Test-driving Your Rails Infrastructure with Chef

All Things Open - 2015

https://github.com/nathenharvey/ato-2015

CHEF

Nathen Harvey

- VP, Community Development at Chef
- Co-host of the Food Fight Show
- Co-organizer of DevOpsDC meetup
- Occasional farmer – http://bit.ly/farmer-nathen
- Love Eggs – http://eggs.chef.io

- @nathenharvey
- nharvey@chef.io

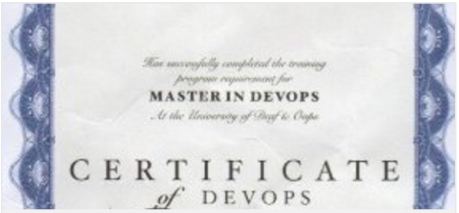
CHEF

Nathen Harvey

- VP, Community Development at Chef
- Co-host of the Food Fight Show
- Co-organizer of DevOpsDC meetup
- Occasional farmer – http://bit.ly/farmer-nathen
- Love Eggs – http://eggs.chef.io

- @nathenharvey
- nharvey@chef.io

Master in DevOps
DevOps
October 2014 – Present (1 year 1 month)



Certificate of Devops

CHEF

Hello!

- System Administrator?

Hello!

- System Administrator?
- Developer?

Hello!

- System Administrator?
- Developer?
- DevOp?

Hello!

- System Administrator?
- Developer?
- DevOp?
- Business Person?

Are you experienced?

- Experience with Infrastructure as Code or Configuration Management?

Are you experienced?

- Experience with Infrastructure as Code or Configuration Management?
- Experience with Chef?

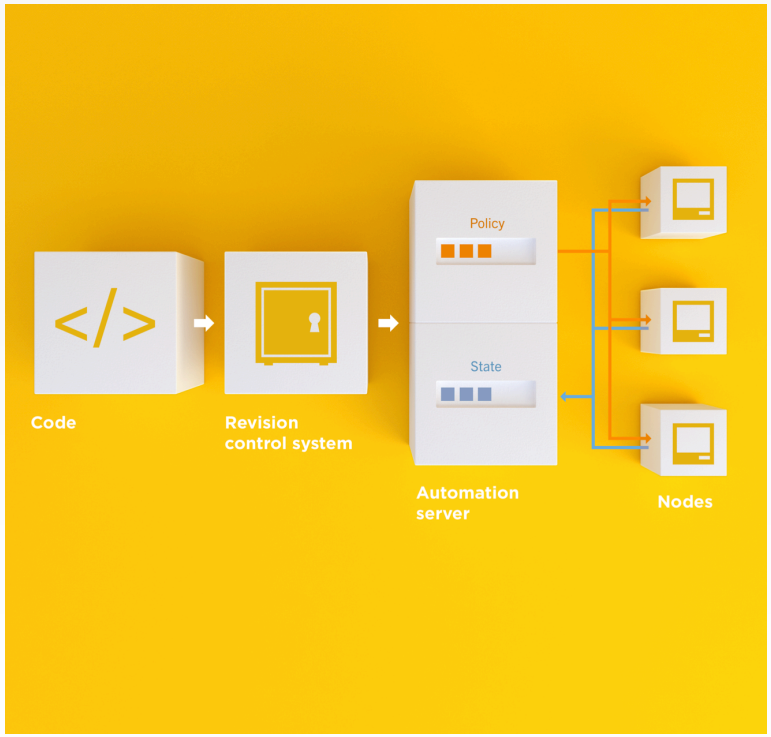
Are you experienced?

- Experience with Infrastructure as Code or Configuration Management?
- Experience with Chef?
- Experience writing automated tests?

Infrastructure as Code

CHEF

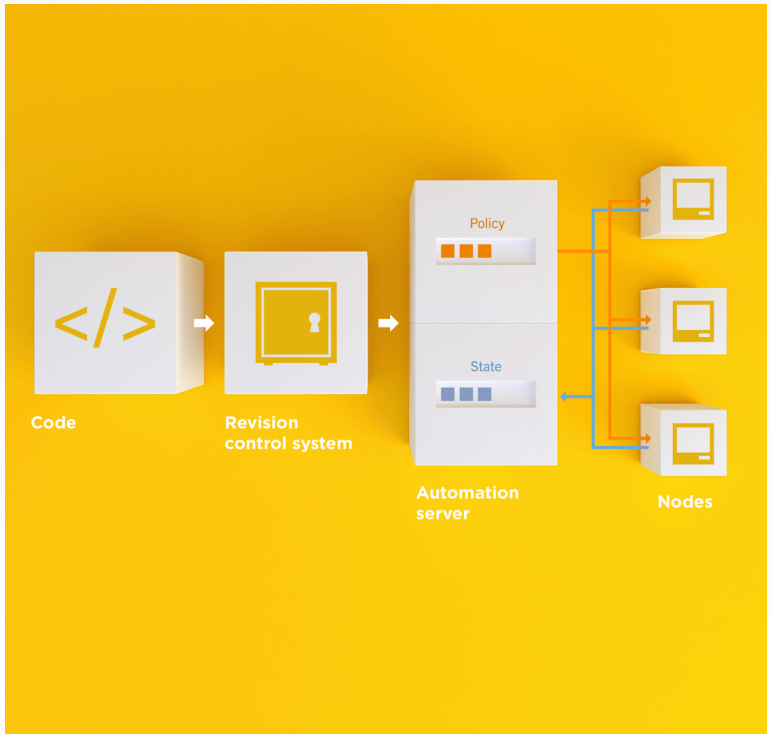
Infrastructure as Code



- Programmatically provision and configure components

CHEF

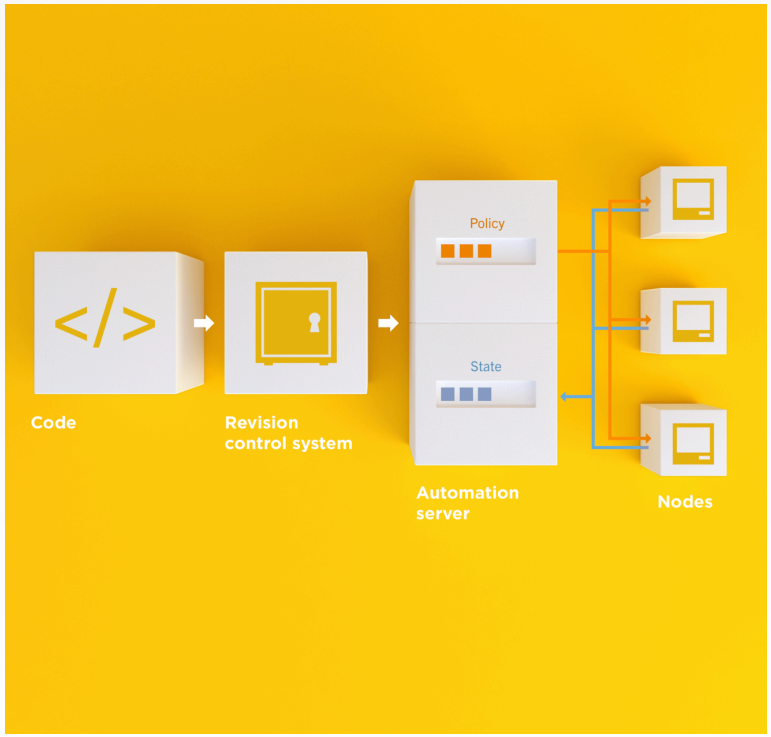
Infrastructure as Code



- Programmatically provision and configure components
- Treat like any other code base

