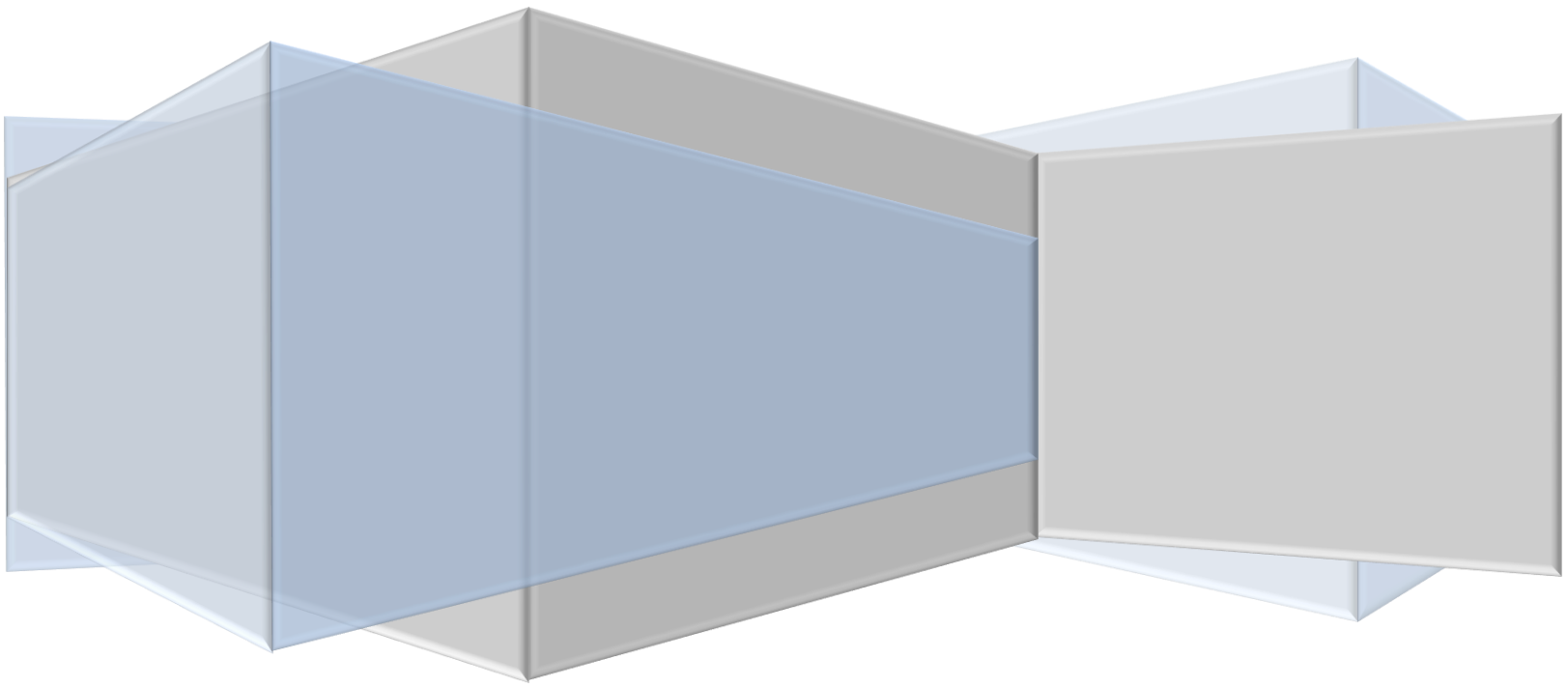
AutomationFactory

Chef Essentials

Chef-server, workstation, node and Chef Tools

Ganesh Palnitkar



Chef-server, workstation, node and Chef Tools

Agenda

1. Configuring Chef workstation
2. Configuring Chef Server
3. Chef Nodes

Chef-server, workstation, node and Chef Tools

Launch 3 ec2 Ubuntu instances, or spin up Vagrant instances.

Connect to using ssh to all three nodes.

