	Infra/DevOps Department	**Process#**	**Chef Infrastructure**
		Revision #	**1.0**
		Implementation Date	04/26/2017
Page #	1 of 11	**Last Reviewed/Update Date**	04/26/2017
Process Owner	Mandeep Kumar	**Approval**	

Chef Infrastructure

1. *Purpose*

This procedure provides a guideline on how to set up Chef Infrastructure .
Chef consists of a Chef server, one or more workstations, and a node where the chef-client is installed. Components names are based on the roles played by each machine in the Chef ecosystem.

Chef Server: This is the central hub server that stores the cookbooks and recipes uploaded from workstations, which is then accessed by chef-client for configuration deployment.
Chef Workstations: This where recipes, cookbooks, and other chef configuration details are created or edited. All these are then pushed to the Chef server from the workstation, where they will be available to deploy to chef-client nodes.
Chef Client: This is the target node where the configurations are deployed in which the chef-client is installed. A node can be any machine (physical, virtual, cloud, network device, etc.)

2. *Scope*

Intended audience is Infrastructure Team and Managers

3. *Prerequisites*

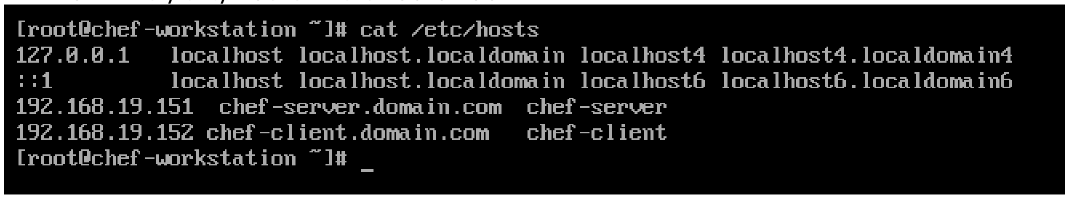
Three RHEL/CENTOS VMs , All three machines should have FQDN resolved by DNS. In test environment, we can modify /etc/hosts file to set FQDN for all three machines. Machines should be able to ping via FQDN.

4. *Responsibilities*

Mandeep Kumar
Mandeep.kumar@boeing.com
Ph.No :206-504-4242

5. *Procedure*

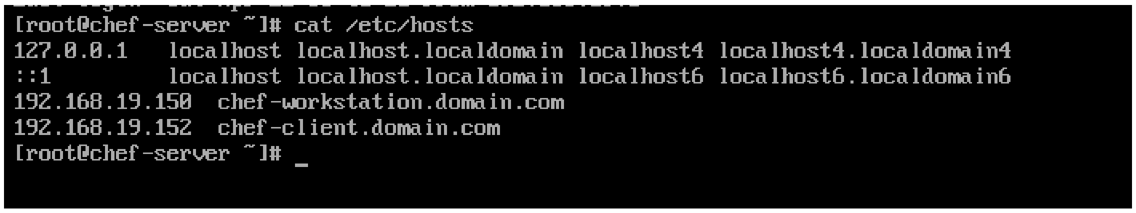
1. Configure three VMs. To configure FQDN as hostname , run the following command "`hostnamectl set-hostname (FQDN of your choice)` "
2. Edit your /etc/hosts file. Below is an example from chef-workstation. That's how the /etc/hosts file should look.



```
[root@chef-workstation ~]# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.19.151  chef-server.domain.com  chef-server
192.168.19.152 chef-client.domain.com   chef-client
[root@chef-workstation ~]# _
```

6. Set Up Chef-server

Sample file from chef-server



```
[root@chef-server ~]# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.19.150  chef-workstation.domain.com
192.168.19.152  chef-client.domain.com
[root@chef-server ~]# _
```

Run The following commands

Installing Chef Server

1. **wget** **https://packages.chef.io/files/stable/chef-server/12.14.0/el/7/chef-server-core-12.14.0-1.el7.x86_64.rpm**

Once the download is complete, install the chef server core using the following command.

2. **rpm -ivh chef-server-core-*.rpm**

If your chef server system does not meet the recommended hardware requirement, this step may fail.

Once the installation is complete, you must reconfigure the chef server components to make up the server to work together. Reconfiguring may take a little bit longer.

3. **chef-server-ctl reconfigure**

Check the status of Chef Server components by using the following command.

