
Setup and Configure Chef Server on Ubuntu

14.04

The central Chef server is one of the core component of chef infrastructure automation. Central Chef server acts as a single point of contact for agents to pull configurations that are applicable for them. Chef server stores cookbooks, metadata of the nodes, environment and policy details etc. Every agent in the environment will be configured to contact a central chef server.

Read : <http://www.slashroot.in/chef-tutorial-beginners-getting-started-introduction>

Before getting started, let's understand the different components of Chef server. Chef server will automatically install the below items on the server.

- **Nginx:** Nginx is an open source webserver that will be used by chef as a front end web interface. Each and every HTTP request that comes to chef server, will actually be going via Nginx web server. Once the request hits Nginx web server, Nginx will then re-route those requests according to the type of request. For example...If the request is to access bookshelf, it will be forwarded to localhost port number 4321(where bookshelf service will be listening.). If it's search related request, then Nginx will forward it to Solr listening on localhost port number 8983(all of these different services are installed by chef server package automatically, without any user intervention.)

Bookshelf is a component of chef server where all cookbook related contents are stored. If you have ever worked with Puppet configuration management tool, then you can compare cookbook with modules of puppet.

- **Chef Web Interface:** This is a Ruby application that listens on localhost port number 9462. Using this interface, administrators can configure and manage different aspects of chef server. Like which cookbooks to be applied on which node etc. This component is also something that gets installed by chef automatically without any manual intervention(we will be doing the installation part shortly). The requests to access the chef user interface will also first reach Nginx web server, which will then redirect it to port number 9462 where the ui application is running.

Please keep the fact in mind that chef web interface is only free for the first 25 nodes. You will have to pay for using this web interface for more number of nodes. However..You can have chef working without the web interface. The operations and features exposed by this web interface is actually doable using command line as well.

- **Chef API:** This is the core API of Chef. The opscode engineering team reworked the API using Erlang programming language. Hence this is called as "Erchef". This API service runs on port number 8000. Here also Nginx redirects requests to localhost port number 8000.
- **Solr:** This component provides indexing and searching feature for Chef. All node details and attributes are stored in the open source search engine called Apache Solr. Solr service exposes itself in the form of a search api to chef using port number 8983. Nginx forwards requests for searches to this localhost port number 8983.
- **PostgreSQL Database Server:** This is a popular open source relational database server. Chef uses this as the primary database for storing things. PostgreSQL server listens on port number 5432.
- **RabbitMQ:** This is used as a message queuing component of chef. Things are grabbed from RabbitMQ for indexing to solr search engine.

As mentioned earlier...None of the above mentioned components need to be configured manually. All these components are configured by chef installation script automatically for you.

Step 1: The very first step is to verify that correct hostname is configured on the server. And the hostname FQDN is actually resolvable. Its very much necessary and recommended to have proper DNS entries in place for your chef server. If you are not in a production environment, you can get it done using hosts file as well. Login to the server, and run the below commands to set correct hostname..

```
edit /etc/hosts      # 127.0.1.1    ubuntu-server.com
/etc/hostname        # ubuntu-server.com
```

Step 2: The second step is to ensure time synchronization. So we need to configure NTP on the server. This can be done by installing ntp package in Ubuntu as shown below..

```
# apt-get update
```

```
# apt-get install ntp
```

```
# service ntp start
```

Step 3: The next step is to download Chef server package from official website

```
# wget https://packages.chef.io/stable/ubuntu/14.04/chef-server-core_12.7.0-1_amd64.deb
```

Step 5: The next step is to install this downloaded package using the below command.

```
# dpkg -i chef-server*
```

Step 6: The next step is to ask chef server to configure itself with the default settings. This can be done by running the below command.

```
# chef-server-ctl reconfigure
```

This command will take a while to complete. This is because it installs and configures all the previously mentioned components of chef server(ie: Nginx, RabbitMQ, Postgresql, Solr etc.). The successful completion of the above command should show you something like the below towards the end..

it show like following

```
Chef Client finished, 39/411 resources updated in 31 seconds  
Chef Server Reconfigured!
```

Step 7: Once the above command is completed successfully, we have chef server components installed and configured on the server. You can actually have an optional web interface for chef as well. This web interface is called as chef manage. It can be installed using the below command.

<https://downloads.chef.io/chef-manage/ubuntu/>

```
# dpkg -i chef-manage*
```

```
# chef-server-ctl reconfigure
```

```
# chef-manage-ctl reconfigure
```

Once the above three commands succeeds, you have the web interface called chef manage ready for you to use. You can access the web interface by either using the IP address of your server or the DNS name of the server(if you have DNS configured correctly in your environment.)

Before we can access the web interface, we need to create an admin user and credentials..This can be done by executing the below command.

```
# chef-server-ctl user-create akhil akhil raj akhil.raj@bizruntime.com secret --filename akhil.pem
```

here ::

"akhil" in the above command is the username.

"akhil raj" is the full name of the user.

"akhil.raj@bizruntime.com" is the email address.

"secret" is the actual password for the user..Replace it with something complex

"akhil.pem" is the private key file for the user(this will be created in the current directory after the command completes.)

You can now access the web interface by using the IP address of the server in the web browser. The login page looks like the below..

