**CHEF INTERMIDIATE LEVEL (Version 11)**

**Audience**: - This is not an introductory document on Chef. This document is designed for System Engineers/Devops guys who having some basic Chef working knowledge and able to understand basic terminologies of chef (e.g. recipe, resource etc.). Basic knowledge of Linux/Unix and some OOPS background (ruby preferred but we don’t have to be expert in it) and Git knowledge is required.

**Objective**: How we can deploy chef in our environment and manage more efficiently.

**Pre-requisites:** We are using Linux OS (CentOS, RHEL) platform for this document.

1. Download the Chef Server and Client RPMs packages from the [chef site.](https://downloads.chef.io/chef-server/)
2. Download Git or install it via yum (yum install git –y).
3. Make sure your chef server has at least 2 GB of memory and 10 GB of disk space (minimum).
4. Host should have fully configured hostname.

# Environment and Installation set up:

### For Chef- Server:

#rpm –ivh chef-server-11.4.x86.rpm

#chef-server-ctl reconfigure

### For workstation:

#mkdir ~/.chef

#scp root@chef-server:/etc/chef-server/admin.pem ~/.chef

# scp root@chef-server:/etc/chef-server/chef-validator.pem ~/.chef

# cat ~/.chef/knife.rb

log\_level :info

log\_location STDOUT

node\_name 'admin'

client\_key '/root/.chef/admin.pem'

validation\_client\_name 'chef-validator'

validation\_key '/root/.chef/admin.pem'

chef\_server\_url 'https://chef-server.example.com:443/'

syntax\_check\_cache\_path '/root/.chef/syntax\_check\_cache'

cookbook\_path [ '/var/chef/cookbooks' ]

knife[:editor] = ‘/usr/bin/vim’

### Chef Node Installation:

#rpm –ivh chef-client.rpm

#vim /etc/chef/client.rb

log\_level :info

log\_location STDOUT

chef\_server\_url ‘https://FQDN-OR-IP-OF-CHEF-SERVER'

Now we have chef server/workstation/node installation completed.

# Operational Manual:

### Add Node to Chef-Server:

To add a node to chef server we have two methods.

1. Manual Process.

Place the chef-validator file to /etc/chef of node.

#scp root@chef-server:/etc/chef-server/chef-validator.pem /etc/chef

And make changes in /etc/chef/client.rb file as mentioned in 1.1.3.

1. Boot strapping.

There is command line mode by which we can remotely install and add node to chef –server.

#knife bootstrap <IP of node > - N <FQDN of node > -x root –P <Password>

Note- We should choose bootstrapping method to add a node.

### Setting up Git repo for chef Environment: To set up Git we need to install Git package on workstation. We can install it via YUM.

#yum install git –y

Go to ‘/var/chef/cookbooks/’ directory and run below commands.

#git init

Note- You’ve now got a local git repository. You can use git locally, like that, if you want. But if you want the thing to have a home on github, do the following.

1. Go to github.
2. Log in to your account.
3. Click the new repository button in the top-right.
4. Click the “Create repository” button.
5. Now, follow the second set of instructions, “Push an existing repository…”
6. # git remote add origin git@github.com:username/new\_repo\_name
7. # git push -u origin master

#Every changes we made in this directory we have to put it under Version control. i.e

#git add

#git commit

Now you can access your Git repo remotely and can make changes from anywhere, it’s most preferred way too**.**

### Creation of cookbook:

#knife node create <cookbook\_name> for example below; I have created a cookbook with name cookbook-test.

#knife cookbook create cookbook-test

This will create cookbook with the name you specify in ‘/var/chef/cookbook’

# cd /var/chef/cookbooks

# tree cookbook-test

cookbook-test/

├── attributes

├── CHANGELOG.md

├── definitions

├── files

│ └── default

├── libraries

├── metadata.rb

├── providers

├── README.md

├── recipes

│ └── default.rb

├── resources

└── templates

└── default

This is the default skeleton of a cookbook.

### Creation of recipe:

All the recipes are placed inside *recipes* directory in cookbook.

*“Recipe files are Ruby applications that define everything that is required to configure a system, including creating and configuring folders, installing and configuring packages, starting services, and so on. A recipe is a subset or “piece” of a cookbook”*

Below is the chef recipe for user creation, Here we are also giving a password to user *test* so for that we have to generate a password.

# openssl passwd -1 "shadwopassword"

>$1$QwuUa80Z$KZkYq8CYnjkdfHKyIsK1tHZ7s0

#cat /var/chef/cookbooks/cookbook-test/recipes/default.rb

group "system-admins" do

gid 1001

end

user "test" do

comment "First test user"

shell "/bin/bash"

home "/home/test"

gid "system-admins"

uid 1002

supports :manage\_home => true

password "$1$QwuUa80Z$KZkYq8CYnjkdfHKyIsK1tHZ7s0"

end

#git add –all

#git commit –m “added recipe for test user creation”

Note- make sure you make version changes in metadata.rb file before committing.

