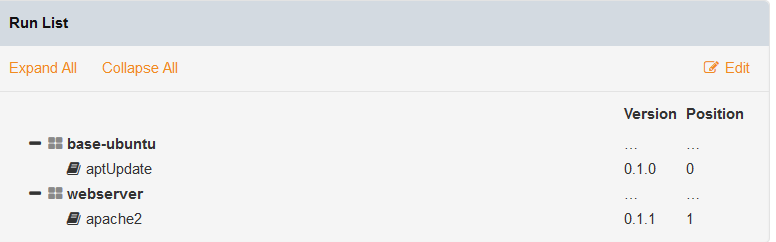
# Important points about Roles

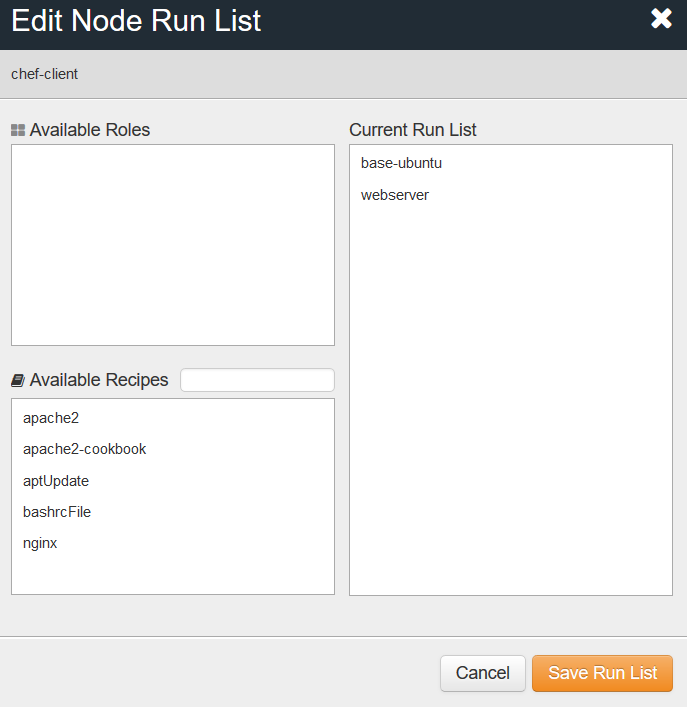
Whenever think about role, think about the different types of nodes like some are web server, some are application server, db servers etc. Roles are not related to some cookbook but for whole environment as they are used by nodes and contains cookbooks. This is the reason roles are directly under chef-repo. Sample role look like below.



Roles basically contains the cookbooks (one on many) and hence are used in runlist.(rather than pushing separate cookbooks in runlist, we push the roles)

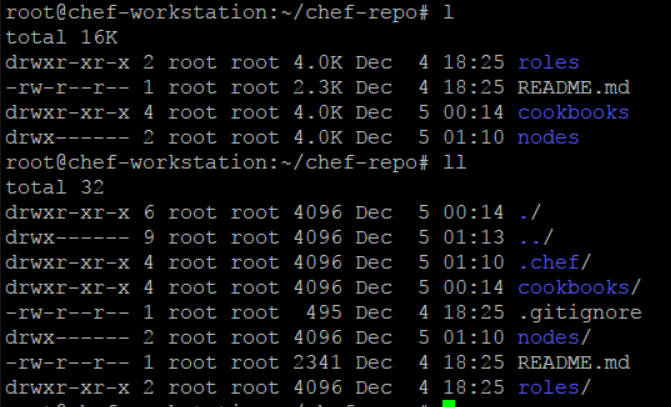
Role runlist at admin console look like this



