

Chef Demo

A non-trivial demo of chef should at least include 3 components.

1. A chef server – the central controller
2. A node – The actual machine where you intend to make a change
3. A workstation – The dev or build box from where one can administer and effect changes.

We need to create two new VMs which will act as Chef Server and Node. The Workstation VM which we have been using till now will continue to be used as the 3rd component in the above list. This demo will require you to switch between these three VMs and be mindful of on what VM a particular command should be run. The juggling, though seems to be a bit overwhelming in the beginning, can be eased by visualizing

- a. The “Workstation” - as a build machine or the machine of an Ops Engineer
- b. The “server” - as a central component which will hold and help to distribute the changes on a multiple of nodes.
- c. The “Node” – Can be multiple, even web scale servers which happen to be our actual production environment on a server-farm. In this case it’s a single VM.

1. Create “Chef-server”

Step 1: Vagrant VM for chef server (Execute on your Host machine, in this case its Windows)

- a. Create a directory called “chef-server” and copy the contents from “devops\base\chef-server” to the chef-server. In the case below the directory is created at “D:\demo\chef-server”
- b. On the command line change directory to D:\demo\chef-server

```
D:\demo\chef-server>dir
Volume in drive D is New Volume
Volume Serial Number is 68FD-D30F

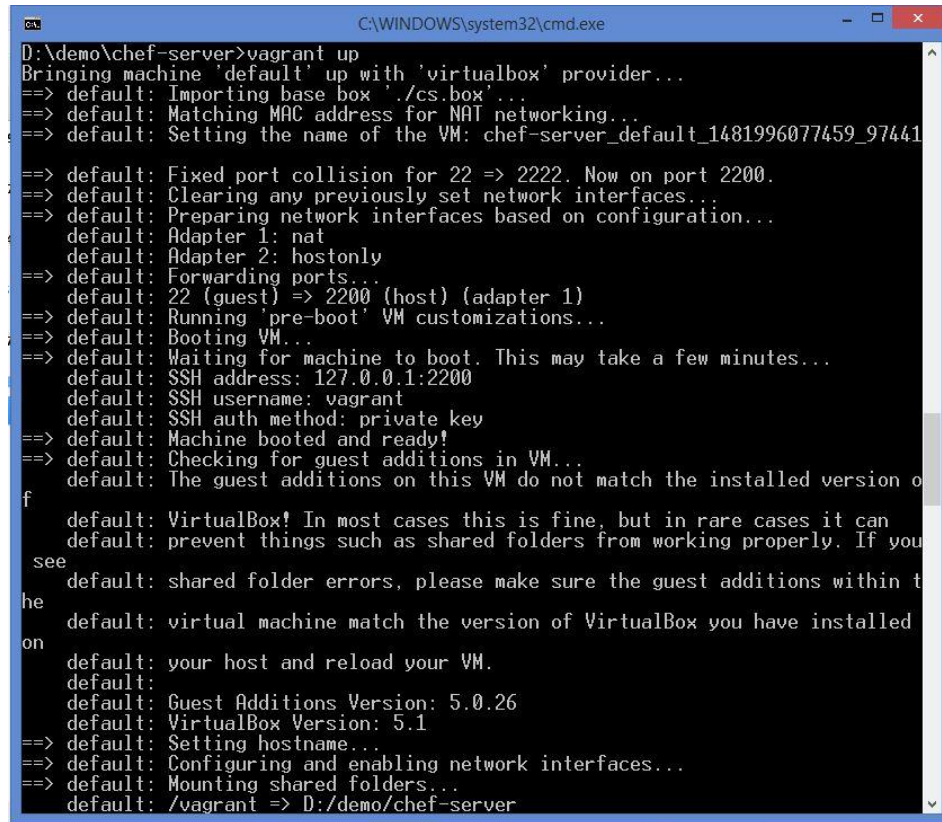
Directory of D:\demo\chef-server

12/17/2016  11:01 PM    <DIR>          .
12/17/2016  11:01 PM    <DIR>          ..
12/17/2016  03:41 PM      2,253,246,880 cs.box
12/17/2016  02:23 PM    <DIR>          secrets
12/17/2016  02:28 PM           3,466 Vagrantfile
                2 File(s)  2,253,250,346 bytes
                3 Dir(s)  144,905,834,496 bytes free

D:\demo\chef-server>
```