4. **chef-server-ctl status**

Output:

```
run: bookshelf: (pid 6084) 387s; run: log: (pid 6114) 385s
run: nginx: (pid 5973) 417s; run: log: (pid 6276) 359s
run: oc_bifrost: (pid 5816) 477s; run: log: (pid 5831) 476s
run: oc_id: (pid 5961) 420s; run: log: (pid 5966) 419s
run: opscode-erchef: (pid 6186) 379s; run: log: (pid 6176) 381s
run: opscode-expander: (pid 6039) 388s; run: log: (pid 6071) 388s
run: opscode-solr4: (pid 5992) 399s; run: log: (pid 5999) 398s
run: postgresql: (pid 5805) 478s; run: log: (pid 5809) 477s
run: rabbitmq: (pid 5767) 480s; run: log: (pid 5760) 481s
run: redis_lb: (pid 5377) 595s; run: log: (pid 6272) 359s
```

Create an Admin user and Organization:

We need to create an admin user. This user will have access to make changes to the infrastructure components in the organization we will be creating. Below command will generate the RSA private key automatically and should be saved to a safe location.

User details are below.
User Name: admin
First Name: admin
Last Name: admin
Email: kumar.26.mandeep@gmail.com
Password: password
File Name: admin.pem
Path: /etc/chef

5. **chef-server-ctl user-create admin admin admin admin@email.com password -f /etc/chef/admin.pem**

Original Command:

chef-server-ctl user-create USER_NAME FIRST_NAME LAST_NAME EMAIL 'PASSWORD' -f PATH_FILE_NAME

It is the time for us to create an organization to hold the chef configurations.

Short Name: Infosys (Note: Name must begin with lowercase letter or digit, may contain lowercase letter, numbers, hyphens, and underscores, and must be between 1 and 255 characters)

Full Organization Name: Infosys (Note: Must begin with non-white space character and must be between 1 and 1023 characters)

Association User: admin (Note: This option will associate the previously created user (admin) with the admins security group on the chef server)

Filename: infosys-validator.pem (Note: command will generate the RSA private key automatically and should be saved to a safe location)
Path: /etc/chef

6. **chef-server-ctl org-create infosys "Infosys" --association_user admin -f /etc/chef/infosys-validator.pem**

Original Command:

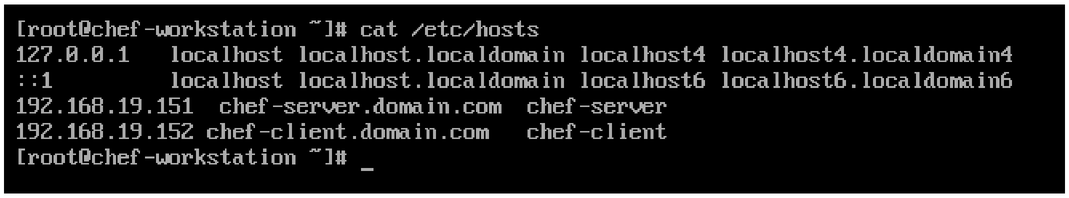
chef-server-ctl org-create short_name 'full_organization_name' --association_user user_name --filename ORGANIZATION-validator.pem
As of now, you will have two .pem keys in /etc/chef directory. In our case, they will be called admin.pem and infosys-validator.pem. Soon we will place these two files in Chef workstation machine.

Firewall:
The Chef server requires the following ports to be open through the firewall. But enabling only 80 and 443 would also do for us.

Run the following command to allow 80 and 443 through the firewall.

7. **firewall-cmd --permanent --zone public --add-service http**
8. **firewall-cmd --permanent --zone public --add-service https**
9. **firewall-cmd --reload**

Installing Chef Workstation



```
[root@chef-workstation ~]# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.19.151  chef-server.domain.com  chef-server
192.168.19.152 chef-client.domain.com   chef-client
[root@chef-workstation ~]# _
```

1. **wget https://packages.chef.io/files/stable/chefdk/1.3.43/el/7/chefdk-1.3.43-1.el7.x86_64.rpm**

Install ChefDK.

2. **rpm -ivh chefdk-*.rpm**

Verify the components of Chef Development Kit.