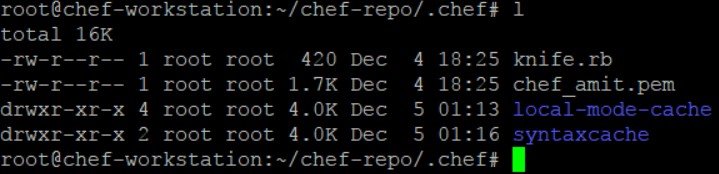


# Understanding Starter Kit, chef-repo, berksfile, metadata.rb and knife command line utility

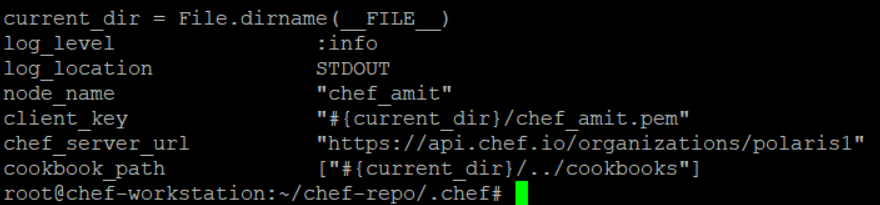
Once the organisation is created on chef server (personal or hosted), we download a starter kit (chef-starter.zip)and unzip it on the chef-workstation. This create the following **chef-repo** folder and contents.



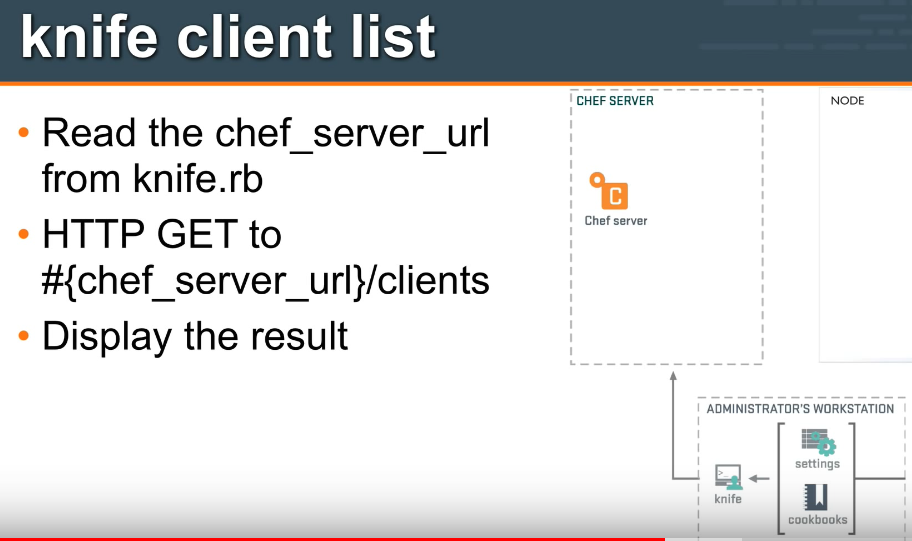
**Knife** is a command line utility which comes under .chef folder. This knife.rb file contains the knife configuration like chef-server url, client keys to do the communication between workstation and chef server.



Knife.rb look like below



**Lets check how knife works with the help of command knife client list**



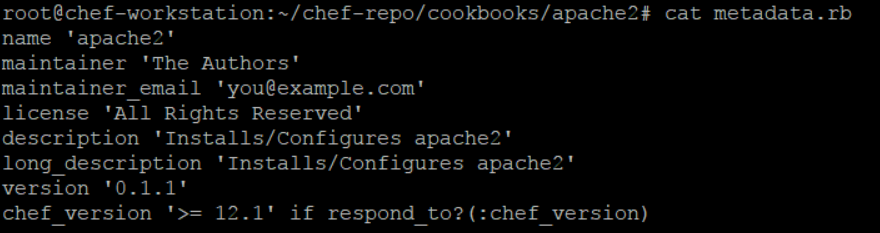
**Ohai** – Ohai is a system profiler tool which gathers the system information about the node and makes it available to the server. This is done at the time of bootstap.( knife node show chef-client –l or ohai is the command)

**Berksfile** is for dependency management for cookbooks. Consider a case where my cookbook is using a community cookbook from chef supermarket. In this case, first I need to download that community cookbook from supermarket and upload it along with my own cookbook to chef server. Berksfile simplifies this workflow for you. With single command (berks install), it downloads all dependent cookbooks (and their dependent cookbooks -- transitive dependencies) from their respective sources (may be from git repository or from supermarket). With another single command berks upload it uploads all these cookbooks to chef server. You do not have to upload them individually with knife cookbook upload. Role of Berksfile in particular cookbooks life-cycle finishes here.