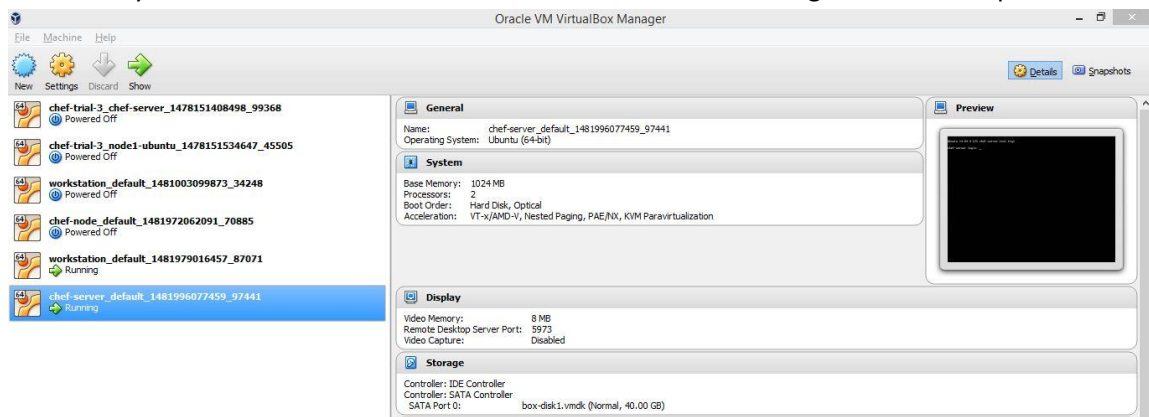
- c. Execute “vagrant up” (Make sure Virtual box is up and running). Your output may vary slightly

```
vagrant up
```



```
D:\demo\chef-server>vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Importing base box './cs.box'...
==> default: Matching MAC address for NAT networking...
==> default: Setting the name of the VM: chef-server_default_1481996077459_97441
==> default: Fixed port collision for 22 => 2222. Now on port 2200.
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
default: Adapter 1: nat
default: Adapter 2: hostonly
==> default: Forwarding ports...
default: 22 (guest) => 2200 (host) (adapter 1)
==> default: Running 'pre-boot' VM customizations...
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
default: SSH address: 127.0.0.1:2200
default: SSH username: vagrant
default: SSH auth method: private key
==> default: Machine booted and ready!
==> default: Checking for guest additions in VM...
default: The guest additions on this VM do not match the installed version of
f
default: VirtualBox! In most cases this is fine, but in rare cases it can
default: prevent things such as shared folders from working properly. If you
see
default: shared folder errors, please make sure the guest additions within t
he
default: virtual machine match the version of VirtualBox you have installed
on
default: your host and reload your VM.
default:
default: Guest Additions Version: 5.0.26
default: VirtualBox Version: 5.1
==> default: Setting hostname...
==> default: Configuring and enabling network interfaces...
==> default: Mounting shared folders...
default: /vagrant => D:/demo/chef-server
```

- d. A new entry should be made in Virtual Box. Click on show for the vagrant to come up



- e. Login to the chef-server VM using the below credentials
Username: vagrant
Password: vagrant

Your chef server is now up- Keep it running!

2. Configure “workstation” to interact with “chef-server”

Step 2: Log in to your workstation VM

Step 3: Make a folder name “chefdemo” under the home directory

```
cd
mkdir chefdemo
cd chefdemo
```

```
vagrant@precise64:~$ pwd
/home/vagrant
vagrant@precise64:~$ mkdir chefdemo
vagrant@precise64:~$ cd chefdemo
vagrant@precise64:~/chefdemo$
```

Step 4: Change directory to chefdemo and create a directory name “.chef” (NB : the “.” is mandatory along with the name “chef”)

```
mkdir .chef
```

Step 5: Copy admin.pem and knife.rb into .chef directory created above

```
cd .chef
cp /vagrant/admin.pem .
cp /vagrant/knife.rb .
ls -ltra
```

```
vagrant@precise64:~/chefdemo$ cd .chef
vagrant@precise64:~/chefdemo/.chef$ cp /vagrant/admin.pem .
vagrant@precise64:~/chefdemo/.chef$ cp /vagrant/knife.rb .
vagrant@precise64:~/chefdemo/.chef$ ls -ltra
total 16
drwxrwxr-x 3 vagrant vagrant 4096 Dec 17 17:43 .
drwxrwxr-x 2 vagrant vagrant 4096 Dec 17 17:47 ..
-rwxrwxr-x 1 vagrant vagrant 1674 Dec 17 17:48 admin.pem
-rwxrwxr-x 1 vagrant vagrant 432 Dec 17 17:48 knife.rb
vagrant@precise64:~/chefdemo/.chef$ _
```