Sign In - Chef Manage - Mozilla Firefox

https://ubuntu-server.com/login

Sign In

Username or Email Address

Password

[Forgot your password?](#)

Don't have an account?
You're one step away from access to all the power and flexibility of Chef. Get ready to automate your infrastructure, accelerate your time to market, manage scale and complexity, and safeguard your systems.
[Click here to get started!](#)

Copyright © 2012 – 2016 Chef Software, Inc. Need help? If you have questions or you're stuck, we're here to help. What's New?

once provide username and password, it shows like below,

Chef Manage - Mozilla Firefox

https://ubuntu-server.com/

Welcome to CHEF MANAGE

Thank you for using Chef!

You are not yet a member of any organizations, so please either create a new organization or accept a pending invitation.

If you are trying to join a specific organization and don't have any invitations, get someone in the organization to send you an invitation, then hit the refresh button.

[Sign Out](#)

Copyright © 2012 – 2016 Chef Software, Inc. Need help? If you have questions or you're stuck, we're here to help. [Feedback](#) [What's New?](#) [About Chef Manage](#)

links ::

install and configure chef-server >>

www.slashroot.in/how-install-and-configure-chef-server-ubuntu-1404

<http://www.slashroot.in/chef-tutorial-beginners-getting-started-introduction>

install chef-manage (web-interface)>>

<https://downloads.chef.io/chef-manage/ubuntu/>

didnt try (if above one fail try this)>>

<https://hostpresto.com/community/tutorials/install-and-configure-chef-server-on-ubuntu-14.04/>

dockerfile installation(not recommending)>>

<https://github.com/tmc/dockerfiles/tree/master/chef-server>

Setup and Configure Chef Workstation on Ubuntu 14.04

You can consider Chef workstation as a place where all the development work of chef happens. This is the place where most of the administrators will start working on creating cookbooks and recipes. The workstation contains a local chef repository. This repository can then be synchronized with the central chef server.

Chef workstation also will have a command line utility called "Knife", which will be used to interact with the central chef server.

If you are new to chef and its working..I would recommend reading the below article first.

Read: [A beginners guide to Chef Configuration Management](#)

This article is the continuation of the previous article in this tutorial series. You can access the previous article, which walks through the steps for installing and configuring Chef server using the below Link.

Read: [Installing And Configuring Chef Server](#)

Chef workstation consists of two primary components as mentioned earlier.

1. Chef Repository
2. Knife Utility

Let's first install Knife utility on our chef workstation. Knife utility and chef client can be installed easily on the workstation by simply firing up the below curl command. The below curl command downloads a bash shell script provided by chef, and then executes it.

If you need to use kitchen in chef then try <https://downloads.chef.io/chef-dk/> for install, not curl command

Before that tell to the workstation about ip and FQDN of chef-server

```
# nano /etc/hosts
```

```
192.168.1.64  ubuntu-server.com
```

```
# curl -L https://omnitruck.chef.io/install.sh | sudo bash
```

Successful completion of the above command will look like the below..

```
Installing chef
installing with dpkg...
Selecting previously unselected package chef.
(Reading database ... 51193 files and directories currently installed.)
Preparing to unpack .../chef_12.11.18-1_amd64.deb ...
Unpacking chef (12.11.18-1) ...
Setting up chef (12.11.18-1) ...
Thank you for installing Chef!
```

Alternatively you can also do the following steps to install chef workstation specific components.

Step 1: Chef officially provides a debian and rpm package called chefdk. chefdk stands for chef development kit. You can download your operating system specific package by navigating to the below URL.

<https://downloads.chef.io/chef-dk/>

Step 2: Second step is download it to your workstation server by simply copying the URL specified in the download page and then pass it as a parameter to wget command.

For Ubuntu, you will be doing something like the below..

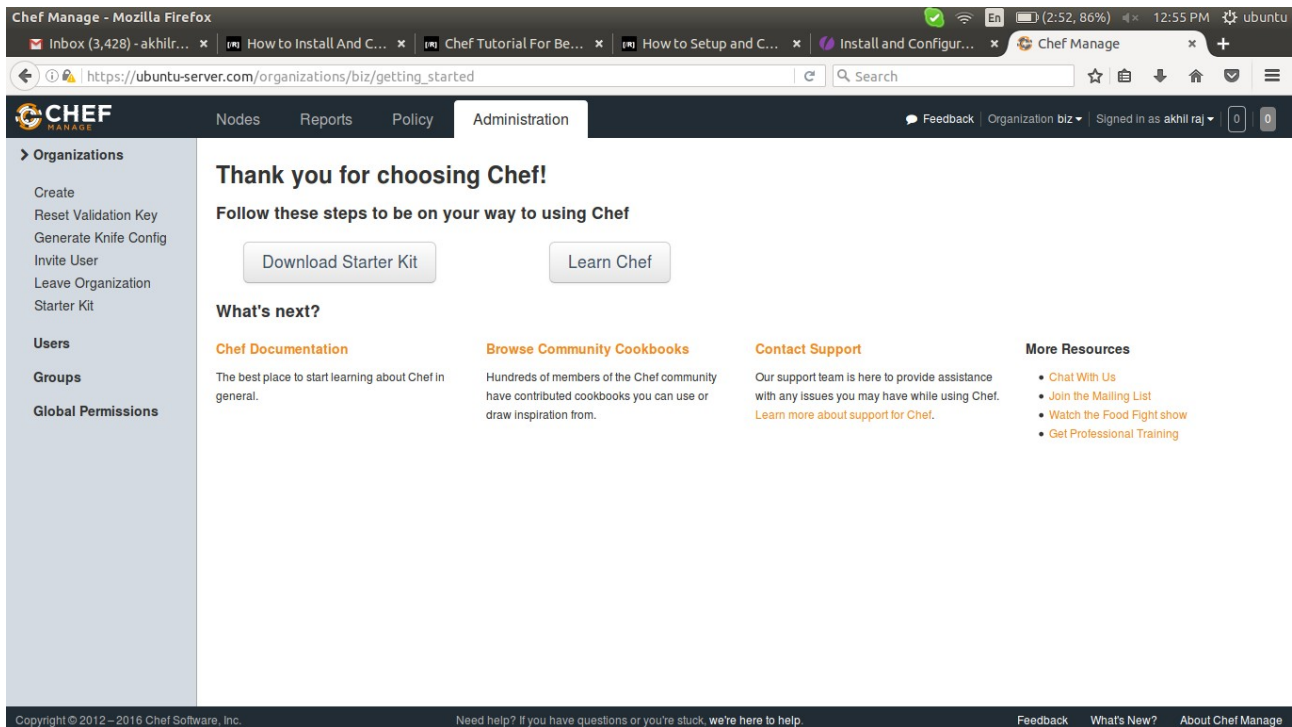
```
# wget https://packages.chef.io/stable/ubuntu/12.04/chefdk_0.15.15-1_amd64.deb
```

```
# dpkg -i chefdk*
```