3. **chef verify**

Output:

```
Running verification for component 'berkshelf'
Running verification for component 'test-kitchen'
Running verification for component 'tk-policyfile-provisioner'
Running verification for component 'chef-client'
Running verification for component 'chef-dk'
Running verification for component 'chef-provisioning'
Running verification for component 'chefspec'
Running verification for component 'generated-cookbooks-pass-chefspec'
Running verification for component 'rubocop'
Running verification for component 'fauxhai'
Running verification for component 'knife-spork'
Running verification for component 'kitchen-vagrant'
Running verification for component 'package installation'
Running verification for component 'openssl'
Running verification for component 'inspec'
Running verification for component 'delivery-cli'
Running verification for component 'git'
Running verification for component 'opscode-pushy-client'
Running verification for component 'chef-sugar'
............................................................
--------------------------------------------
Verification of component 'kitchen-vagrant' succeeded.
Verification of component 'openssl' succeeded.
Verification of component 'delivery-cli' succeeded.
Verification of component 'test-kitchen' succeeded.
Verification of component 'rubocop' succeeded.
Verification of component 'opscode-pushy-client' succeeded.
Verification of component 'berkshelf' succeeded.
Verification of component 'fauxhai' succeeded.
Verification of component 'inspec' succeeded.
Verification of component 'tk-policyfile-provisioner' succeeded.
Verification of component 'chefspec' succeeded.
Verification of component 'knife-spork' succeeded.
Verification of component 'git' succeeded.
Verification of component 'chef-dk' succeeded.
Verification of component 'chef-sugar' succeeded.
Verification of component 'chef-client' succeeded.
Verification of component 'generated-cookbooks-pass-chefspec' succeeded.
Verification of component 'package installation' succeeded.
Verification of component 'chef-provisioning' succeeded.
```

Some of the users may want to set Ruby version default to Ruby version installed with Chef. Check the current Ruby location.

4. **which ruby**

This command will yield you a result if your machine has Ruby installed. Run the below command to load CheDK variables to user profile file.

5. **echo 'eval "$(chef shell-init bash)"' >> ~/.bash_profile**

Load the user profile.

6. **. ~/.bash_profile**

Now, check the Ruby. You should get the similar output.

7. **Which ruby**

```
/opt/chefdk/embedded/bin/ruby
.
```

8. **cd ~**
9. **chef generate repo chef-repo**

This command places the basic chef repo structure into a directory called “**chef-repo**” in your home directory.

10. **ls -al ~/chef-repo/**

Output:

```
total 32
drwxr-xr-x. 8 root root 4096 Nov 12 18:30 .
dr-xr-x---. 5 root root 4096 Nov 12 18:29 ..
-rw-r--r--. 1 root root 1133 Nov 12 18:29 chefignore
```

```
-rw-r--r--. 1 root root  255 Nov 12 18:29 .chef-repo.txt
drwxr-xr-x. 3 root root   36 Nov 12 18:29 cookbooks
drwxr-xr-x. 3 root root   36 Nov 12 18:29 data_bags
drwxr-xr-x. 2 root root   41 Nov 12 18:29 environments
drwxr-xr-x. 7 root root 4096 Nov 12 18:29 .git
-rw-r--r--. 1 root root  106 Nov 12 18:29 .gitignore
-rw-r--r--. 1 root root   70 Nov 12 18:29 LICENSE
-rw-r--r--. 1 root root 1499 Nov 12 18:29 README.md
drwxr-xr-x. 2 root root   41 Nov 12 18:29 roles
```

<u>Add version control:</u>
Go to the **chef-repo** directory and initialize it.

11. **cd ~/chef-repo/**
12. **git init**

Now, let's create a hidden directory called "**.chef**" under the **chef-repo** directory. This hidden directory will hold the RSA keys that we created on the Chef server.

13. **mkdir -p ~/chef-repo/.chef**

Since this hidden directory stores the RSA keys, it should not be exposed to the public. To do that we will add this directory to "**.gitignore**" to prevent uploading the contents to GitHub.

14. **echo '.chef' >> ~/chef-repo/.gitignore**

Add and commit all existing files.

15. **cd ~/chef-repo/**
16. **git add .**
17. **git commit -m "initial commit"**

Check the status of the directory.

18. **git status**

Output:
```
nothing to commit, working directory clean
```

<u>Copy the RSA Keys to the Workstation:</u>

The RSA keys (**.pem**) generated when setting up the Chef Server will now need to be placed on the workstation. Place it under "**~/chef-repo/.chef**" directory.

19. **scp -pr root@chef-server:/etc/chef/admin.pem ~/chef-repo/.chef/**

20. **scp -pr root@chef-server:/etc/chef/infosys-validator.pem ~/chef-repo/.chef/**

Create knife.rb File:

Knife is a command line interface for between a local chef-repo and the Chef server. To make the knife to work with your chef environment, we need to configure it by creating knife.rb in the "**~/chef-repo/.chef/**" directory.
Now, create and edit the knife.rb file using your favorite editor.