Step 6: Make an entry to /etc/hosts. In this case an entry is already made. Check the same in /etc/hosts. Check for the entry “10.1.1.33 chef-server.test”

```
cat /etc/hosts
```

```
vagrant@precise64:~/chefdemo/.chef$ cat /etc/hosts
127.0.0.1    localhost
127.0.1.1    precise64

# The following lines are desirable for IPv6 capable hosts
::1         ip6-localhost ip6-loopback
fe00::0     ip6-localnet
ff00::0     ip6-mcastprefix
ff02::1     ip6-allnodes
ff02::2     ip6-allrouters
10.1.1.33   chef-server.test
vagrant@precise64:~/chefdemo/.chef$ _
```

Step 7: Change directory to “/home/vagrant/chefdemo” and execute “knife ssl fetch” and “knife ssl check”

```
cd /home/vagrant/chefdemo
knife ssl fetch
knife ssl check
```

```
vagrant@precise64:~/chefdemo/.chef$ cd ..
vagrant@precise64:~/chefdemo$ knife ssl fetch
WARNING: Certificates from chef-server.test will be fetched and placed in your t
trusted_cert
directory (/home/vagrant/chefdemo/.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_test in /home/vagrant/chefdemo/.chef/trusted_
certs/chef-server_test.crt
vagrant@precise64:~/chefdemo$ knife ssl check
Connecting to host chef-server.test:443
Successfully verified certificates from `chef-server.test'
vagrant@precise64:~/chefdemo$
```

3. Upload a cook book

Step 8: Create a directory called “cookbooks” under the directory “chefdemo”

Step 9: Our setup comes with a predefined cookbook called devopsdemo. This cookbook will help us build the complete CD pipeline. Copy the cookbook devopsdemo to cookbooks

```
mkdir cookbooks
cd cookbooks
cp -r /vagrant/devopsdemo .
```

```

vagrant@precise64:~/chefdemo$ mkdir cookbooks
vagrant@precise64:~/chefdemo$ cd cookbooks
vagrant@precise64:~/chefdemo/cookbooks$ cp -r /vagrant/devopsdemo .
vagrant@precise64:~/chefdemo/cookbooks$ ls -ltra
total 12
drwxrwxr-x 4 vagrant vagrant 4096 Dec 17 17:53 ..
drwxrwxr-x 3 vagrant vagrant 4096 Dec 17 17:53 .
drwxrwxr-x 8 vagrant vagrant 4096 Dec 17 17:53 devopsdemo
vagrant@precise64:~/chefdemo/cookbooks$ cd devopsdemo
vagrant@precise64:~/chefdemo/cookbooks/devopsdemo$ ls -ltra
total 56
drwxrwxr-x 3 vagrant vagrant 4096 Dec 17 17:53 ..
drwxrwxr-x 3 vagrant vagrant 4096 Dec 17 17:53 .delivery
drwxrwxr-x 8 vagrant vagrant 4096 Dec 17 17:53 .git
-rwxrwxr-x 1 vagrant vagrant 481 Dec 17 17:53 .kitchen.yml
-rwxrwxr-x 1 vagrant vagrant 185 Dec 17 17:53 .gitignore
-rwxrwxr-x 1 vagrant vagrant 1133 Dec 17 17:53 cheffignore
-rwxrwxr-x 1 vagrant vagrant 47 Dec 17 17:53 Berksfile
drwxrwxr-x 2 vagrant vagrant 4096 Dec 17 17:53 files
-rwxrwxr-x 1 vagrant vagrant 593 Dec 17 17:53 metadata.rb
drwxrwxr-x 2 vagrant vagrant 4096 Dec 17 17:53 recipes
-rwxrwxr-x 1 vagrant vagrant 58 Dec 17 17:53 README.md
drwxrwxr-x 3 vagrant vagrant 4096 Dec 17 17:53 spec
drwxrwxr-x 3 vagrant vagrant 4096 Dec 17 17:53 test
drwxrwxr-x 8 vagrant vagrant 4096 Dec 17 17:53 .
vagrant@precise64:~/chefdemo/cookbooks/devopsdemo$