Now upload the new cookbook/recipe to chef server.

#knife cookbook upload cookbooks cookbook-test

### Creation of roles:

In Simple term “*Roles are collection of cookbooks/recipes” .*We can apply these roles to any node and change the attributes values if necessary.

*“Attributes in chef are key value parameters and used by the recipes and templates.”*

#knife role create role-test

Here we are creating a role name “role-test”

To add cookbooks/recipes in it, juts simply edit it.

#Knife role edit role-test

And put the name of cookbook/recipes in it.

To apply the roles to any node, just simply edit the node with knife i.e.

#knife node edit <node\_name>

And place the “role [role-test]” in its run list.

:wq

### Creation of Environment:

An environment is a way to map an organization’s real-life workflow to what can be configured and managed when using Chef Server. Every organization begins with a single environment called the \_default environment, which cannot be modified (or deleted). Additional environments can be created to reflect each organization’s patterns and workflow. For example, creating production, staging, testing, and development environments. Generally, an environment is also associated with one (or more) cookbook versions.

#knife environment create Production

Here we are creating environment name “production” for our Production nodes. We can add nodes in this environment and specify the desired cookbooks, roles that we want to run on Production nodes.

To add nodes to this environemt simply edit the nodes via knife command.

#knife node edit client.domain #Here cliet.domain is our node name.

{

"name": "client.domain",

"chef\_environment": "Production",

"normal": {

"tags": [

]

},

"run\_list": [

"recipe[cookbook-test]"

]

}

Note- We can specify desired cookbook version in environment and attributes values as well.

#knife environment edit Production

{

"name": "Production",

"description": "For Production Machines",

"cookbook\_versions": {

**“cookbook-test”: “0.1.0”**

},

"json\_class": "Chef::Environment",

"chef\_type": "environment",

"default\_attributes": {

},

"override\_attributes": {

}

}

### Creation of Data Bags:

To store the password and other secret information we use Data bags in chef, it encrypt the password so that it can’t be read by others. In general 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.

For example we have created a recipe for user creation and we have provided user “test” password in sadow format but it can be read by anyone if out workstation or node machine is compromised so to avaoid that scenario we create a encrypted data bag so even in cse of workstation get compromised out secret password will remain intact.

First lets create a key to encrypt our secret data bag .

#openssl rand –base64 512 |tr –d ‘\r\n’ > secret\_key

Here we have generated a secret key file and stored in “secret\_key” file trimmed all the newline characters.

Let’s create a data bag which will have test user password.

#knife data bag create user\_passwd

And create a data bag item as well

#knife data bag create user\_passwd test\_user --secret-file /path\_to\_secret\_key

So now the details are encrypted and node can read the password information.

# knife data bag show create user\_passwd test\_user

id: test\_user

passwd:

cipher: aes-256-cbc

encrypted\_data: 5bwD6MtqlBgfjT8e2XQN2zEND+ppWZva6/p2mWgDIrk=

iv: NFXcXIHsyBih/5t/QH3w+A==

version: 1

We can call this information in our recipe for password information. E.g.

#vim /var/chef/cookbooks/cookbook-test/recipes/default.rb

Password=EncryptedDataBagItem.load(user\_passwd, test\_user)

group "system-admins" do

gid 1001

end

user "test" do

comment "First test user"

shell "/bin/bash"

home "/home/test"

gid "system-admins"

uid 1002

supports :manage\_home => true

password **Password[‘password]**

end

**Note**- we also need to copy the secrete key to the nodes as well so they can decrypt it , Just copy the secret\_key to node under /etc/chef/encypted\_data\_bag\_secret , The name should be same “encypted\_data\_bag\_secret”. Now you can remove the “secret\_key” file from your workstation.

### Creation of templates.

*Template files are templates that recipes use to create other files, such as configuration files. Template files typically let you modify the configuration file by overriding attributes—which can be done without touching the cookbook—instead of rewriting a configuration file. The standard practice is that whenever you expect to change a configuration file on an instance even slightly, you should use a template file.*

There is directory called templates in cookbook skeleton, we can put the templates there and push them to the nodes via recipes. Templates are blue-prints. It could be any apache file or any other configuration file.

Let’s create a simple template for index.html.

Go to template directory under default and create a file index.html.erb , here the extension of template file will always .erb(Embedded ruby)

#vim cookbook-test/templates/default/index.html.erb

Hi, this is Index page for apache.

:wq

Make below changes in recipe:

template “/var/www/html/index.html” do

source “index.html.erb”

action :create

owner ‘root’

mode ‘755 ‘

end

Note- Here if you notices we have given full path of index.html file which is required. Default action of template is create so it’s optional.

# Example for better Understanding.

Let’s create a cookbook for NTP server which will configure and start NTP server on nodes.