**metadata.rb** is referred by chef-client while actually converging the node. It uses this file to download all the required cookbooks from chef server (assuming that these cookbooks are now available on chef server by using berkshelf or knife) to the node to successfully complete the chef-client run. Sample metadata file look like below.

Here metadeta can be used for versioning of cookbook as well as chef load only that version of cookbook whatever is mention under version attribute of the metadata.

This cookbook versioning is very important in case of uploading different versions of cookbook on different environments like dev, preprod, prod.



## Data Bags

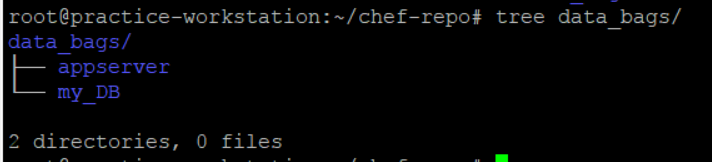
Chef data bags the file place holders which are used to store the sensitive information like password.

They are created with below command.

Knife data bag create bag name (this will create the data bag on server not on workstation)

This can be downloaded from server to workstation by knife download data\_bags

Or other way is to simply create data bags under chef-repo folder like below.



# Wrapper cookbook

If we want to use an ntp cookbook to set a server's time, we can download the community ntp cookbook from Opscode. After downloading, instead of running it directly, we can modify it according to our infrastructure server's time setting.Following is a representation of wrapper cookbooks for more understanding of the concept: We can create another myinfra-ntp cookbook with the following settings and change the attribute settings in the following manner:

myinfra-ntp/attributes/default.rb

default['ntp']['peers'] = ['ntp1.myinfra.com', 'ntp2.myinfra.com']

After this, we can add attributes to the recipes:

myinfra/recipes/default.rb

include\_recipe 'ntp'

Now, we can simply run recipe[myinfra-ntp] in our running infrastructure, and the default settings from cookbook will automatically come up.

#### Chef Installation

Once machine is build change the host name in vim /etc/hostname to chef-server

Set the fully qualified domain name (only require for home lab)

Go to vim /etc/hosts

And make the below entry at the bottom of file (here ip address is the ip of same server)

192.168.1.22 chef-server.local chef-server

Download the installation file from the below location

*Wget* [*https://packages.chef.io/files/stable/chef-server/12.17.5/ubuntu/16.04/chef-server-core\_12.17.5-1\_amd64.deb*](https://packages.chef.io/files/stable/chef-server/12.17.5/ubuntu/16.04/chef-server-core_12.17.5-1_amd64.deb)

*To download the chef development kit for workstation (check the OS and version you are using)*

*Wget* <https://packages.chef.io/files/stable/chefdk/2.4.17/ubuntu/16.04/chefdk_2.4.17-1_amd64.deb>

*Dpkg –i chefdk\_2.4.17-1\_amd64.deb*

#### Working with Workstation

**Create a recipe hello.rb like below**

file 'motd' do

content 'Hello world!'

end

and then apply it with chef-apply hello.rb now this will create a file name motd.

If you change the content of motd and again run chef-apply hello.rb. it will maintain the state and change the file content to again Hello world! As specified in recipe.