```

Step 10: Upload cookbook “devopsdemo”. The below command will upload the cookbook to the “chef-server”. Knife requires a “.chef” directory to run and also to know where the server is located. Like how git looks for .git folder in the current directory path, knife too takes the same approach. Hence you can run a knife command anywhere within the “chefdemo” directory in our case.

```
knife cookbook upload devopsdemo
```

```

vagrant@precise64:~/chefdemo/cookbooks/devopsdemo/files$ knife cookbook upload d
evopsdemo
Uploading devopsdemo      [0.1.0]
Uploaded 1 cookbook.
vagrant@precise64:~/chefdemo/cookbooks/devopsdemo/files$

```


4. Create a Node

Step 11: Create a directory called “chef-node” and copy the contents from “devops\base\chef-node” to the “chef-node”. In the case below the directory is created at “D:\demo\chef-node”

```
D:\demo\chef-node>dir
Volume in drive D is New Volume
Volume Serial Number is 68FD-D30F

Directory of D:\demo\chef-node

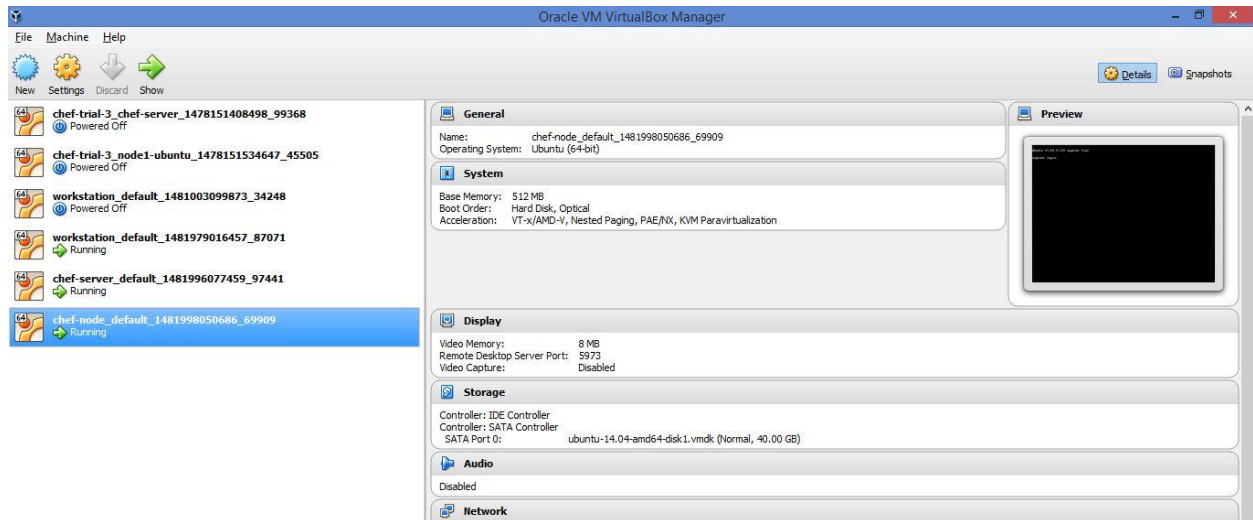
12/17/2016  11:35 PM    <DIR>          .
12/17/2016  11:35 PM    <DIR>          ..
12/17/2016  04:09 PM       1,469,269,565  cn.box
12/17/2016  04:13 PM           3,412  Vagrantfile
                2 File(s)  1,469,272,977 bytes
                2 Dir(s)  144,905,842,688 bytes free

D:\demo\chef-node>
```

Execute vagrant up (Make sure Virtual box is up and running). Your output may vary slightly

```
D:\demo\chef-node>vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Importing base box 'ubuntu-14.04.box'...
==> default: Matching MAC address for NAT networking...
==> default: Setting the name of the VM: chef-node_default_1481998050686_69909
==> default: Fixed port collision for 22 => 2222. Now on port 2201.
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
default: Adapter 1: nat
default: Adapter 2: hostonly
==> default: Forwarding ports...
default: 22 (guest) => 2201 (host) (adapter 1)
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
default: SSH address: 127.0.0.1:2201
default: SSH username: vagrant
default: SSH auth method: private key
default: Warning: Remote connection disconnect. Retrying...
default:
default: Vagrant insecure key detected. Vagrant will automatically replace
default: this with a newly generated keypair for better security.
default:
default: Inserting generated public key within guest...
default: Removing insecure key from the guest if it's present...
default: Key inserted! Disconnecting and reconnecting using new SSH key...
==> default: Machine booted and ready!
==> default: Checking for guest additions in VM...
default: The guest additions on this VM do not match the installed version of
f
```