- Check OS version by running,
  `$ cat /etc/issue`
- Update hostname information by applying relevant names in /etc/hostname file.
- Update host file in /etc/hosts with IP-add and Hostname info
- Check firewall rule by using command,
  `$ iptables –L`
- Incase if there are firewall rules applied, flush the rules with command,
  `$ iptables –F`

Preparing and Installing Chef-Workstation

A **chef workstation** is a computer that is configured to run Knife, to synchronize with the chef-repo, and interact with a single Chef server.
The workstation is the location, from which most users will do most of their work, including:

- Developing cookbooks and recipes
- Keeping the chef-repo synchronized with version source control
- Using Knife to upload items from the chef-repo to the Chef server
- 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.

The actual infrastructure coordination and configuration does not take place on the Chef server. This work is done on a workstation which then uploads the data to the server to influence the Chef environment.

Download the ‘chefdk’ package from chef.io download location by using command,

```
$wget https://packages.chef.io/stable/ubuntu/12.04/chefdk_0.19.6-1_amd64.deb
```

Install the package using dpkg installer,

Chef-server, workstation, node and Chef Tools

```
$ dpkg –i chefdk_0.19.6-1_amd64.deb
```

The package also installs other apps like as below,

- Chef Development Kit Version: 0.19.6
- chef-client version: 12.15.19
- berks version: 5.1.0
- kitchen version: 1.13.2

Now, try writing a ruby code for the Chef to apply it on the node. We are doing this on the workstation as that's the way it is. The workstation is there for the developer to work on, play with the recipes, test it ion the workstation and then deploy it to the server.

First and foremost task is to create a chef repository.

```
$ chef generate repo <repo name>
```

Once the repository is created jump into the cookbook repository and run the command to generate cookbook.

Create a sample file say, 'hello.rb'. This is called the chef recipe file.

Update the file with,

```
package 'apache2' do
 action  [:install]
end

service 'apache2' do
 action [:enable, :start]
end

file '/var/www/html/index.html' do
 content '<html>
  <body>
<h1>
     Hello World!
</h1>
        </body>
      </html>'
end
```

Ruby Block

Try running the file using command,

```
$ chef-apply <hello.rb>
```

And check the index.html output by running 'curl' command.

In case of package, if we do not specify any attribute, the default action would be to install.

If there multiple recipes that we want to apply on to the node, we need to manage those in order. To do that, we use cookbooks.

```
$ chef generate cookbook <cookbook-name>
```

The centralized repository that's contributed by community has many cookbook ready for use, is available at 'Supermarket.chef.io'

The cookbook can be downloaded by using command,

```
$ knife cookbook site download <cookbook-name>
```

```
$ knife cookbook upload <cookbook-name>
```
----- upload updated cookbook to Chefserver.

Some important commands to be remembered..

```
# knife cookbook create apache
```

```
# cat ~/chef-repo/cookbooks/apache/metadata.rb
name             'apache'
maintainer       'YOUR_COMPANY_NAME'
maintainer_email 'YOUR_EMAIL'
license          'All rights reserved'
description      'Installs/Configures apache'
long_description IO.read(File.join(File.dirname(__FILE__),
'README.md'))
version          '0.1.0'
```

Optional command with some more parameters passed.,

```
# knife cookbook create <cookbook-name> -C "<user-name>" -m "<e-mail
address>" -I gplv3 -r rdoc
```

In the above command,

- C is the copyright holder name. You can also use –copyright option.
- m is the email address. You can also use –email option.
- I is the license type. This will automatically add the appropriate license notice in the files that you create under this cookbook. The following are the valid license types you can use. In this example, we are using gplv3, which will add the GPL v3 license notice in the header. You can also use –license option. ▪apachev2 – This will use Apache v2.0
- gplv2 – This option is for GPL v2

Chef-server, workstation, node and Chef Tools

- gplv3 – For GPL v3
- mit – For MIT

- none – This option will add this notice to the files: “Proprietary – All Rights Reserved”
- r option will define the format for the README files in this cookbook. Use “rdoc” option for Ruby docs, and “md” for markdown. Instead of -r you can also use –readme-format
- o (or –cookbook-path) – This option is used when you want to specify a different directory path where you want the cookbooks to be created. In the above example, we are not using this option.