CHEF

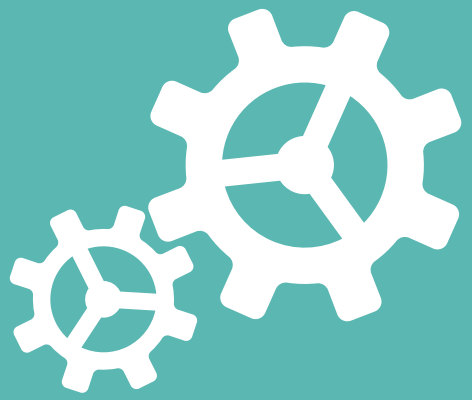
Infrastructure as Code



- Programmatically provision and configure components
- Treat like any other code base
- Reconstruct business from code repository, data backup, and compute resources

CHEF

Automation

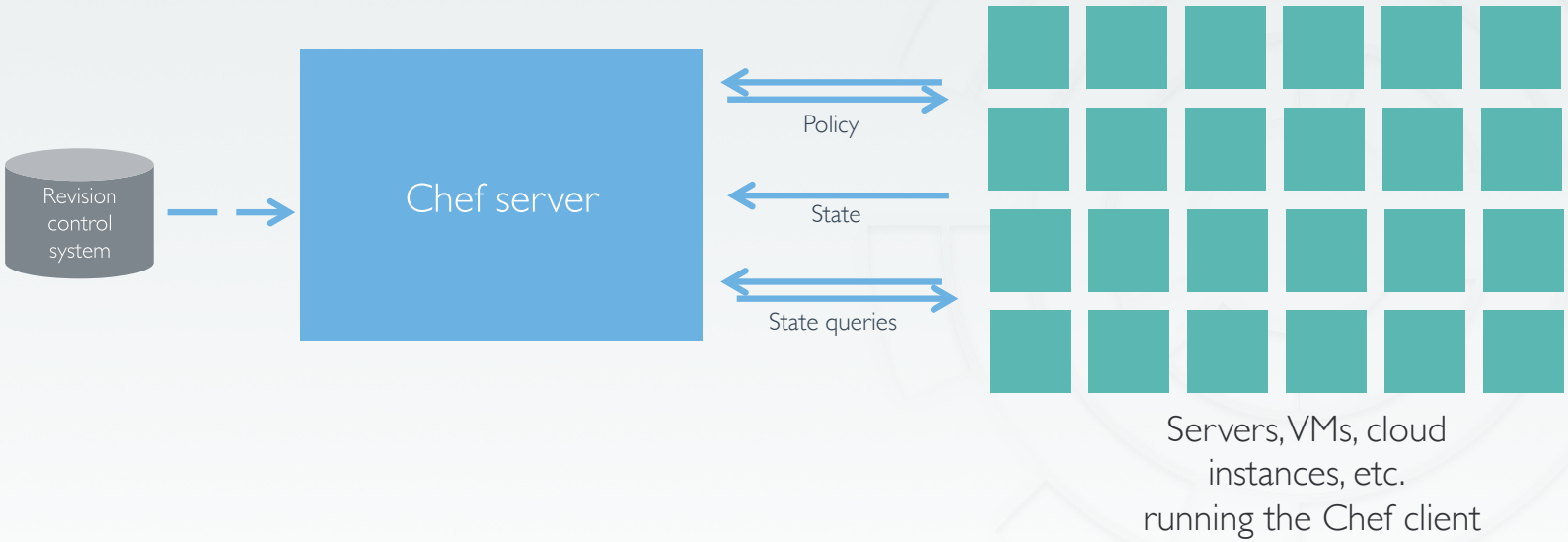


Turn infrastructure into code—infrastructure as code is versionable, testable and repeatable. Manual processes become a thing of the past.

- Automated, full-stack application policies
- Package and service installation
- Versionable, testable, repeatable workflow
- Scalable application policies
- Management of interdependencies across nodes

CHEF

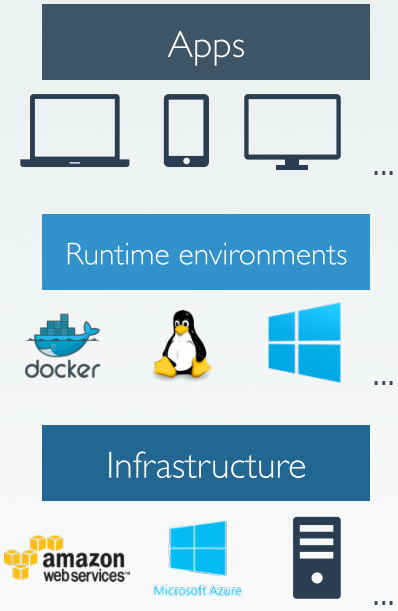
Chef Server



- The Chef server stores policy and configuration data
- The Chef client periodically runs on each node in the network
- Chef clients poll the server for the latest policies
- Chef clients notify the server of their states and can query for the states of other nodes

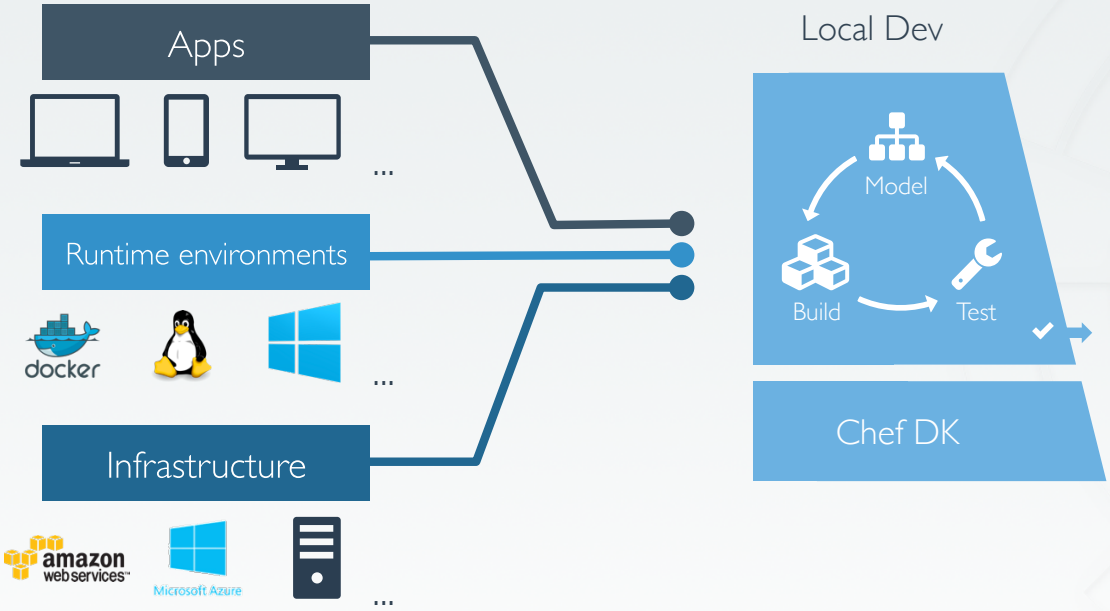
CHEF

Automate Infrastructure and Applications



CHEF

Automate Infrastructure and Applications



CHEF

Describe Infrastructure as Code

```
httpd_service 'customers' do
  mpm 'prefork'
  action [:create, :start]
end

httpd_config 'customers' do
  instance 'customers'
  source 'customers.conf.erb'
  notifies :restart, 'httpd_service[customers]'
end

directory '/var/www/customers/public_html' do
  recursive true
end
```