21. **vi ~/chef-repo/.chef/knife.rb**

In this file, paste the following information:

```
current_dir = File.dirname(__FILE__)
log_level                :info
log_location             STDOUT
node_name                "admin"
client_key               "#{current_dir}/admin.pem"
validation_client_name   "infosys-validator"
validation_key           "#{current_dir}/infosys-validator.pem"
chef_server_url          "https://chef-server.domain.com/organizations/infosys"
syntax_check_cache_path  "#{ENV['HOME']}/.chef/syntaxcache"
cookbook_path            ["#{current_dir}/../cookbooks"]
```

Adjust the following items to suit for your infrastructure.
node_name: This the username with permission to authenticate to the Chef server. Username should match with the user that we created on the Chef server.
client_key: The location of the file that contains user key that we copied over from the Chef server.
validation_client_name: This should be your organization's **short name** followed by **-validator**.
validation_key: The location of the file that contains validation key that we copied over from the Chef server. This key is used when a chef-client is registered with the Chef server.
chef_server_url: The URL of the Chef server. It should begin with **https://**, followed by **IP address** or **FQDN** of Chef server, organization name at the end just after **/organizations/**.
{current_dir} represents ~/chef-repo/.chef/ directory, assuming that knife.rb file is in ~/chef-repo/.chef/. So you don't have to write the fully qualified path.

Testing Knife:

Now, test the configuration by running knife client list command. Make sure you are in **~/chef-repo/** directory.

22. cd ~/chef-repo/
23. Knife client list

You may get an error like below on your first attempt:
ERROR: SSL Validation failure connecting to host: chef-server.domain.com
SSL_connect returned=1 errno=0 state=error: certificate verify failed
ERROR: Could not establish a secure connection to the server.
Use `knife ssl check` to troubleshoot your SSL configuration.
If your Chef Server uses a self-signed certificate, you can use
`knife ssl fetch` to make knife trust the server's certificates.

Original Exception: OpenSSL::SSL::SSLError: SSL Error connecting to https://chef-server.domain.com/organizations/infosys/clients - SSL_connect returned=1 errno=0 state=error: certificate verify failed
To resolve this issue, we need to fetch the Chef server's SSL certificate on our workstation beforehand running the above command.

24.knife ssl fetch

This command will add the Chef server's certificate file to trusted certificate directory.
WARNING: Certificates from chef-server.domain.com will be fetched and placed in your trusted_cert
directory (/root/chef-repo/.chef/trusted_certs).

Knife has no means to verify these are the correct certificates. You should
verify the authenticity of these certificates after downloading.

Adding certificate for chef-server.domain.comin /root/chef-repo/.chef/trusted_certs/chef-server_infosys local.crt
Once the SSL certificate has been fetched, run the previous command to test the knife configuration.

25.knife client list

Output:
infosys-validator
The output confirms the verification has been completed successfully.

Bootstrapping a Node

Bootstrapping a node is a process of installing chef-client on a target machine so that it can run as a chef-client node and communicate with the chef server.
From the workstation, you can bootstrap the node either by using the node's root user, or a user with elevated privileges.

1. **knife bootstrap chef-client.domain.com -x root -P pass --sudo**

Important options:
-x: The ssh username
-P: The ssh password
-p: The ssh port
-N: Set your chef-client node name. Leaving this out will usually make hostname being used as the chef-client node name.
–sudo: If the user name on the node will need to use sudo to perform administrative actions, then use this flag. Note: It will prompt you for sudo the sudo password.
Since I didn't use -N in the command, the hostname will become chef node name.
Output:

```
Doing old-style registration with the validation key at /root/chef-repo/.chef/infosys-validator.pem...
Delete your validation key in order to use your user credentials instead

command.
```

2. **knife node list**

Output:

```
chef-client.domain.com
```

Get the client node details.

3. **knife client show chef-client.domain.com**

Output:

```
admin:     false
chef_type: client
name:      chef-client.domain.com
validator: false
```

Alternative method of bootstrapping a node. Ssh public key should be present in CHEF-CLIENT(node)

```
knife bootstrap chef-client.domain.com   --ssh-user  root --sudo --identity-file
~/.ssh/private_key
```

Creating and uploading a cookbook:

To Generate a cook book :

```
chef generate cookbook cookbook_name
knife cookbook upload cookbook_name
knife node run_list add NODE_NAME "recipe[]"
```

Applying cookbook:

Run chef-client on the node for which we have added the run list