Chef Starter kit

Chef starter kit is nothing but an archive that contains the directory structure for doing development work on chef workstation. It contains the cookbook and knife settings.

You can simply download it and copy the archive to your chef workstation server.



When you click on the download button above, you will be shown a message with regard to resetting of the user keys. Proceed with it. This new key will be part of the starter kit that we are downloading now.

Once downloaded(chef-starter.zip), copy it down to the chef workstation server. Once copied, you can then decompress it as shown below.

unzip chef-starter.zip

The above command will create a directory called chef-repo in the current location. This directory contains all the basic settings for interacting with chef server from workstation.

It also has a cookbook directory inside, where all cookbooks will be stored and later pushed to central chef server.

Let's get inside the "chef-repo" directory, and see what's inside.

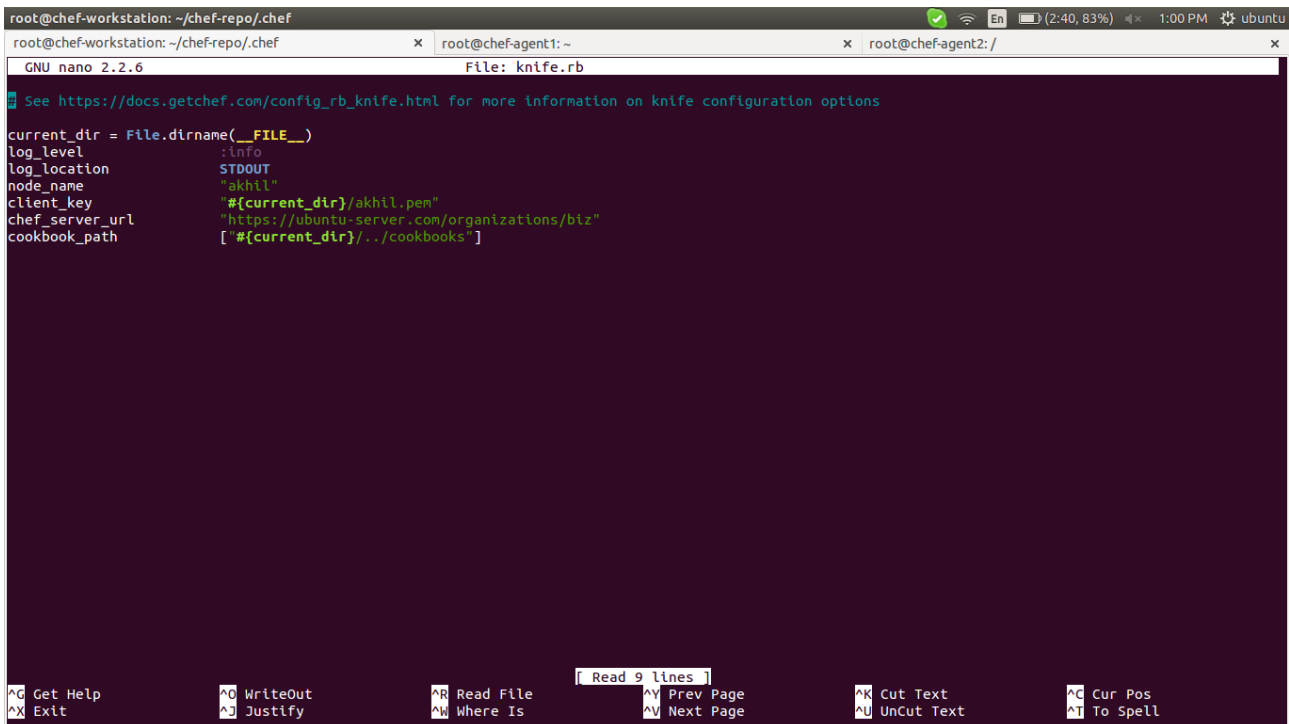
```
root@chef-workstation: ~/chef-repo
root@chef-workstation: ~/chef-repo x root@chef-agent1: ~ x root@chef-agent2: / x
root@chef-workstation:~/chef-repo# ls
README.md cookbooks roles
root@chef-workstation:~/chef-repo# ll
total 28
drwxr-xr-x 5 root root 4096 Oct 20 05:31 ./
drwx----- 4 root root 4096 Oct 20 05:31 ../
drwxr-xr-x 3 root root 4096 Oct 20 06:04 .chef/
-rw-r--r-- 1 root root 495 Oct 19 22:26 .gitignore
-rw-r--r-- 1 root root 2341 Oct 19 22:26 README.md
drwxr-xr-x 3 root root 4096 Oct 19 22:26 cookbooks/
drwxr-xr-x 2 root root 4096 Oct 19 22:26 roles/
root@chef-workstation:~/chef-repo#
```

There is a directory called ".chef" as evident from the directory listing above. This .chef directory contains the private key for the user. Also this ".chef" directory contains knife configuration file called knife.rb.