Test the Code

```
describe 'apache::default' do
  context 'When all attributes are default, on an unspecified platform' do

    let(:chef_run) do
      runner = ChefSpec::ServerRunner.new
      runner.converge(described_recipe)
    end

    it 'converges successfully' do
      expect { chef_run }.to_not raise_error
    end

    it 'installs apache' do
      expect(chef_run).to install_package 'apache2'
    end
  end
end
```

Building your policy

Resources and Recipes

CHEF

Resources

- Piece of the system and its desired state

Resources - Package

Package that should be installed

```
package "mysql-server" do
  action :install
end
```

Resources - Service

Service that should be running and restarted on reboot

```
service "iptables" do
  action [ :start, :enable ]
end
```

Resources - Service

File that should be generated

```
file "/etc/motd" do
  content "Property of Chef Software"
end
```

Resources - Cron

Cron job that should be configured

```
cron "restart webserver" do
  hour '2'
  minute '0'
  command 'service httpd restart'
end
```

Resources - User

User that should be managed

```
user "nginx" do
  comment "Nginx user <nginx@example.com>"
  uid 500
  gid 500
  supports :manage_home => true
end
```

Resources - DSC

DSC resource that should be run

```
dsc_script 'emacs' do
  code <<-EOH
  Environment 'texteditor'
  {
    Name = 'EDITOR'
    Value = 'c:\\emacs\\bin\\emacs.exe'
  }
  EOH
end
```

Resources – Registry Key

Registry key that should be created

```
registry_key "HKEY_LOCAL_MACHINE\
\SOFTWARE\\Microsoft\\Windows\
\CurrentVersion\\Policies\\System"
do
  values [{
    :name => "EnableLUA",
    :type => :dword,
    :data => 0
    }]
  action :create
end
```

Resources

- Piece of the system and its desired state

- http://docs.chef.io/chef/resources.html

Test and Repair

Resources follow a test and repair model

```
package "vim"
```

Test and Repair

Resources follow a **test** and repair model

```
package "vim"
```

Test Is vim installed?

CHEF

Test and Repair

Resources follow a **test** and repair model

```
package "vim"
```

Test Is vim installed?