**Now delete this file with**

file 'motd' do

action :delete

end

Now install apache on workstation like below and apply and then run the local host to test



To check the port number

netstat –tulpno

## Sample Cookbook creation

Go to chef directory and fun command - chef generate cookbook apache2-cookbook



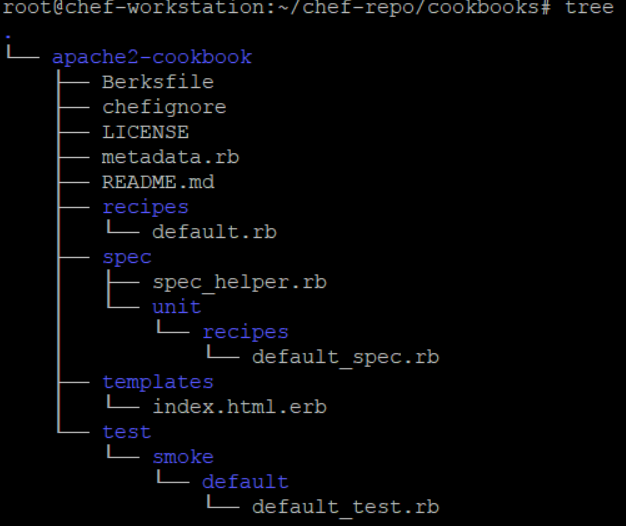
Note – default.rb in side recipe always executed first in cook book.

Generating template – writing big code like template inside recipe.rb file can be a mess incase of big configurations.

So it’s good to have the template separated from recipe code. Run below command to create a template in a cookbook

root@chef-workstation:~/chef-repo/cookbooks# chef generate template apache2-cookbook/ index.html

It includes the template folder like below.

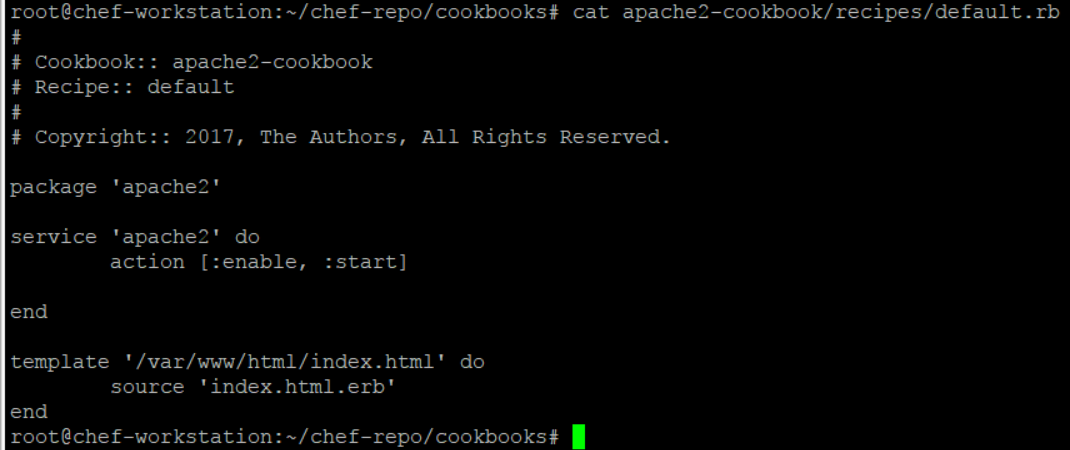


Now change the template like below

vim apache2-cookbook/templates/index.html.erb

Hello World

Change the default recipe like below



Now run the cookbook like below

root@chef-workstation:~/chef-repo/cookbooks# chef-client --local-mode --runlist 'recipe[apache2-cookbook]'

**Now upload the cookbook to the chef server (internet not standalone)**

Login to manage chef website and create a organisation and download a chef-started key

Now unzip this key chef-starter.zip on workstation. This will create the communication between server, workstation and nodes.