Step 12: A new entry should be made in Virtual Box. Click on “show” for the vagrant to come up



Step 13: Login to the chef-node VM using the below credentials

Username : vagrant

Password : vagrant

Step 14: Make entry to /etc/hosts of node. A node needs to know the address of the server and hence we need to add “chef-server.test” as a part of /etc/hosts file

```
echo "10.1.1.33 chef-server.test" | sudo tee -a /etc/hosts
```

```
vagrant@node1-ubuntu:~$ echo "10.1.1.33 chef-server.test" | sudo tee -a /etc/hosts
10.1.1.33 chef-server.test
vagrant@node1-ubuntu:~$ _
```

5. Bootstrap the above created Node

Step 15: On your host machine (windows) go to D:/demo/chef-node and execute “vagrant ssh-config”.

Note the directory of private key

```
D:\demo\chef-node>vagrant ssh-config
Host default
  HostName 127.0.0.1
  User vagrant
  Port 2222
  UserKnownHostsFile /dev/null
  StrictHostKeyChecking no
  PasswordAuthentication no
  IdentityFile "C:/Users/Dell lap/.vagrant.d/boxes/.-VAGRANTSLASH-cn1.box/0/virtualbox/vagrant_private_key"
  IdentitiesOnly yes
  LogLevel FATAL
D:\demo\chef-node>
```

Step 16: On your windows machine Copy the private key from the above address to
D:\demo\workstation

Step 17: On your workstation VM execute the following to bootstrap the node node1-ubuntu(Note the knife command below is a one single command with lots of options!). The knife command has to be executed from “chefdemo” directory itself as “chefdemo” contains the directory .chef which has the connection details to “chef-server”

```
cd /home/vagrant/chefdemo

knife bootstrap 10.1.1.34 --ssh-user vagrant --sudo --identity-file
/vagrant/vagrant_private_key --node-name node1-ubuntu
```

```
vagrant@precise64:~/chefdemo$ knife bootstrap 10.1.1.34 --ssh-user vagrant --sud
o --identity-file /vagrant/vagrant_private_key --node-name node1-ubuntu
Node node1-ubuntu exists, overwrite it? (Y/N) Y
Client node1-ubuntu exists, overwrite it? (Y/N) Y
Creating new client for node1-ubuntu
Creating new node for node1-ubuntu
Connecting to 10.1.1.34
10.1.1.34 bash: warning: setlocale: LC_ALL: cannot change locale (en_US)
10.1.1.34 -----> Existing Chef installation detected
10.1.1.34 Starting the first Chef Client run...
10.1.1.34 Starting Chef Client, version 12.17.44
10.1.1.34 resolving cookbooks for run list: []
10.1.1.34 Synchronizing Cookbooks:
10.1.1.34 Installing Cookbook Gems:
10.1.1.34 Compiling Cookbooks...
10.1.1.34 [2016-12-18T05:02:33+00:00] WARN: Node node1-ubuntu has an empty run l
ist.
10.1.1.34 Converging 0 resources
10.1.1.34
10.1.1.34 Running handlers:
10.1.1.34 Running handlers complete
10.1.1.34 Chef Client finished, 0/0 resources updated in 03 seconds
vagrant@precise64:~/chefdemo$
```

Step 18: Show node

```
knife node show node1-ubuntu
```

```
vagrant@precise64:~/chefdemo$ knife node show node1-ubuntu
Node Name:   node1-ubuntu
Environment: _default
FQDN:        node1-ubuntu
IP:          10.0.2.15
Run List:
Roles:
Recipes:
Platform:    ubuntu 14.04
Tags:
vagrant@precise64:~/chefdemo$
```


Step 19: knife client list

```
knife client list
```

```
vagrant@precise64:~/chefdemo$ knife client list
devopsdemo-validator
node1-ubuntu
vagrant@precise64:~/chefdemo$
```

Step 20: Attach the cookbook as run_list of the node that we have created.

```
knife node run_list set node1-ubuntu 'recipe[devopsdemo]'
```

```
vagrant@precise64:~/chefdemo$ knife node run_list set node1-ubuntu 'recipe[devopsdemo]'
node1-ubuntu:
  run_list: recipe[devopsdemo]
vagrant@precise64:~/chefdemo$
```