`# knife cookbook upload -a` -- upload all cookbook at once

`# knife cookbook upload <cookbook-name> -d` -- uploads all dependent cookbooks as well.

`# knife cookbook upload <cookbook-name> --freeze` -- after this command, one can upload any cookbooks

`# knife cookbook upload <cookbook-name> --force` -- in case the cookbook upload is to be forced even after it's frozen.

`# knife cookbook upload -a --concurrency` -- incase the cookbooks are to be uploaded are large in-size, we can upload those concurrently.

`# knife cookbook delete <cookbook-name> -p` -- delete a cookbook permanently.

If there multiple versions of cookbooks uploaded, chef will ask the one to be deleted.
we can also use wild characters to delete cookbooks in multiples.

`# knife cookbook download <cookbook-name>` -- download a cookbook from server.

`# knife cookbook show <cookbook-name>`

Download Chef-client package on Ubuntu workstation using command

```
$ wget https://packages.chef.io/stable/ubuntu/12.04/chef_12.16.42-1_amd64.deb
```

Install it with command,

```
$ dpkg -I chef_12.16.42-1_amd64.deb
```

Check version with command,

```
$ chef-client --version
```

Knife

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

Chef-server, workstation, node and Chef Tools

- Cookbooks and recipes Roles
- Stores of JSON data (data bags), including encrypted data
- Environments
- Cloud resources, including provisioning
- The installation of the chef-client on management workstations.

Configuring knife

We can configure knife by running command knife configure -i on linux machine.
After running this command it will ask for chef URL and key files, assign it accordingly.
Second option is we can download knife configuration directly from chef server and can place it to our machine where we want to configure workstation. This will give you a directory say chef-repo. In this directory a hidden directory named .chef will be there. Inside this directory you would be having one knife.rb file, one user's key having knife authorization and one server-validation key.
You can test the knife by running this command:

```
# knife client list
```

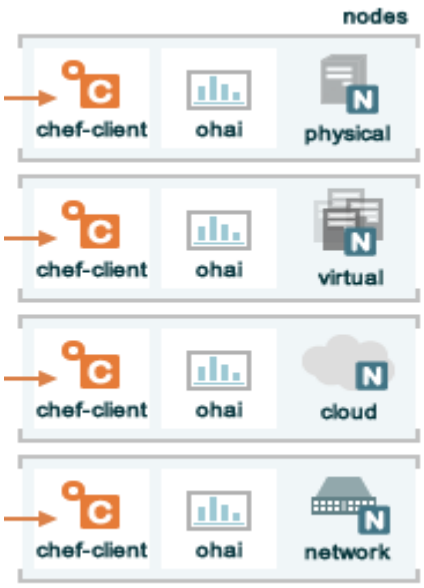
(This will lists all the clients associated with the organization for which knife is configured).

If this works fine. it means your workstation has been configured correctly.

Chef-server, workstation, node and Chef Tools

Node:

- Each node stores its own private key locally.
- This private key is generated as part of the bootstrap process that initially installs the chef-client on the node.
- The first time chef-client runs on that node, it uses the chef-private key, if the client key is not available then the **chef-validator** is used to to authenticate, but then on each subsequent run it uses the private key generated for that client by the Chef server.



Chef-server, workstation, node and Chef Tools

Setup your Chef Server

Either you can opt for hosted chef server or you can setup your own on premises Chef Server.
For Hosted Chef, Just signup to https://manage.chef.io/signup and create organization.