Now upload the cookbook on chef internet server

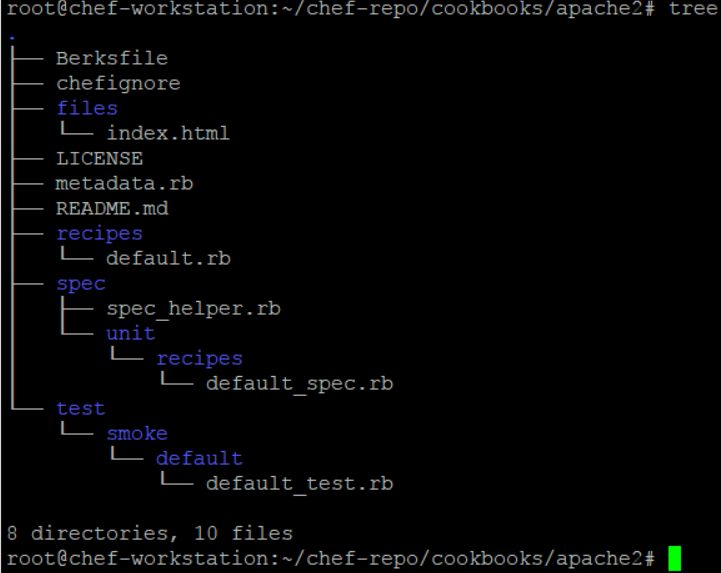
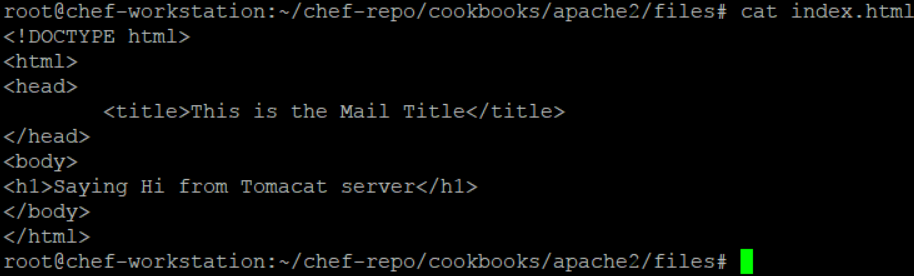
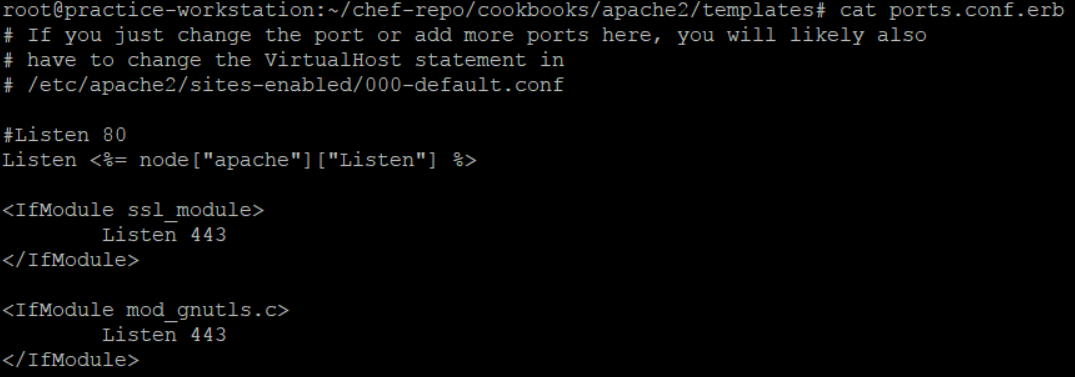
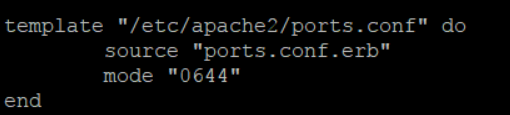
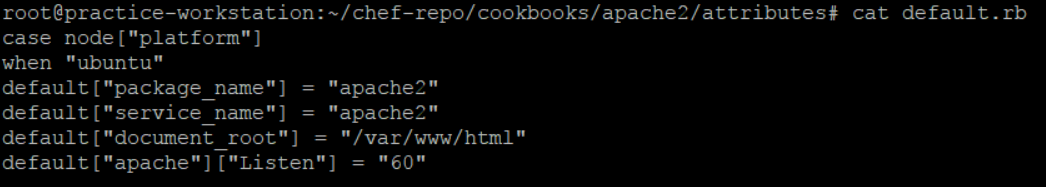
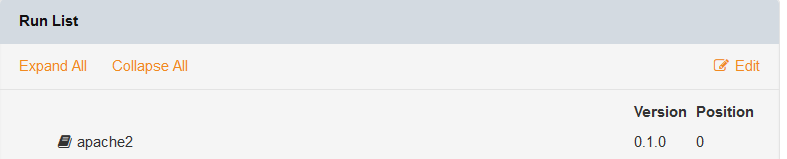
root@chef-workstation:~/chef-repo# knife cookbook upload apache2-cookbook