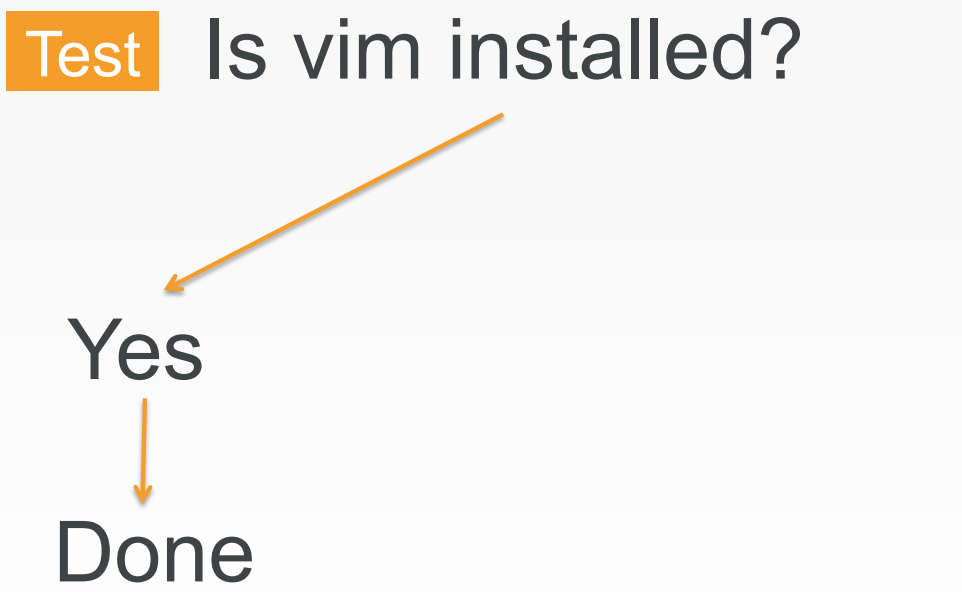
Yes

CHEF

Test and Repair

Resources follow a **test** and repair model

```
package "vim"
```

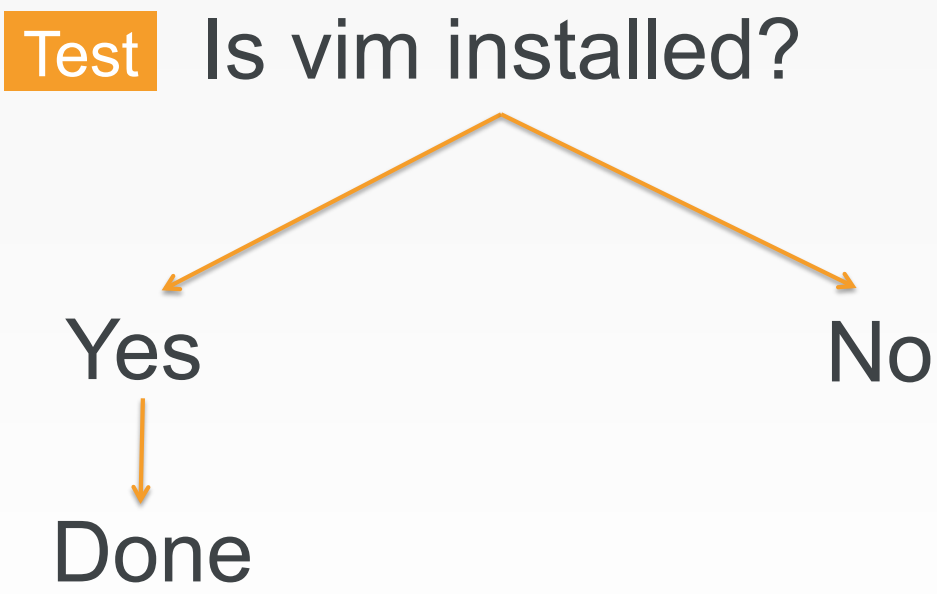


CHEF

Test and Repair

Resources follow a **test** and repair model

```
package "vim"
```

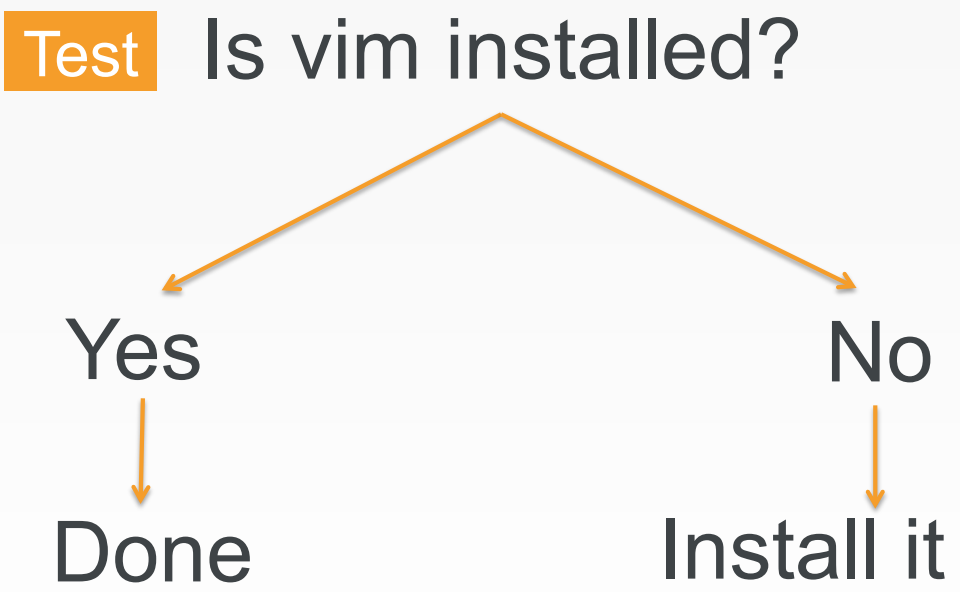


CHEF

Test and Repair

Resources follow a **test** and repair model

```
package "vim"
```

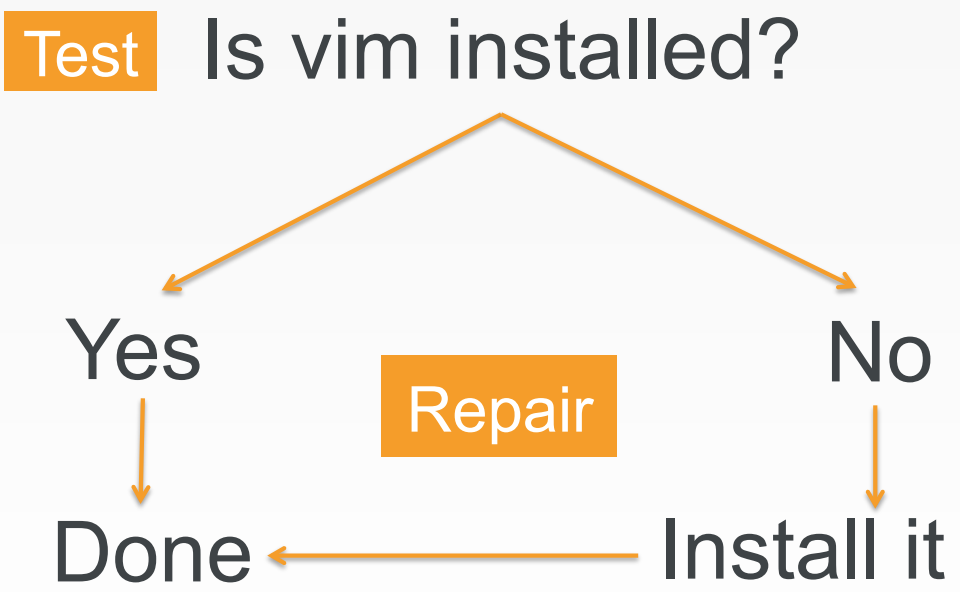


CHEF

Test and Repair

Resources follow a **test** and **repair** model

```
package "vim"
```



CHEF

Resources

- package
- template
- service
- directory
- user
- group
- dsc_script
- registry_key
- powershell_script
- cron
- mount
- route
- …and more!

Recipes

- Policy is defined as a collection of **resources** in **recipes**. There are lots of abstractions on top of this but **resources** are the basic building blocks.

Test-driven Infrastructure

Change policy with confidence

CHEF

Faster Feedback

- Speed-up the feedback loops with automated testing.
- Have confidence in your changes before you run them in production

Chef Testing

- Did chef-client complete successfully?
- Did the recipe put the node in the desired state?
- Are the resources properly defined?
- Does the code follow our style guide?

Test-driving infrastructure

- We are going to use a relatively simple scenario
- We are going to explore many facets of testing
- We are going to follow a test-first, test-driven model

Our Scenario

- We want a simple rails app deployed on a server.
- We will only have time to build some of it in today's workshop.
- You will leave with working code

Create a directory for your Chef project

```
$ chef generate repo chef-repo
```

```
Compiling Cookbooks...
Recipe: code_generator::repo
  * directory[/home/chef/chef-repo] action create
    - create new directory /home/chef/chef-repo
  * template[/home/chef/chef-repo/LICENSE] action create
    - create new file /home/chef/chef-repo/LICENSE
    - update content in file /home/chef/chef-repo/LICENSE from none to aed48c
    (diff output suppressed by config)
  * cookbook_file[/home/chef/chef-repo/README.md] action create
    - create new file /home/chef/chef-repo/README.md
    - update content in file /home/chef/chef-repo/README.md from none to 69567b
    (diff output suppressed by config)
  * cookbook_file[/home/chef/chef-repo/Rakefile] action create
    - create new file /home/chef/chef-repo/Rakefile
```

Change into that directory

```
$ cd chef-repo
```

Create an apache cookbook

```
$ chef generate cookbook cookbooks/apache
```

```
Compiling Cookbooks...
Recipe: code_generator::cookbook
  * directory[/home/chef/chef-repo/cookbooks/apache] action create
    - create new directory /home/chef/chef-repo/cookbooks/apache
  * template[/home/chef/chef-repo/cookbooks/apache/metadata.rb] action
create_if_missing
    - create new file /home/chef/chef-repo/cookbooks/apache/metadata.rb
    - update content in file /home/chef/chef-repo/cookbooks/apache/
metadata.rb from none to 4c0e2d
    (diff output suppressed by config)
  * template[/home/chef/chef-repo/cookbooks/apache/README.md] action
create_if_missing
```

Questions to ask when testing

- **Did chef-client complete successfully?**
- Did the recipe put the node in the desired state?
- Are the resources properly defined?
- Does the code following our style guide?

Chef client success status

- Requirements to verify chef-client success:
  - A place to store the cookbook artifact

Chef client success status

- Requirements to verify chef-client success:
  - A place to store the cookbook artifact
  - A chef-client with access to the cookbook

Chef client success status

- Requirements to verify chef-client success:
  - A place to store the cookbook artifact
  - A chef-client with access to the cookbook
  - A target server running the same OS as production

Test Kitchen

- Test harness to execute code on one or more platforms
- Driver plugins to allow your code to run on various cloud and virtualization providers
- Includes support for many testing frameworks
- Included with ChefDK

CHEF

Configuring the Kitchen

OPEN IN EDITOR: `cookbooks/apache/.kitchen.yml`

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-12.04
  - name: centos-6.4

suites:
  - name: default
    run_list:
      - recipe[apache::default]
    attributes:
```

SAVE FILE!

CHEF

.kitchen.yml

- driver - virtualization or cloud provider

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-12.04
  - name: centos-6.4

suites:
  - name: default
    run_list:
      - recipe[apache::default]
    attributes:
```

.kitchen.yml

- provisioner - application to configure the node

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-12.04
  - name: centos-6.4

suites:
  - name: default
    run_list:
      - recipe[apache::default]
    attributes:
```

.kitchen.yml

- platforms - target operating systems

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-12.04
  - name: centos-6.4

suites:
  - name: default
    run_list:
      - recipe[apache::default]
    attributes:
```

.kitchen.yml

- suites - target configurations

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-12.04
  - name: centos-6.4

suites:
  - name: default
    run_list:
      - recipe[apache::default]
    attributes:
```

Move to the apache cookbook directory

```
$ cd ~/chef-repo/cookbooks/apache
```

CHEF

Update .kitchen.yml

OPEN IN EDITOR: .kitchen.yml

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-12.04

suites:
  - name: default
    run_list:
      - recipe[apache::default]
    attributes:
```

SAVE FILE!