The trusted_certs directory created only after run some commands for connecting to server, it will come later in this documentation

```
root@chef-workstation: ~/chef-repo/.chef
root@chef-workstation: ~/chef-repo/.chef x root@chef-agent1: ~ x root@chef-agent2: / x
root@chef-workstation:~/chef-repo/.chef# ll
total 20
drwxr-xr-x 3 root root 4096 Oct 20 06:04 ./
drwxr-xr-x 5 root root 4096 Oct 20 05:31 ../
-rw-r--r-- 1 root root 1678 Oct 19 22:26 akhil.pem
-rw-r--r-- 1 root root 418 Oct 19 22:26 knife.rb
drwxr-xr-x 2 root root 4096 Oct 20 06:04 trusted_certs/
root@chef-workstation:~/chef-repo/.chef#
```


Knife.rb file will be read by knife tool while interacting with central chef server from workstation. The contents of knife.rb file looks like the below.



```
root@chef-workstation: ~/chef-repo/.chef
root@chef-workstation: ~/chef-repo/.chef
GNU nano 2.2.6 File: knife.rb
See https://docs.getchef.com/config_rb_knife.html for more information on knife configuration options

current_dir = File.dirname(__FILE__)
log_level      :info
log_location   STDOUT
node_name      "akhil"
client_key     "#{current_dir}/akhil.pem"
chef_server_url "https://ubuntu-server.com/organizations/biz"
cookbook_path  ["#{current_dir}/../cookbooks"]

^G Get Help      ^O WriteOut      ^R Read File      Read 9 lines
^X Exit          ^J Justify       ^W Where Is      ^Y Prev Page
                  ^K Cut Text      ^V Next Page
                  ^U UnCut Text    ^C Cur Pos
                  ^T To Spell
```

client_key: Specifies the path of the private key file associated with the user. This private key will be used to authenticate workstation against the chef server.

chef_server_url: This is the full URL of the chef server with the organization path(we did create an organization during our previous tutorial of installing chef server. Our organization's name is "slashroot")

cookbook_path: The absolute path where cookbooks will be stored on the workstation. This is because knife will be syncing cookbooks to server, it will also be downloading cookbooks, will be creating new cookbooks etc.

client_name: This is the user name associated with the organization. We created a user named "sarath" in our previous tutorial.

Once all the above mentioned items are configured and ready. Let's verify the connectivity between chef workstation and chef server. As mentioned earlier, knife is going to be our tool while interacting with central chef.

Note :::

Please remember the fact that you should execute knife commands only after navigating to the chef-repo directory. This is because knife looks for a directory called .chef, which contains our knife.rb settings file.

knife ssl fetch

The above command interacts with the chef server (by using the URL defined inside knife.rb file), and grabs the SSL certificates of the server. This SSL certificate is then stored inside a new directory called trusted_certs.

Once the SSL certificates are added to the trusted list of knife. You can then verify the connectivity to central chef server using the below command.

knife ssl check

*Connecting to host ubuntu-server.com:443
Successfully verified certificates from `ubuntu-server.com'*

If things are correctly configured..You should be able to see the above output. Now knife can do all the operations on the chef server using the user we have created(which is defined inside knife.rb), and the organization associated with that user.

Links ::

workstation configuration >>

<http://www.slashroot.in/how-setup-and-configure-chef-workstation>

kitchen installation>>

<https://learn.chef.io/local-development/rhel/get-started-with-test-kitchen/>

<https://docs.chef.io/kitchen.html>

<https://www.safaribooksonline.com/blog/2014/11/27/opscode-chef-vagrant-test-kitchen/>

<http://kitchen.ci/docs/getting-started/>

<http://kitchen.ci/>

Bootstrap a node using chef workstation

Bootstrapping a node is nothing but the job of installing and configuring chef agent on a server that needs to be automated via chef. Installing and configuring a node with chef agent which will then start pulling configuration from central chef server, is a single knife command away as shown below.