**Now bootstrap the node and run the below command on workstation.**

This will install chef client on nodes and pull the cookbook from chef internet server.

***knife bootstrap 192.168.169.6 --ssh-user root --ssh-password 'redhat' --sudo --use-sudo-password --node-name chef-client --run-list 'recipe[apache2-cookbook]'***

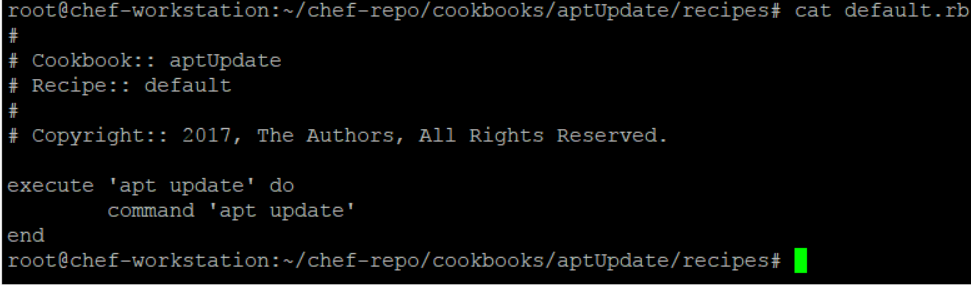
# Sample Cookbook 1 – apache2

1. On the workstation inside the /chef-repo/cookbooks, generate the new cookbook with command *chef generate cookbook apache2*
2. It will create a new cookbook apache2 like below
3. 
4. *Now edit the default recipe inside the recipe folder*
5. 
6. *Create a* ***files*** *folder inside the apache2 cookbook and create a index.html file like below*
7. 
8. *If we want to change the port of apache from 80 to some other then create a separate template like below*
9. 
10. *Make change in recipe like below.*
11. 
12. *As this is conf file change, make sure to restart the apache service in recipe rather than just start.*
13. *Also add port attribute like below.*
14. 
15. *Now upload the cookbook to the chef server like* ***knife cookbook upload apache2***
16. *Before we Pull and apply this cookbook on the node, we need to create a run list on node like below.*
17. *There are 2 ways of creating the run list i.e through server admin console and through command from workstation.*
18.  *or*
19. *knife node run\_list add chef-client recipe[apache2]*
20. *once run list is created, run the* ***chef-client*** *command from the node machine.*