CHEF

List the Test Kitchens

```
$ kitchen list
```

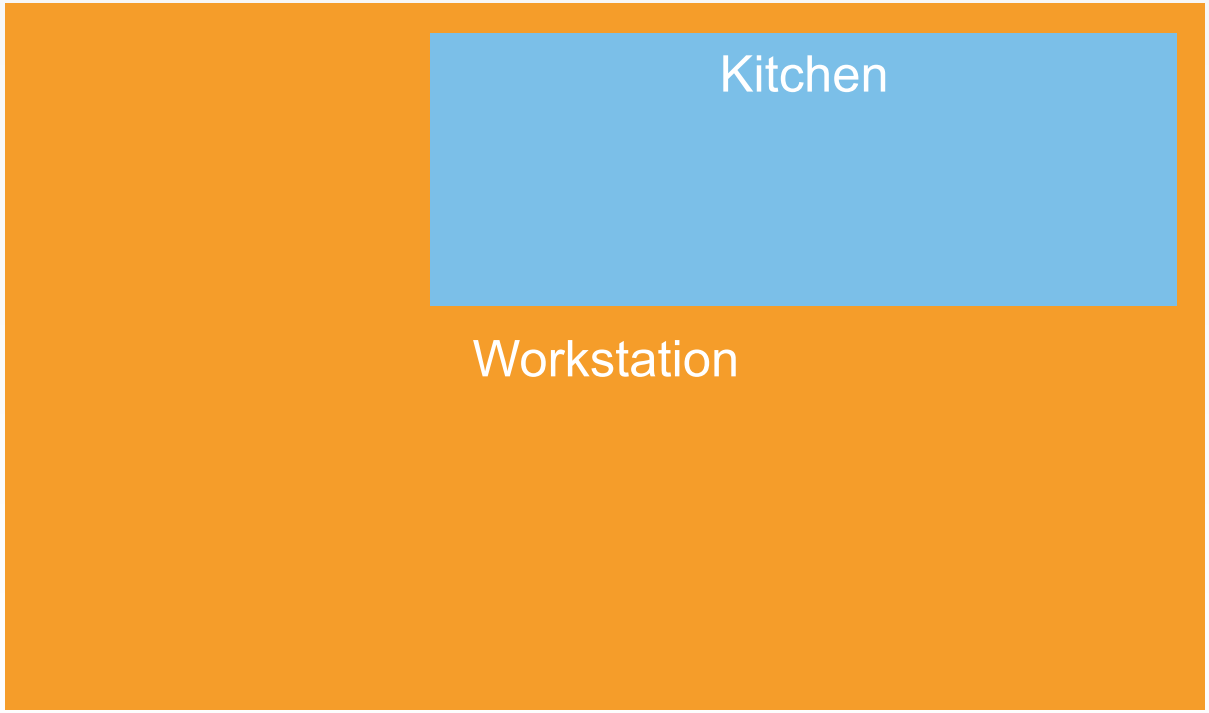
```
Instance                Driver   Provisioner   Last Action
default-ubuntu-1204     Vagrant  ChefZero      <Not Created>
```

Create the kitchen

```
$ kitchen create
```

```
-----> Starting Kitchen (v1.3.1)
-----> Creating <default-ubuntu-1204>...
       Sending build context to Docker daemon  2.56 kB
       Sending build context to Docker daemon
       Step 0 : FROM ubuntu:12.04
        ---> 0b310e6bf058
       Step 1 : RUN dpkg-divert --local --rename --add /sbin/initctl
        ---> Running in 73201c2a8836
       Leaving 'local diversion of /sbin/initctl to /sbin/initctl.distrib'
        ---> c8039cd87665
       Removing intermediate container 73201c2a8836
       Step 2 : RUN ln -sf /bin/true /sbin/initctl
        ---> Running in 4e79ba940fe4
        ---> ecc3ffe49a30
       Removing intermediate container 4e79ba940fe4
```