Before that tell to the client about ip and FQDN of chef-server

nano /etc/hosts

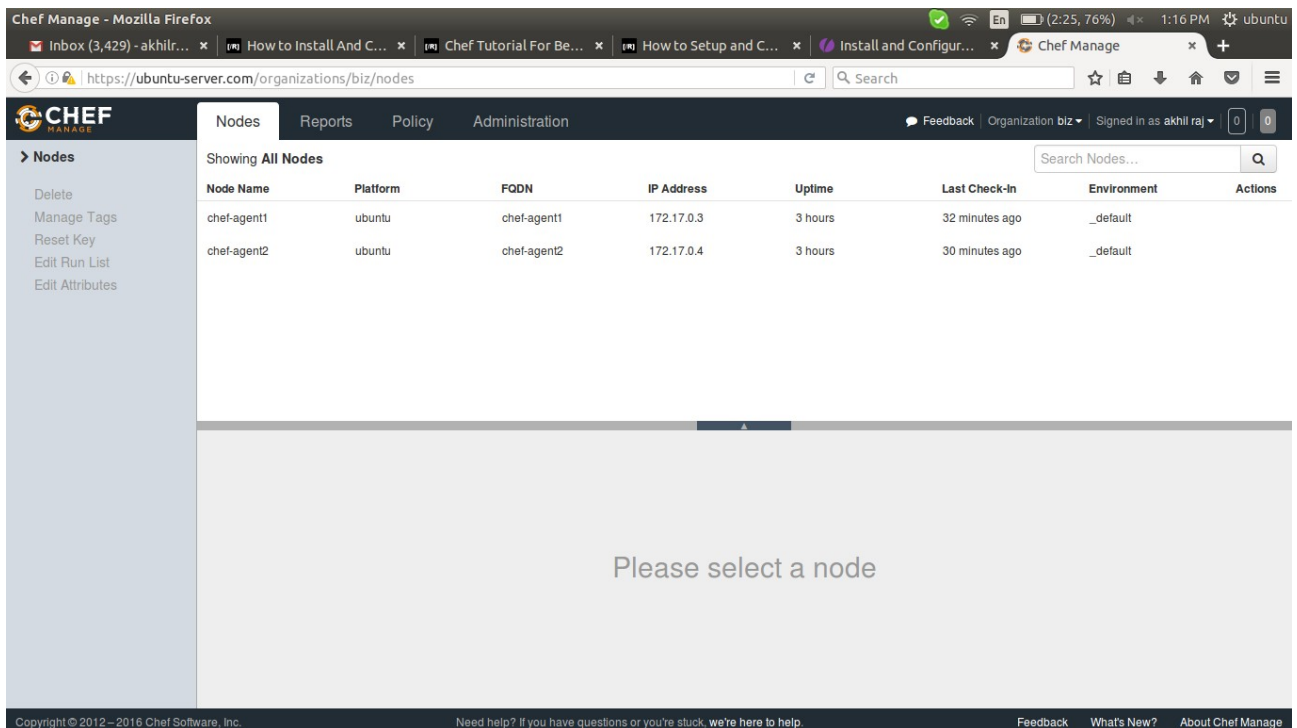
192.168.1.64 ubuntu-server.com

knife bootstrap 172.17.0.4 --sudo --ssh-user chef --ssh-password chef --node-name chef-agent2 where,

172.17.0.4 >> **node ip**
--sudo >> **used for fill permission for user in that node**
chef >> **user-name**
chef >> **password for user # username and password are same for my scenario**
chef-agent2 >> **node name for that node # use FQDN of that node for better understanding**

```
root@chef-workstation: ~/chef-repo
root@chef-workstation: ~/chef-repo x root@chef-agent1: ~ x root@chef-agent2: / x
root@chef-workstation:~/chef-repo# knife bootstrap 172.17.0.4 --sudo --ssh-user chef --ssh-password chef --node-name chef-agent2
Creating new client for chef-agent2
Creating new node for chef-agent2
Connecting to 172.17.0.4
172.17.0.4 knife sudo password:
172.17.0.4 -----> Installing Chef Omnibus (-v 12)
172.17.0.4 downloading https://omnitruck-direct.chef.io/chef/install.sh
172.17.0.4 to file /tmp/install.sh.3875/install.sh
172.17.0.4 trying wget...
172.17.0.4 ubuntu 14.04 x86_64
172.17.0.4 Getting information for chef stable 12 for ubuntu...
172.17.0.4 downloading https://omnitruck-direct.chef.io/stable/chef/metadata
a7ve128pubuntu8py=14.048m=x86_64
172.17.0.4 to file /tmp/install.sh.3879/metadata.txt
172.17.0.4 trying wget...
172.17.0.4 sha1 a2de7d933734d3a0c4b859576d0be472c5cd55f7
172.17.0.4 sha256 7073541beb4294c994d4035a49afcf06ab45b3b3933b98a65b8059b7591df6b8
172.17.0.4 url https://packages.chef.io/files/stable/chef/12.15.19/ubuntu/14.04/chef_12.15.19-1_and64.deb
172.17.0.4 version 12.15.19
172.17.0.4 downloaded metadata file looks valid...
172.17.0.4 downloading https://packages.chef.io/files/stable/chef/12.15.19/ubuntu/14.04/chef_12.15.19-1_and64.deb
172.17.0.4 to file /tmp/install.sh.3879/chef_12.15.19-1_and64.deb
172.17.0.4 trying wget...
172.17.0.4 Comparing checksum with sha256sum...
172.17.0.4 Installing chef 12
172.17.0.4 installing with dpkg...
172.17.0.4 Selecting previously unselected package chef.
(Reading database ... 16444 files and directories currently installed.)
172.17.0.4 Preparing to unpack .../chef_12.15.19-1_and64.deb ...
172.17.0.4 Unpacking chef (12.15.19-1) ...
172.17.0.4 Setting up chef (12.15.19-1) ...
172.17.0.4 Thank you for installing Chef!
172.17.0.4 Starting the first Chef Client run...
172.17.0.4 Starting Chef Client, version 12.15.19
172.17.0.4 resolving cookbooks for run list: []
172.17.0.4 Synchronizing Cookbooks:
172.17.0.4 Installing Cookbook Gems:
172.17.0.4 Compiling Cookbooks...
172.17.0.4 [2016-10-20T07:16:38+00:00] WARN: Node chef-agent2 has an empty run list.
172.17.0.4 Converging 0 resources
172.17.0.4
172.17.0.4 Running handlers:
172.17.0.4 Running handlers complete
172.17.0.4 Chef Client finished, 0/0 resources updated in 03 seconds
```

by this way we can many nodes with company requirement



The screenshot shows the Chef Manage web interface in a Mozilla Firefox browser. The address bar displays the URL <https://ubuntu-server.com/organizations/biz/nodes>. The interface has a dark header with the Chef logo and navigation tabs: Nodes, Reports, Policy, and Administration. The 'Nodes' tab is active. On the left, a sidebar lists actions: Delete, Manage Tags, Reset Key, Edit Run List, and Edit Attributes. The main content area shows 'Showing All Nodes' with a search bar. Below is a table with two nodes:

Node Name	Platform	FQDN	IP Address	Uptime	Last Check-in	Environment	Actions
chef-agent1	ubuntu	chef-agent1	172.17.0.3	3 hours	32 minutes ago	_default	
chef-agent2	ubuntu	chef-agent2	172.17.0.4	3 hours	30 minutes ago	_default	

Below the table, a large grey box contains the text 'Please select a node'. The footer includes copyright information (© 2012–2016 Chef Software, Inc.), a help link, and navigation links for Feedback, What's New?, and About Chef Manage.

links ::

node installation >>

<http://www.slashroot.in/how-setup-and-configure-chef-workstation>

Create and Upload a cookbook to chef server from workstation

The second operation that the chef workstation does is to create a cookbook which acts as the main basic block of configuration that will be applied on nodes.

You can create a skeleton cookbook by using the below command.

knife cookbook create name

note ::

Again..Please do not forget the fact that knife commands needs to from chef-repo directory(as it requires .chef directory, which has the knife.rb settings file)

You can then upload this cookbook that we just created to central chef server using the below command.

knife cookbook upload name

for testing from work-station :

knife bootstrap 172.17.0.4 --sudo --ssh-user chef --ssh-password chef --node-name chef-agent2 --run-list 'recipe[user]'

or run the following command in node side

chef-client

to set the run list in server

knife node run_list set chef-agent2 'recipe[user]'

knife node run_list add chef-agent2 'recipe[user]'

to edit a node info

knife node edit node-name -a

if found some error related to editor

export "EDITOR=nano"

links ::

<http://stackoverflow.com/questions/17562837/chef-how-to-set-editor-for-knife>

to list nodes

knife node list -w

to include another recipe in one recipe

1) create a cookbook

2) create a recipe (default.rb)

3) add a line in 2nd recipe file (default.rb) as follow

include_recipe 'another-cookbook-name'

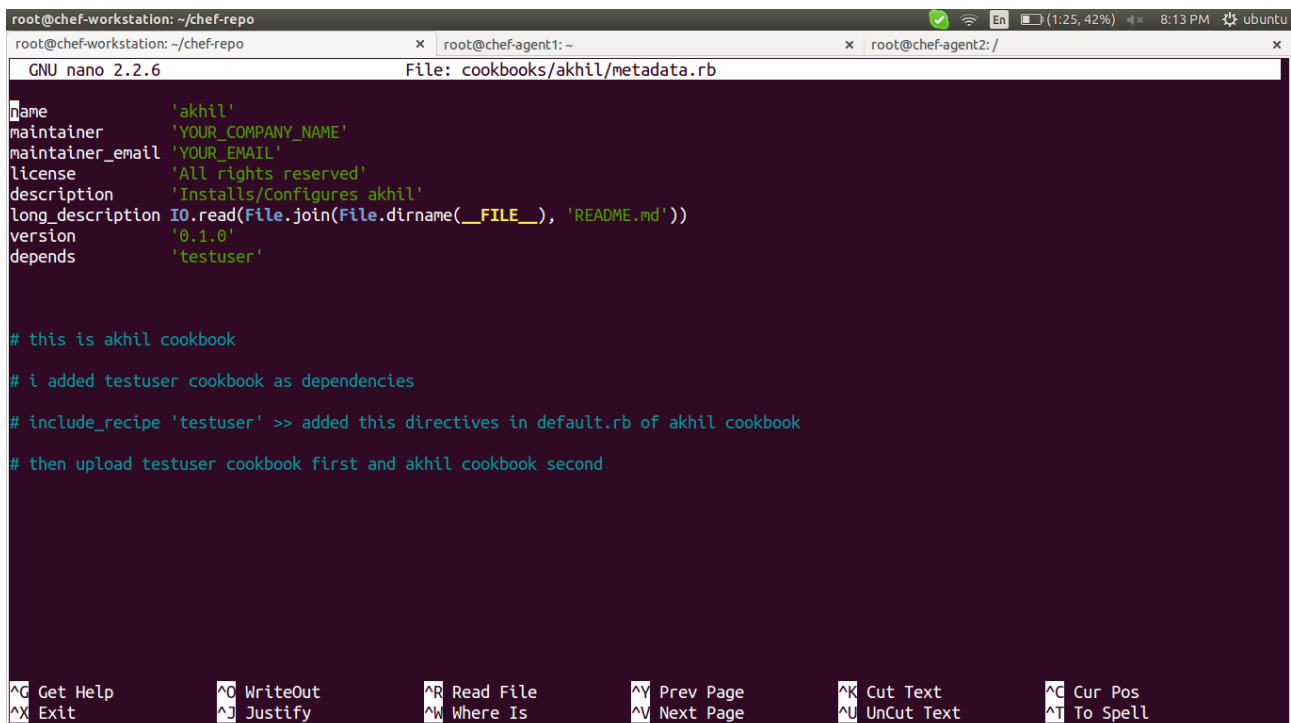
4) edit metadata.rb of the cookbook which have include_recipe directives in default.rb and add the

following line.