Step 21: run chef-client on the node. The below is a single command with a lot of options. Please make sure to run this as a single command itself

```
knife ssh 10.1.1.34 'sudo chef-client' --manual-list --ssh-user vagrant --identity-file /vagrant/vagrant_private_key
```

Note: The above command though will run will show an error-ed exit code. If you look at the screen shot of output below you will notice that the command is looking for a file called MyAppDemo.war. “.war” file will be generated by the Jenkins build system. In the next few steps after we introduce CD through Jenkins, this step will move to a success. Infact Jenkins will use the same command in its pipeline to make a chef-client run

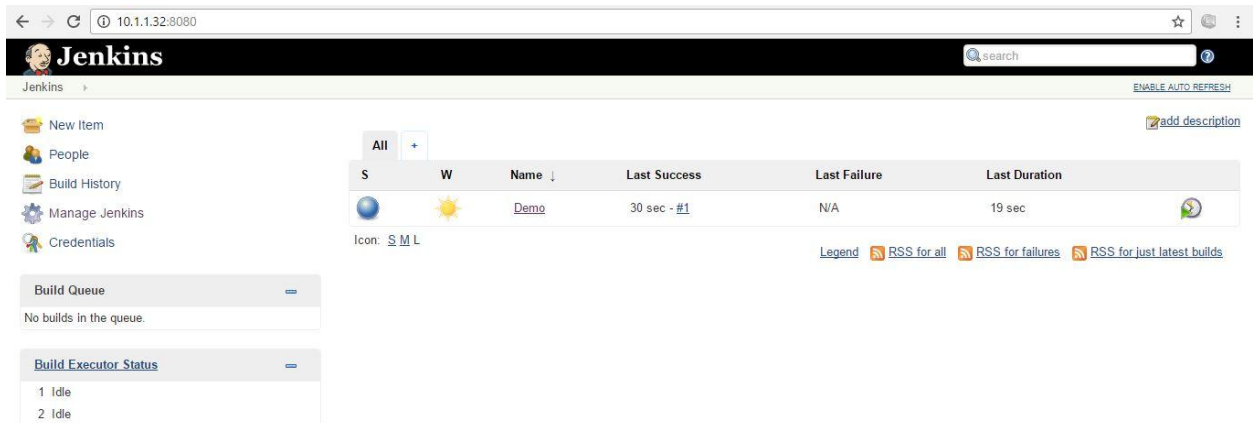
```
10.1.1.34 [2016-12-18T12:23:46+00:00] ERROR: Running exception handlers
10.1.1.34 Running handlers complete
10.1.1.34 [2016-12-18T12:23:46+00:00] ERROR: Exception handlers complete
10.1.1.34 Chef Client failed. 0 resources updated in 02 seconds
10.1.1.34 [2016-12-18T12:23:46+00:00] FATAL: Stacktrace dumped to /var/chef/cache/chef-stacktrace.out
10.1.1.34 [2016-12-18T12:23:46+00:00] FATAL: Please provide the contents of the stacktrace.out file if you file a bug report
10.1.1.34 [2016-12-18T12:23:46+00:00] ERROR: cookbook_file[/var/lib/tomcat7/webapps/MyAppDemo.war] (devopsdemo::default line 7) had an error: Chef::Exceptions::FileNotFound: Cookbook 'devopsdemo' (0.1.0) does not contain a file at any of these locations:
10.1.1.34   files/ubuntu-14.04/MyAppDemo.war
10.1.1.34   files/ubuntu/MyAppDemo.war
10.1.1.34   files/default/MyAppDemo.war
10.1.1.34   files/MyAppDemo.war
10.1.1.34 [2016-12-18T12:23:46+00:00] FATAL: Chef::Exceptions::ChildConvergeError: Chef run process exited unsuccessfully (exit code 1)
vagrant@precise64:~/chefdemo$
```