#knife cookbook create my-ntp-server

#vim my-ntp-server/attributes/default.rb

default[:ntp][:servers] = ["8.8.8.8"]

case platform

when "redhat","centos","fedora","scientific"

default[:ntp][:service] = "ntpd"

when "ubuntu","debian"

default[:ntp][:service] = "ntp"

else

default[:ntp][:service] = "ntpd"

end

**Note**- Here if we see, we have mention NTP server address 8.8.8.8 and as per platform NTP service name. So we will call these values in out recipes and templates.

#vim my-ntp-server/templates/default/ntp.conf.rb

# Generated by Chef for <%= node[:fqdn] %>

# node[:fqdn] = ohai data collected on node !

# Local modifications will be overwritten.

restrict -6 ::1

<% node[:ntp][:servers].each do |ntpsrv| -%>

server <%= ntpsrv %> iburst

restrict <%= ntpsrv %> nomodify notrap noquery

<% end -%>

restrict default kod nomodify notrap nopeer noquery

restrict -6 default kod nomodify notrap nopeer noquery

restrict 127.0.0.1

server 127.127.1.0 # local clock

driftfile /var/lib/ntp/drift

keys /etc/ntp/keys

#vim my-ntp-server/recipes/ntp.rb

yum\_package "ntp" do

action [:install]

end

template "/etc/ntp.conf" do

source "ntp.conf.erb"

owner "root"

group "root"

mode 0644

notifies :restart, resources(:service => node[:ntp][:service])

end

service node[:ntp][:service] do

service\_name node[:ntp][:service]

action [:enable,:start,:restart]

end

Let’s discuss each portion (resource) one by one. Firstly we are installing NTP package on nodes via YUM. Second we are placing out configuration file from template. And last it will start the NTP service on the node.

**Note**- In template we are using ERB so it seems little confusing and complicated but if we look closer we are just using loops and calling variables from attributes.

Now we can add this run list in node and run chef-client that will make necessary changes on the node.

# Miscellaneous commands

### Knife search

We can use this command to search the node’s attributes and its run\_list information.

#knife search "name:<write the name of node that you want to search>"

#knife search "platform:ubuntu\*"

#knife search "platform:\*" -a macaddress

#knife search "platform:ubuntu\*" -a uptime

#knife search "platform:ubuntu\*" -a virtualization.system

#knife search "platform:ubuntu\*" -a network.default\_gateway

#knife search "platform:\*" -a languages.ruby.version

#knife search "platform:\*" -a kernel.machine

#knife search users "groups:admin" -i

## SSH Examples

#knife ssh "name: <write the name of node to ssh " -x root -P PASSWORD uptime

#knife ssh "NOT platform:win\*" -P PASSWORD ls

#knife ssh "chef\_environment:Helix-Uptime-VM and NOT platform:win\*" \

-x root -P PASSWORD uptime

##Chef Client

How do I automatically bootstrap a linux box?

#knife bootstrap 10.0.0.2 -N client.domain -x root -P PASSWORD

How do I make the chef client run so it will apply my new recipes on Linux nodes?

#knife ssh "NOT platform:win\*" -x root -P PASSWORD chef-client

### Knife exec

##Cookbooks

How do I remove a recipe from every node which is in dev environment?

#knife exec -E "nodes.transform(“chef\_environment:dev“) \

{|n| puts n.run\_list.remove(“recipe[chef-client::upgrade]“); n.save }"

How do I remove a role from every node?

#knife exec -E "nodes.find(“role:web\_server”) \

{|n| n.run\_list.remove(“role[web\_server]“); }"

How do I add a node in environment?

#knife exec -E "nodes.find(“fqdn :client.domain”) \

{|n| n.chef\_environment(“dev”);n.save }"

##Nodes

#Move all the nodes in \_default environment to the production-VM environment.

#knife exec -E "nodes.transform(“chef\_environment:\_default”) \

{ |n| n.chef\_environment(“production-VM”) }"

Remove a recipe from all nodes in an Environment

#knife exec -E "nodes.transform(“chef\_environment:dev”) \

{|n| puts n.run\_list.remove(“recipe[chef-client::upgrade]“); n.save }"

#Roles

How do I add a Linux role to all the boxes that are not window?

#knife exec -E "nodes.transform(“NOT platform:win\*“) \

{|n| puts n.run\_list << “role[linux]“; n.save }"

How do I remove a Linux role from all the boxes that are not windows?

#knife exec -E "nodes.transform(“NOT platform:win\*“) \

{|n| puts n.run\_list.remove(“role[linux]“); n.save }"

Remove all nodes from a given role

#knife exec -E "nodes.find(“role:web\_server”) \

{|n| n.run\_list.remove(“role[web\_server]“); n.save}"

How do I add a role to an environment?

#knife exec -E 'nodes.transform("chef\_environment:my-production-env") \

{|n| puts n.run\_list << "role[hosts\_file]"; n.save }'