depends cookbook name

5) upload both cookbook to chef-master (there is a order for uploading, see below figure)

6) done



```
root@chef-workstation: ~/chef-repo
root@chef-workstation: ~/chef-repo x root@chef-agent1: ~ x root@chef-agent2: /
GNU nano 2.2.6 File: cookbooks/akhil/metadata.rb

name 'akhil'
maintainer 'YOUR_COMPANY_NAME'
maintainer_email 'YOUR_EMAIL'
license 'All rights reserved'
description 'Installs/Configures akhil'
long_description IO.read(File.join(File.dirname(__FILE__), 'README.md'))
version '0.1.0'
depends 'testuser'

# this is akhil cookbook

# i added testuser cookbook as dependencies

# include_recipe 'testuser' >> added this directives in default.rb of akhil cookbook

# then upload testuser cookbook first and akhil cookbook second

^G Get Help      ^O WriteOut     ^R Read File    ^Y Prev Page    ^K Cut Text      ^C Cur Pos
^X Exit          ^J Justify      ^W Where Is     ^V Next Page    ^U UnCut Text   ^T To Spell
```

link :

<http://stackoverflow.com/questions/16947393/should-i-use-include-recipe-or-add-the-recipe-to-run-list>

check the server whether the cookfile uploaded or not in policy

The screenshot shows the Chef Manage web interface in a Mozilla Firefox browser. The address bar displays the URL `https://ubuntu-server.com/organizations/biz/cookbooks`. The interface has a dark header with the Chef logo and navigation tabs: Nodes, Reports, Policy (selected), and Administration. On the right of the header, there are links for Feedback, Organization biz, and a user profile for 'akhil raj'. A left sidebar contains a menu with options: Cookbooks, Roles, Data Bags, Environments, and Clients. The main content area is titled 'Showing All Cookbooks' and contains a table with two columns: 'Cookbook' and 'Current Version'. The table lists three entries: 'akhil' with version '0.1.0', 'apache' with version '0.1.0', and 'user' with version '0.1.0'. Below the table, a large grey box contains the text 'Please select a cookbook'. The footer includes copyright information for 2012-2016 Chef Software, Inc., a help link, and additional links for Feedback, What's New?, and About Chef Manage.

Cookbook	Current Version
akhil	0.1.0
apache	0.1.0
user	0.1.0

then take node

The screenshot shows the Chef Manage web interface in a Mozilla Firefox browser. The address bar displays the URL `https://ubuntu-server.com/organizations/biz/nodes`. The interface has a dark header with the Chef logo and navigation tabs: Nodes (selected), Reports, Policy, and Administration. On the right of the header, there are links for Feedback, Organization biz, and a user profile for 'akhil raj'. A left sidebar contains a menu with options: Nodes, Delete, Manage Tags, Reset Key, Edit Run List, and Edit Attributes. The main content area is titled 'Showing All Nodes' and includes a search bar labeled 'Search Nodes...'. Below the search bar is a table with columns: Node Name, Platform, FQDN, IP Address, Uptime, Last Check-In, Environment, and Actions. The table lists two nodes: 'chef-agent1' with platform 'ubuntu', FQDN 'chef-agent1', IP '172.17.0.3', uptime '3 hours', last check-in '8 hours ago', and environment '_default'; and 'chef-agent2' with platform 'ubuntu', FQDN 'chef-agent2', IP '172.17.0.4', uptime '8 hours', last check-in '2 hours ago', and environment '_default'. Below the table, a large grey box contains the text 'Please select a node'. The footer includes copyright information for 2012-2016 Chef Software, Inc., a help link, and additional links for Feedback, What's New?, and About Chef Manage.

Node Name	Platform	FQDN	IP Address	Uptime	Last Check-In	Environment	Actions
chef-agent1	ubuntu	chef-agent1	172.17.0.3	3 hours	8 hours ago	_default	
chef-agent2	ubuntu	chef-agent2	172.17.0.4	8 hours	2 hours ago	_default	

then click on right most side of any node and choose edit run list

The screenshot shows the Chef Manage web interface in a Mozilla Firefox browser. The URL is <https://ubuntu-server.com/organizations/biz/nodes/chef-agent1>. The 'Nodes' tab is selected, displaying a table of nodes. The table has columns: Node Name, Platform, FQDN, IP Address, Uptime, Last Check-in, Environment, and Actions. Two nodes are listed: 'chef-agent1' and 'chef-agent2'. The 'chef-agent1' row is highlighted. A dropdown menu is open for the 'chef-agent1' node, showing options: Delete, Manage Tags, Reset Key, Edit Run List (highlighted), and Edit Attributes. Below the table, the 'Node: chef-agent1' details are visible, including 'Last Check-in: 8 Hours Ago', 'Uptime: 3 Hours', and 'Environment: _default'.