6. Set up CD

Continuous Delivery

Step 22: Open the Jenkins dashboard by opening the below url on browser

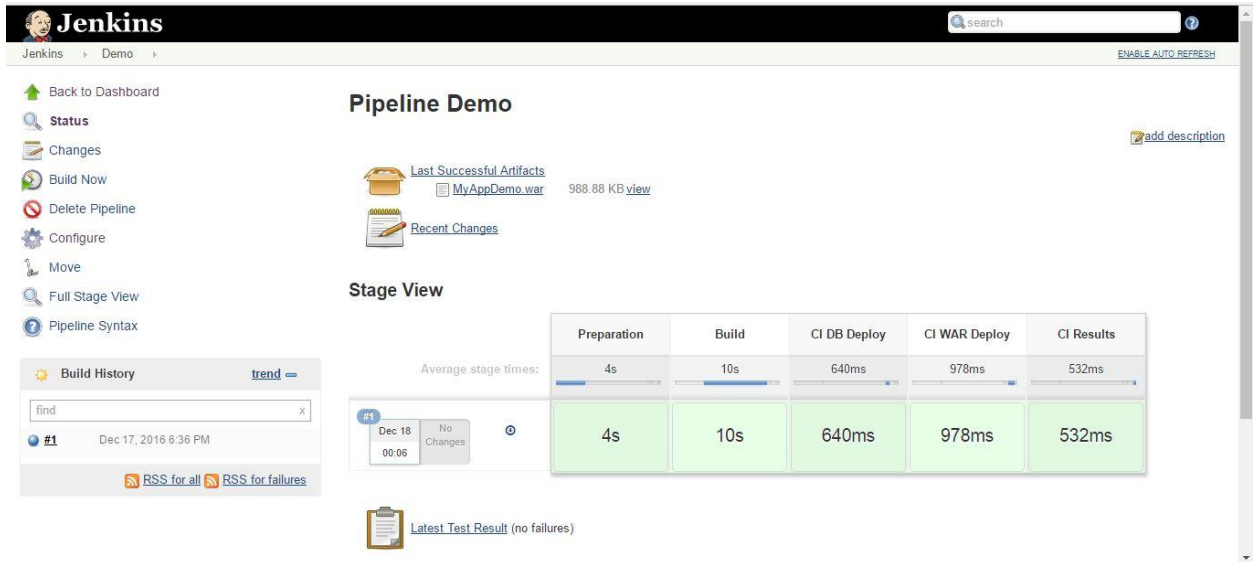
10.1.1.32:8080



The screenshot shows the Jenkins dashboard at 10.1.1.32:8080. The left sidebar contains links for 'New Item', 'People', 'Build History', 'Manage Jenkins', and 'Credentials'. The main area displays a table of builds for the 'Demo' job. The table has columns for 'S' (Status), 'W' (Icon), 'Name', 'Last Success', 'Last Failure', and 'Last Duration'. A single build is listed with a status of 'Success' (blue sphere icon) and a duration of '19 sec'. Below the table, there are sections for 'Build Queue' (showing 'No builds in the queue') and 'Build Executor Status' (showing '1 Idle' and '2 Idle' executors).

S	W	Name	Last Success	Last Failure	Last Duration
Success	[Icon]	Demo	30 sec - #1	N/A	19 sec

Step 23: Click on Demo. This will open the pipeline dashboard



The screenshot shows the Jenkins Pipeline dashboard for the 'Demo' job. The left sidebar includes links like 'Back to Dashboard', 'Status', 'Changes', 'Build Now', 'Delete Pipeline', 'Configure', 'Move', 'Full Stage View', and 'Pipeline Syntax'. The main area is titled 'Pipeline Demo' and shows 'Last Successful Artifacts' (MyAppDemo.war, 988.88 KB) and 'Recent Changes'. Below this is the 'Stage View' section, which displays a table of stage durations. The table has columns for 'Preparation', 'Build', 'CI DB Deploy', 'CI WAR Deploy', and 'CI Results'. The durations are: Preparation (4s), Build (10s), CI DB Deploy (640ms), CI WAR Deploy (978ms), and CI Results (532ms). A 'Build History' section on the left shows a build from Dec 17, 2016, at 6:36 PM.

Stage	Preparation	Build	CI DB Deploy	CI WAR Deploy	CI Results
Average stage times:	4s	10s	640ms	978ms	532ms
#1	4s	10s	640ms	978ms	532ms

Step 24: Click on configure and scroll down to Pipeline

Step 25: Replace the existing script content with the contents of the file "devops\jenkins-scripts\ci-cd.pipeline.txt"

Step 26: On line number 4 of script, provide the right URL of your git code base. In this case it is “https://github.com/AdityaSP/MyAppDemo”

Step 27: Click on build now. Keep your fingers crossed. If everything goes Jenkins Pipeline should a success message!