Kitchen created

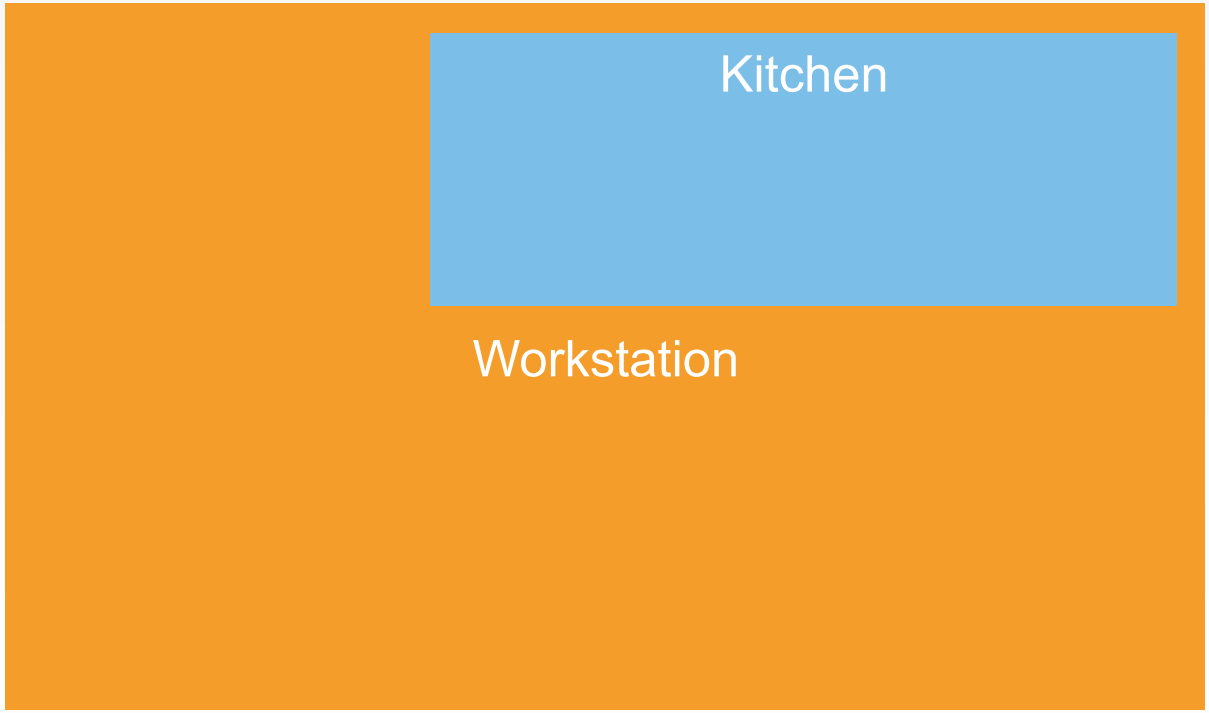


Login to the kitchen

```
$ kitchen login
```

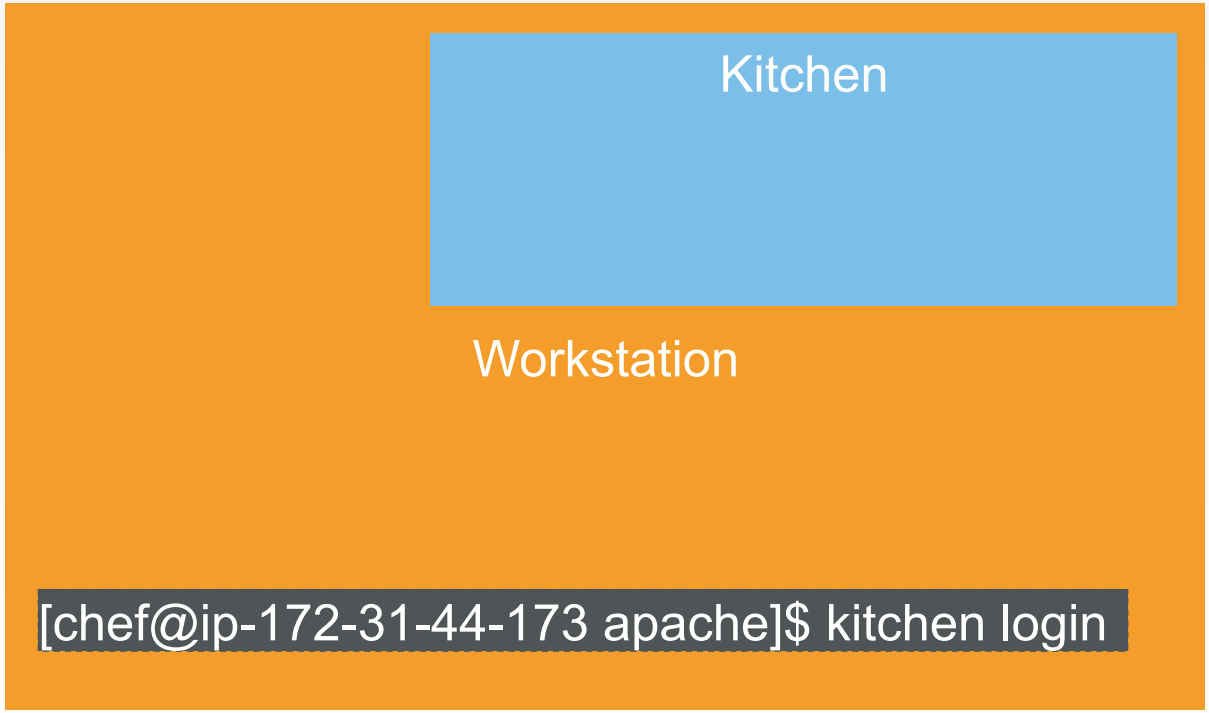
```
Welcome to Ubuntu 12.04.4 LTS (GNU/Linux 3.11.0-15-
generic x86_64)

 * Documentation:  https://help.ubuntu.com/
New release '14.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Mon Oct 19 10:14:38 2015 from 10.0.2.2
```