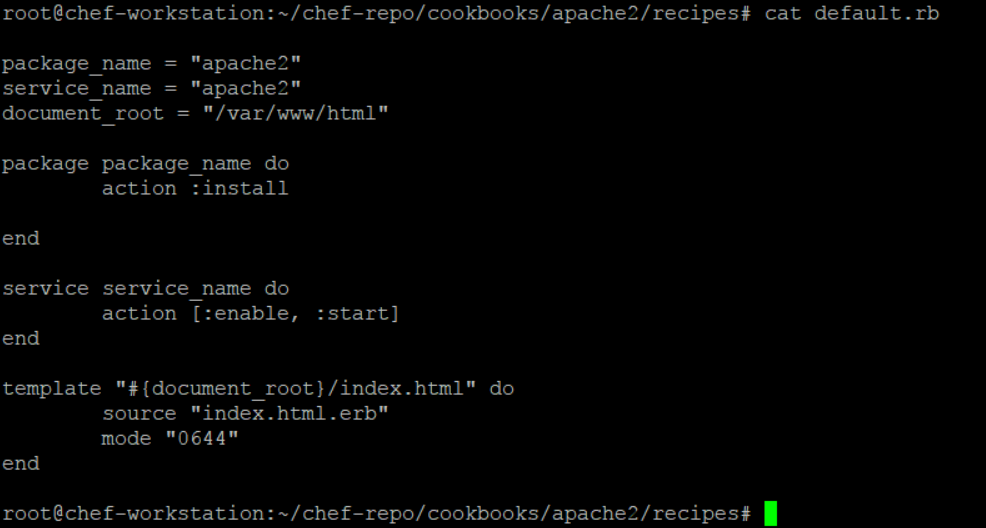
# Sample cookbook2 – nginx

1. Completed with the same approach below is the recipe file.
2. 

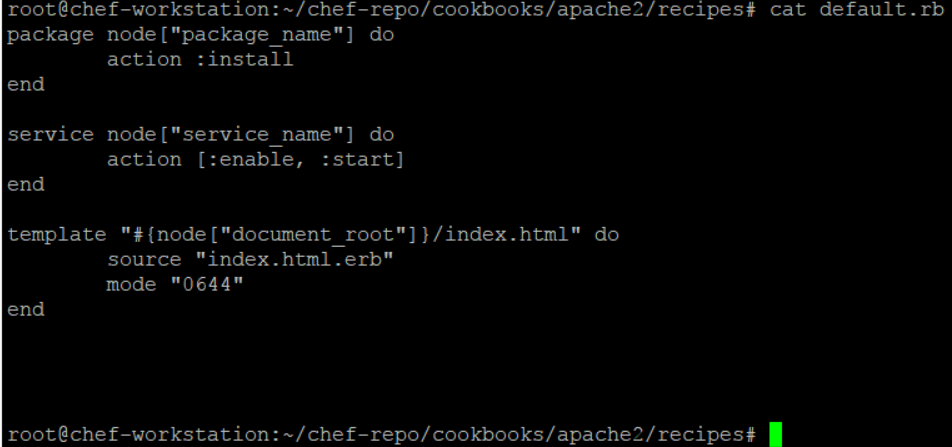
# Sample cookbook3 – apt update

1. Completed with same approach with the below recipe file
2. 

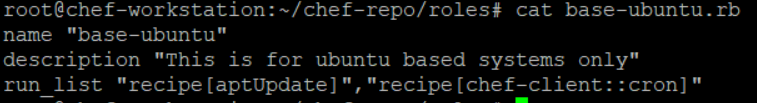
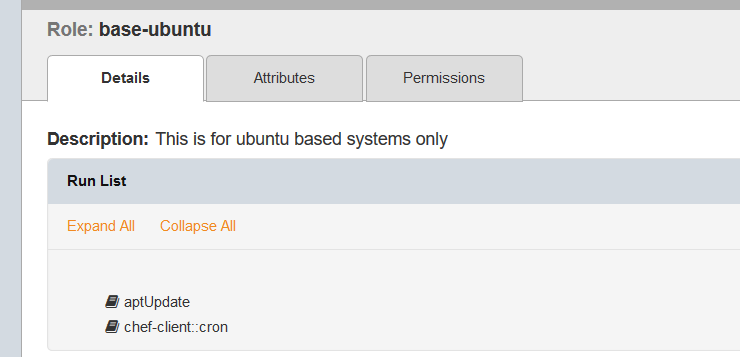
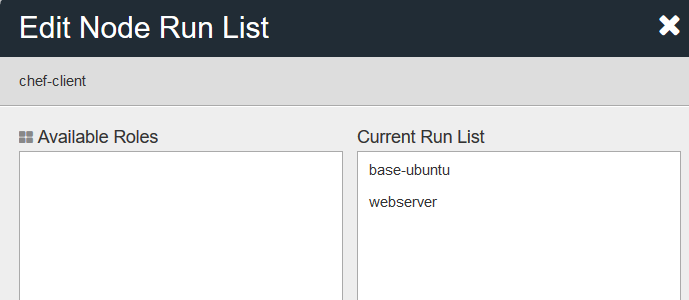
# Sample Cookbook4 – apache4 with template