Node Name	Platform	FQDN	IP Address	Uptime	Last Check-in	Environment	Actions
chef-agent1	ubuntu	chef-agent1	172.17.0.3	3 hours	8 hours ago	_default	[Settings Icon]
chef-agent2	ubuntu	chef-agent2	172.17.0.4	8 hours	2 hours ago	_default	[Settings Icon]

then add the cookbook which needed for this particular node by drag the cookbook from available recipes to current run list

The screenshot shows the same Chef Manage interface, but with the 'Edit Node Run List' modal dialog open. The dialog has a title bar 'Edit Node Run List' and a close button. It contains two main sections: 'Available Roles' and 'Current Run List'. The 'Available Roles' section is empty. The 'Current Run List' section contains the role 'apache'. Below these sections, there is a search bar for 'Available Recipes' with the results 'akhil' and 'user' listed. At the bottom of the dialog are 'Cancel' and 'Save Run List' buttons. The background shows the 'Nodes' tab with the 'chef-agent1' node selected.

RECIPE

******Recipe******

to install package

```
package 'package-name' do
  action 'install'
end
```

******Recipe******

to start the service

```
service 'service-name' do
  action 'start'      # action [ :enable, :start ] for start at booting.
end
```

******Recipe******

to create a file

```
file '/path/file-name' do
  content 'some-data'
  owner 'root'
  group 'root'
  mode '777'
end
```

******Recipe******

to add some file from master,

use
cookbook_file instead of file in above
and add some file in files/default directory of cookbook

or

template instead of file in above
and add some file in templates/default directory of cookbook

```
cookbook_file '/path/file-name' do
```

```
    source 'file-name-in /files/default/ '
    owner 'root'
    group 'root'
    mode '777'
end

template '/path/file-name' do
    source 'file-name-in /templates/default/ '
    owner 'root'
    group 'root'
    mode '777'
end
```

*****Recipe*****

to create directory

```
directory '/path/dir-name' do
    owner 'root'
    group 'root'
    mode '777'
end
```

*****Recipe*****

to create a group

```
group 'group-name' do
    action 'create'
    gid '1500'
end
```

*****Recipe*****

to create a user

```
user 'user-name' do
    action 'create'
    uid '1500'
    gid '1500'           # use group if create a group already with this gid
    password 'any'
    home '/home/user-name'
    shell '/bin/bash'
end
```

*****Recipe*****

to notify

```
notifies :restart, 'service[apache2]'          # if there is a service in the name of apache2
```

*****Recipe*****

to execute a command

```
execute 'update-upgrade' do
  command 'apt-get update && apt-get upgrade -y'
  environment 'path' => '/bin:/sbin:/usr/bin:/usr/sbin'      # optional, will work without path also
  action 'run'
end
```

*****Recipe*****

```
notifies          # notify some command when change in this command
subscribes         # notify this command when change in another command
```

```
:before
:immediate
:delayed
```

eg:

```
notifies :action, 'resource[name]', :timer      # action specifies for that command not for this
                                                command
```

```
notifies :restart, 'file[/var/www/html/index.html]', :immediate
```

```
subscribes :action, 'resource[name]', :timer    # action specifies for this command not for that
                                                command
```

```
subscribes :restart, 'file[/var/www/html/index.html]', :immediate
```

*****Recipe*****

notifies

Ruby Type: Symbol, 'Chef::Resource[String]'

A resource may notify another resource to take action when its state changes. Specify a 'resource[name]', the :action that resource should take, and then the :timer for that action. A resource may notify more than one resource; use a notifies statement for each resource to be notified.

A timer specifies the point during the chef-client run at which a notification is run. The following

timers are available:

:before

Specifies that the action on a notified resource should be run before processing the resource block in which the notification is located.

:delayed

Default. Specifies that a notification should be queued up, and then executed at the very end of the chef-client run.

:immediate, :immediately

Specifies that a notification should be run immediately, per resource notified.

The syntax for notifies is:

notifies :action, 'resource[name]', :timer

to do this command before some command

notifies :restart, 'file[/var/www/html/index.html]', :immediate

******Recipe******

subscribes

Ruby Type: Symbol, 'Chef::Resource[String]'

A resource may listen to another resource, and then take action if the state of the resource being listened to changes. Specify a 'resource[name]', the :action to be taken, and then the :timer for that action.

A timer specifies the point during the chef-client run at which a notification is run. The following timers are available:

:before

Specifies that the action on a notified resource should be run before processing the resource block in which the notification is located.

:delayed

Default. Specifies that a notification should be queued up, and then executed at the very end of the chef-client run.

:immediate, :immediately

Specifies that a notification should be run immediately, per resource notified.

subscribes :action, 'resource[name]', :timer

eg ::

subscribes :restart, 'file[/var/www/html/index.html]', :immediate

******Recipe******

Arguments

The following arguments can be used with the `not_if` or `only_if` guard properties:

`:user`

Specify the user that a command will run as. For example:

```
not_if 'grep adam /etc/passwd', :user => 'adam'
```

`:group`

Specify the group that a command will run as. For example:

```
not_if 'grep adam /etc/passwd', :group => 'adam'
```

`:environment`

Specify a Hash of environment variables to be set. For example:

```
not_if 'grep adam /etc/passwd', :environment => {  
  'HOME' => '/home/adam'  
}
```

`:cwd`

Set the current working directory before running a command. For example:

```
not_if 'grep adam passwd', :cwd => '/etc'
```

******Recipe******

******Links******

recipe >>

https://docs.chef.io/resource_execute.html

source-file >>

https://docs.chef.io/resource_cookbook_file.html

execute >>

https://docs.chef.io/resource_execute.html

user-creation>>

https://docs.chef.io/resource_user.html

running nodes >>

<https://www.linode.com/docs/applications/chef/creating-your-first-chef-cookbook>

knife and recipe >>

https://docs.chef.io/knife_node.html