Kitchen login



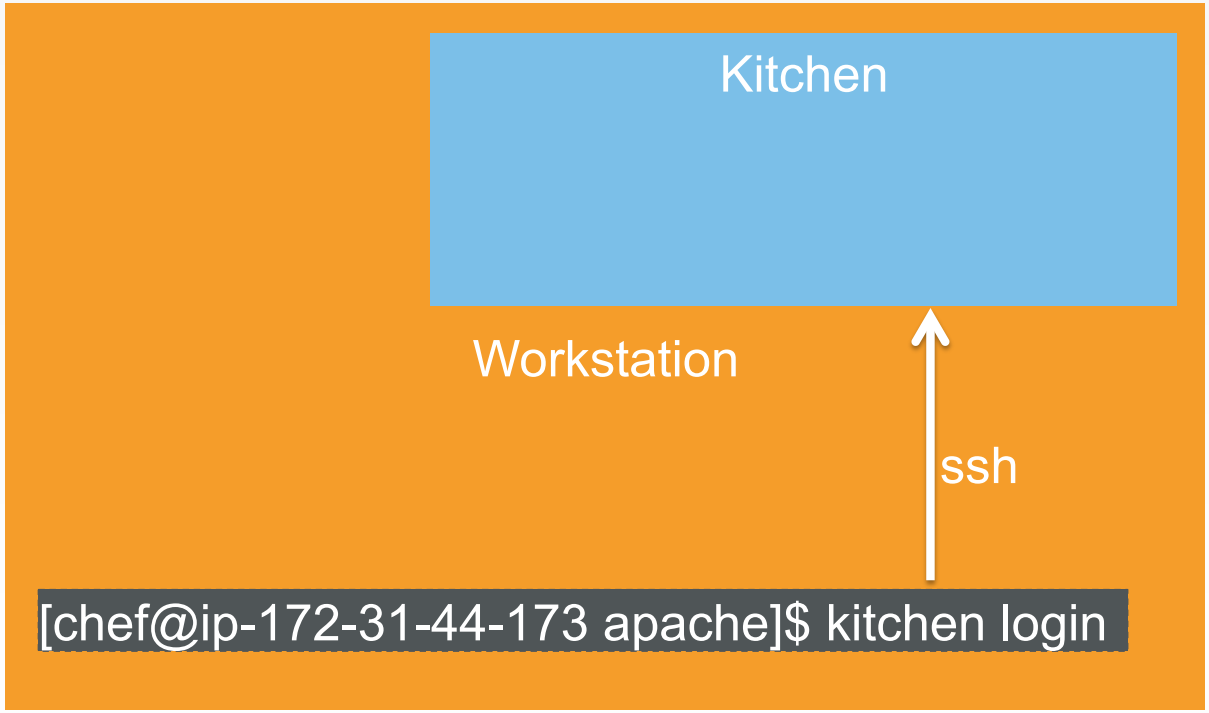
CHEF

Kitchen login



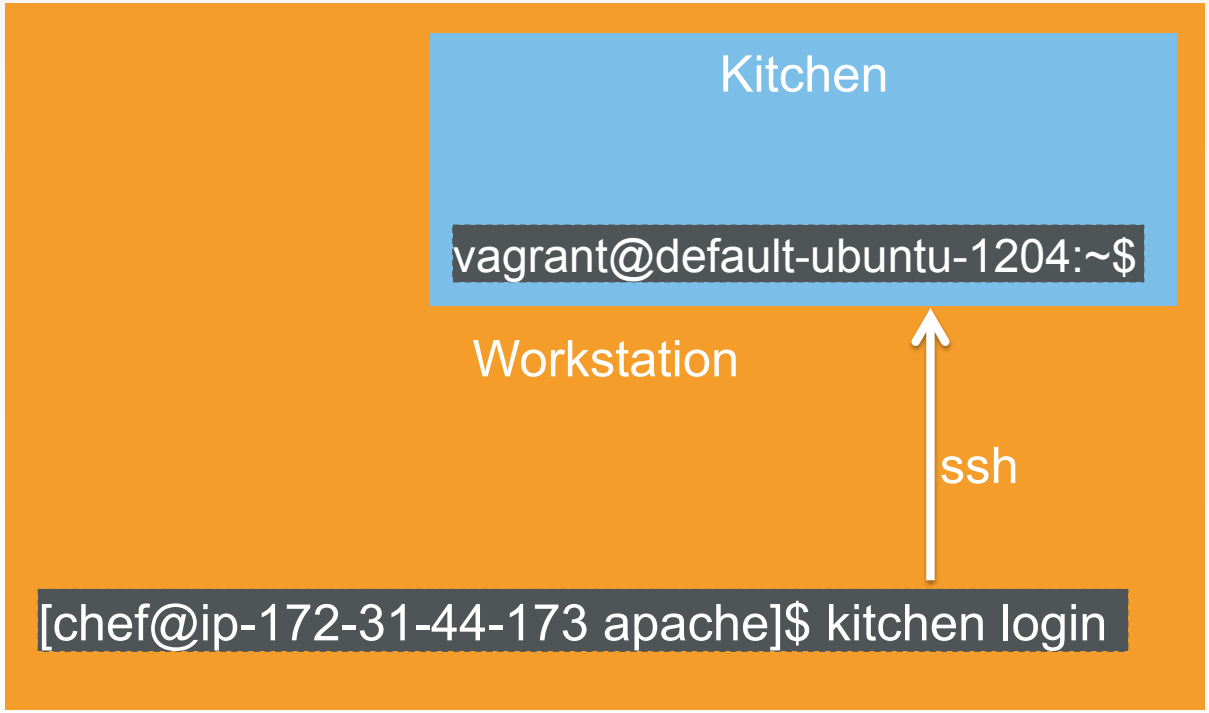
CHEF

Kitchen login



CHEF

Kitchen login



CHEF

Chef client success status

- Requirements to verify chef-client success:
  - A target server running the same OS as production
  - A chef-client with access to the cookbook

Lab – Apply our policy

- **Problem:** We have not applied our policy to the test environment.
- **Success Criteria:** The default apache recipe will be applied in the test environment

Leave the kitchen

```
$ exit
```

```
logout
Connection to 127.0.0.1 closed.
```

Go to the right place

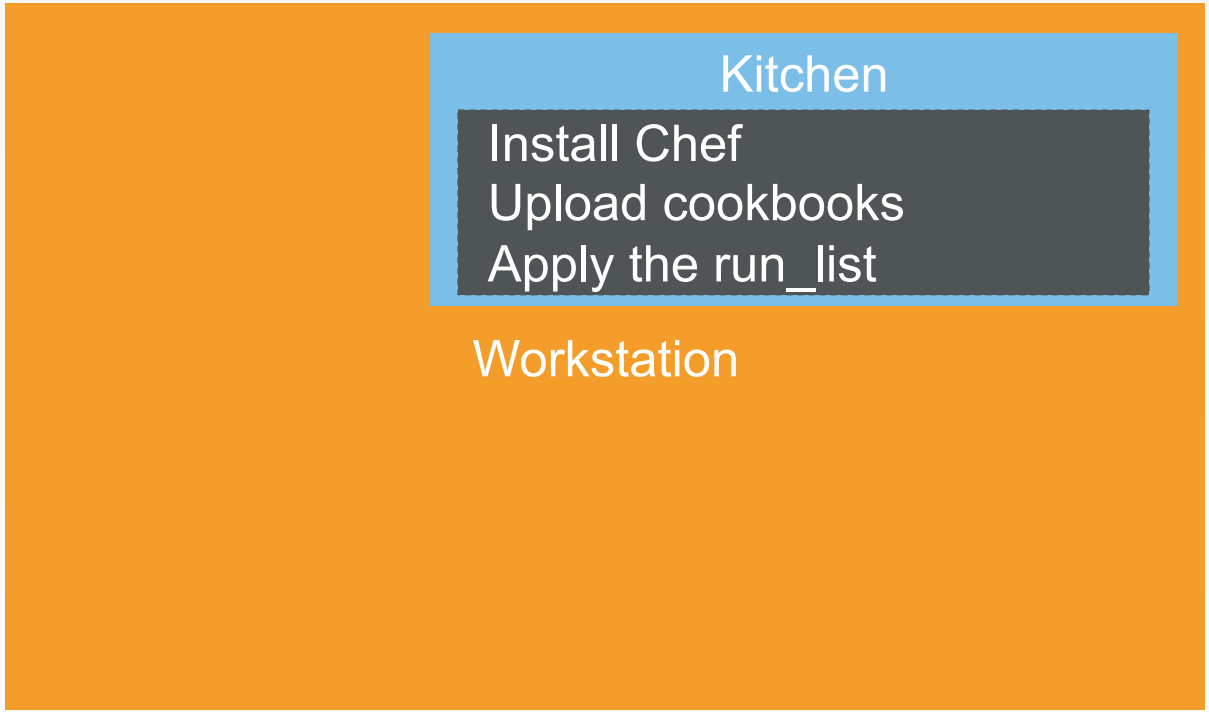
```
$ cd ~/chef-repo/cookbooks/apache
```

Apply our policy

```
$ kitchen converge
```

```
-----> Starting Kitchen (v1.3.1)
-----> Converging <default-ubuntu-1204>...
       Preparing files for transfer
       Preparing dna.json
       Resolving cookbook dependencies with Berkshelf 3.2.3...
       Removing non-cookbook files before transfer
       Preparing validation.pem
       Preparing client.rb
-----> Installing Chef Omnibus (install only if missing)
       downloading https://www.chef.io/chef/install.sh
         to file /tmp/install.sh
       trying wget...
       sudo: /etc/sudoers.d/kitchen is mode 0644, should be 0440
       Downloading Chef  for ubuntu...
```

Kitchen converge



CHEF

Questions to ask when testing

- ✓ Did chef-client complete successfully?
- Did the recipe put the node in the desired state?
- Are the resources properly defined?
- Does the code following our style guide?

Verifying node state

Serverspec

CHEF

Login to the kitchen

```
$ kitchen login
```

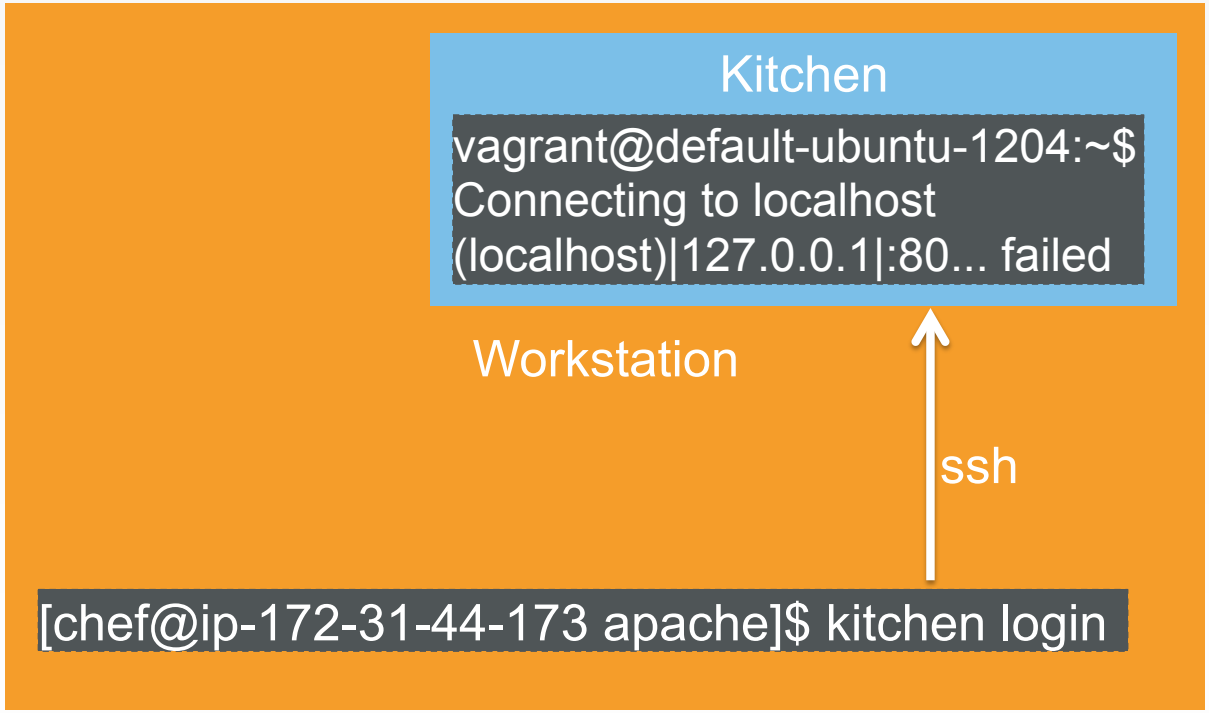
```
Welcome to Ubuntu 12.04.4 LTS (GNU/Linux 3.11.0-15-
generic x86_64)

 * Documentation:  https://help.ubuntu.com/
New release '14.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Mon Oct 19 10:14:38 2015 from 10.0.2.2
```