To install Chef Server,

- Run command on chefserver instance to download and install the Chef-server package,
  `$ wget` https://packages.chef.io/files/stable/chef-server/12.17.33/ubuntu/16.04/chef-server-core_12.17.33-1_amd64.deb

  ```
  $ dpkg –i chef-server-core_12.17.33-1_amd64.deb
  ```

- Once the installation is complete, we should run chef-server-ctl reconfigure command to start the Chef Server services. It configures the components that make up the server to work together in our specific environment.
  ```
  $ chef-server-ctl reconfigure
  ```

- Once configured run the commands to check all components are started with related information,
  ```
  $ chef-server-ctl status.
  ```

  Also run a test command,
  ```
  $ chef-server-ctl test
  ```

- Once the Server is configured, up and running, create admin user that will be used to login to the Chef server and the organization.
  A user is created with below information.,
  1. Username: admin
  2. First Name: admin
  3. Last Name: admin
  4. Email: admin@example.com
  5. Password: password
  6. Filename: admin.pem

  Also create Organization using below information.,

  ```
  Short Name:
  Long Name:
  Association User
  Filename: orgname.pem
  ```

- In order to link workstations and nodes to the Chef server, an administrator and an organization need to be created with associated RSA private keys. From the home directory, create a .chef directory to store the keys:

  ```
  $ mkdir .chef
  ```

Chef-server, workstation, node and Chef Tools

- Now create the admin user.
  $ sudo chef-server-ctl user-create <username><firstname><lastname><e-mail><password> <file location>

```
$ sudo chef-server-ctl user-create admin admin admin admin@autofact.com
password -f ~/.chef/admin.pem
```

- With this we now have a private key file located in above path.

- Now let's create organization with the org-create sub command,

```
$ sudo chef-server-ctl org-create organization "org.com" --
association_user admin -f ~/.chef/orgname.pem
```

- Check the certificate files by running ls command.
- To manage the Chef server remotely, we can install a GUI app on the chef-server.

```
$ sudo chef-server-ctl install opscode-manage
$ sudo opscode-manage-ctl reconfigure
$ sudo chef-server-ctl reconfigure
```

- Now we can access the GUI tool using link https://<chef-server IP address>/login

There are other useful plugins.

Opscode Push Jobs plugin

```
$ sudo chef-server-ctl  install opscode-push-job-server
$ sudo opscode-push-job-server-ctl reconfigure
$ sudo chef-server-ctl reconfigure
```

Chef Reporting:

```
$ sudo chef-server-ctl install opscode reporting
$ chef-server-ctl reconfigure
$ opscode-reporitng-ctl reconfigure
```

Analytics Plugin

```
$ sudo chef-server-ctl install opscode-analytics
$ echo 'analytics_fqdn "FQDN"' | sudo tee -a /etc/opscode-
analytics/opscode-analytics.rb
$ sudo opscode-analytics-ctl reconfigure
$ sudo chef-server-ctl reconfigure.
```

Download the starter kit that will contain the complete chef-repo directory. We'll use chef-repo in upcoming steps.

Chef-server, workstation, node and Chef Tools

Further to this copy the server.crt file from /var/opt/opscode/nginx/ca/hostname.crt to ~/chefrepo/.chef/trusted_certs/

Once this is done, then we can run the knife command from chef workstation against the chef server.

Run the **knife ssl check** command on chef workstation.

Register Node with Chef Server

```
$ knife bootstrap <ipaddress or dns-name> --ssh-user <username> --  ssh-password 'password' --sudo --use-sudo-password --node-name    <hostname>    --run-list 'recipe[cookbook-name]'
```

Chef-server, workstation, node and Chef Tools

On windows environment (create Chef-Workstation).

Bootstrapping Chef on Windows systems requires an additional knife plugin, knife-windows This plugin uses WinRM to allow you to call native objects in Windows remotely.

The plugin adds a few subcommands, notably knife bootstrap windows winrm and knife bootstrap windows ssh, as well as custom bootstrap templates designed for Windows.

Once you have installed the knife-windows plugin, you should be able to bootstrap your Windows system using a command similar to:

knife bootstrap windows winrm ipaddress -x Administrator -P 'super_secret_password'