1. Completed with same approach like apache2.
2. Changed only 2 files like below
3. 
4. 

# Sample Cookbook5 – Apache with attributes

1. Completed with the same approach as apache2 with the below changes
2. 
3. 

# Sample Cookbook6 – Download chef-client (for cron job setting) from supermarket

1. Download the chef-client and its depending cookbooks with below 2 command (chose any one)
2. knife cookbook site install chef-client (Use this – this is download and install the chef-client and all its depending cookbooks latest versions)
3. knife cookbook site download chef-client (here you will have to manually download and unzip the chef-client and its depending cookbooks in cookbook/ folder.
4. The above commands will connect to supermarket (<https://supermarket.chef.io/>) and download it.
5. Upload all the cookbooks to the chef servers
6. Now update the role like below. Here we have only made the recipe cron of cookbook chef-client, the part of role run\_list.
7. 
8. Now push this role to the runlist of the node like below.
9. 
10. 
11. Now from the node run chef-client and check the cronjob with crontab -l

#### Difference between Ansible puppet and chef

**Chef** – This is for large infra( 1000+ servers), follows the server agent architect and fast because it make connection and run the command only when the state of the nodes is change, so less overhead.(every 30 min checking of state)

**Ansible** - for small infra (less than 1000 servers), agentless and it’s slow because it always go and create the ssh connections to all the agents and execute command even if there is no change in the state of the nodes. (Every 30 min checking of state). In push mode, Ansible requires a solid SSH connection - even if there is no change in the module, it still go to the nodes and run the commands. SSH communication slows down in scaled environments

# Life cycle of a cookbook

**Generate cookbook in chef-repo/cookbook**

**Write recipe**

**Upload cookbook to server**

**Update run list with this cookbook**

**Run chef-client from node**

#### Chef-Commands

1. To bootstrap node from workstation - knife bootstrap 192.168.169.6 --ssh-user root --ssh-password 'redhat' --sudo --use-sudo-password --node-name chef-client --run-list 'recipe[apache2-cookbook]'
2. Command to generate cookbook – chef generate cookbook cook\_book\_name
3. To apply a recipe – chef-apply recipe\_name
4. To generate cookbook - chef generate cookbook cookbook\_name
5. To check the syntax of cookbook – knife cookbook test cookbook\_name
6. To upload cookbook - knife cookbook upload cookbook\_name
7. To show all the information of cookbook - knife cookbook show cookbook\_name
8. To list down all the nodes - knife node list
9. To check particular node information - knife node show node\_name
10. To check the system profile of node or all the node configuration - knife node show chef-client –l or ohai
11. To check the system information like memory - knife node show chef-client -a memory.free( here with . you can access all the subattributes as well)
12. To add runlist through command - *knife node run\_list add chef-client recipe[cookbook\_name]*
13. To remove runlist through command - *knife node run\_list add chef-client recipe[cookbook\_name]*
14. *To add role in runlist - knife node run\_list add chef-client 'role[role\_name]'*
15. *To remove role from runlist - knife node run\_list remove chef-client 'role[role\_name]'*
16. *To upload roles to server - knife role from file role\_name*
17. *To show all the roles - knife role list*
18. *To show more info about particular role - knife role show role\_name*
19. To upload environment - knife environment from file env\_name.rb (ex. Production.rb)
20. To list all environments - knife environment list
21. To show some particular environment - knife environment show env\_name ( like dev)
22. knife cookbook site install chef-client (**Use this** – this will download and install the chef-client and all its depending cookbooks latest versions)
23. knife cookbook site download chef-client (here you will have to manually download and unzip the chef-client and its depending cookbooks in cookbook/ folder.
24. to know the latest version of any cookbook on supermarket - knife supermarket show httpd | grep latest

# Downloading cookbooks from chef supermarket

Check the below sample module for full info on this.

**Sample Cookbook6 – Download chef-client (for cron job setting) from supermarket**

<https://supermarket.chef.io/>

# Useful cookbook to have