Manually inspect the test node

```
$ wget http://localhost
```

```
--2015-10-19 13:17:11--  http://localhost/
Resolving localhost (localhost)... 127.0.0.1
Connecting to localhost (localhost)|127.0.0.1|:80...
failed: Connection refused.
```

Kitchen login



CHEF

Lab – Verify node state

- **Problem**: Manually verifying the state of the test node is tedious and error-prone.
- **Success Criteria**: The end state of the node is automatically tested.

Serverspec

- Write tests to verify your servers
- Not dependent on Chef
- Defines many resource types
  - package, service, user, etc.
- Works well with Test Kitchen
- http://serverspec.org/

CHEF

Leave the Kitchen

```
$ exit
```

```
logout
Connection to localhost closed.
```

Move to the proper directory

```
$ cd ~/chef-repo/cookbooks/apache
```

Write a Serverspec test

OPEN IN EDITOR: test/integration/default/serverspec/default_spec.rb

```
require 'spec_helper'

describe 'apache::default' do

  # Serverspec examples can be found at
  # http://serverspec.org/resource_types.html

  it 'does something' do
    skip 'Replace this with meaningful tests'
  end

end
```

SAVE FILE!

CHEF

Generic Expectation Form

```
describe "<subject>" do
  it "<description>" do
    expect(thing).to eq result
  end
end
```

Awesome Expectations

OPEN IN EDITOR: test/integration/default/serverspec/default_spec.rb

```
require 'spec_helper'

describe "apache::default" do
  it "is awesome" do
    expect(true).to eq true
  end
end
```

SAVE FILE!

CHEF

Run the serverspec test

```
$ kitchen verify
```

```
-----> Running serverspec test suite
       /opt/chef/embedded/bin/ruby -I/tmp/busser/suites/serverspec -I/tmp/
busser/gems/gems/rspec-support-3.1.2/lib:/tmp/busser/gems/gems/rspec-
core-3.1.7/lib /opt/chef/embedded/bin/rspec --pattern /tmp/busser/suites/
serverspec/\*\*/\*_spec.rb --color --format documentation --default-path /
tmp/busser/suites/serverspec

       apache::default
         is awesome

       Finished in 0.02823 seconds (files took 0.99875 seconds to load)
       1 example, 0 failures
       Finished verifying <default-centos-64> (0m5.03s).
```

How would you test our criteria?

- What would you test to make sure apache is running?

Verify package is installed

OPEN IN EDITOR: test/integration/default/serverspec/default_spec.rb

```
require 'spec_helper'

describe "apache" do
  it "is awesome" do
    expect(true).to eq true
  end

  it "is installed" do
    expect(package("apache2")).to be_installed
  end
end
```

SAVE FILE!

CHEF

Exercise the test

```
$ kitchen verify
```

```
 apache::default
         is awesome
         is installed (FAILED - 1)

       Failures:

         1) apache::default is installed
            Failure/Error: expect(package 'apache2').to be_installed
              expected Package "apache2" to be installed
              /bin/sh -c dpkg-query\ -f\ \'\$\{Status\}\'\ -W\ apache2\ \|\
grep\ -E\ \'\^\(install\|hold\)\ ok\ installed\$\'
              No packages found matching apache2.
```

Test is failing, make it pass

- Test-driven development involves
  - Write a test to verify something is working
  - Watch the test fail
  - Write just enough code to make the test pass
  - Repeat

Update our cookbook

OPEN IN EDITOR: ~/chef-reop/cookbooks/apache/recipes/default.rb

```
package "apache2"
```

SAVE FILE!

CHEF

Converge the node again

```
$ kitchen converge
```

```
-----> Starting Kitchen (v1.3.1)
-----> Converging <default-ubuntu-1204>...
       Preparing files for transfer
       Preparing dna.json
       Resolving cookbook dependencies with Berkshelf 3.2.3...
       Removing non-cookbook files before transfer
       Preparing validation.pem
       Preparing client.rb
-----> Chef Omnibus installation detected (install only if missing)
       sudo: /etc/sudoers.d/kitchen is mode 0644, should be 0440
       Transferring files to <default-ubuntu-1204>
       sudo: /etc/sudoers.d/kitchen is mode 0644, should be 0440
       Starting Chef Client, version 12.2.1
       [2015-04-15T22:02:01+00:00] WARN: Child with name 'dna.json' found in multiple directories: /tmp/
kitchen/dna.json and /tmp/kitchen/dna.json
       resolving cookbooks for run list: ["apache::default"]
```

Exercise the test

```
$ kitchen verify
```

```
 apache
         is awesome
         is installed

       Finished in 0.48165 seconds (files took 1.05
seconds to load)
       2 examples, 0 failures
       Finished verifying <default-centos-64>
(0m5.64s).
-----> Kitchen is finished. (0m11.84s)
```

What else will you test?

- Is the service running?
- Is the port accessible?
- Is the expected content being served?

- Make sure everything works from a fresh kitchen, too!

Extend the Serverspec test

OPEN IN EDITOR: test/integration/default/serverspec/default_spec.rb

```
describe 'apache' do
  it "is installed" do
    expect(package 'apache2').to be_installed
  end

  it "is running" do
    expect(service 'apache2').to be_running
  end

  it "responds to http requests" do
    expect(command("curl localhost").exit_status).to eq 0
  end
end
```

SAVE FILE!

CHEF

Verify the kitchen

```
$ kitchen verify
```

```
Failures:

  1) apache::default is running
     Failure/Error: expect(service 'apache2').to be_running
     expected Service "apache2" to be running
    /bin/sh -c service\ apache2\ status\ \&\&\ service\ apache2\ status\ \|\ grep\
\'running\'
               Apache2 is NOT running.

  2) apache::default responds to http requests
     Failure/Error: expect(command("curl localhost").exit_status).to eq 0

     expected: 0
     got: 7
```

Update our cookbook

OPEN IN EDITOR: ~/chef-repo/cookbooks/apache/recipes/default.rb

```
package "apache2”

service "apache2” do
  action :start
end
```

SAVE FILE!

CHEF

Converge the node again

```
$ kitchen converge
```

```
-----> Starting Kitchen (v1.3.1)
-----> Converging <default-ubuntu-1204>...
       Preparing files for transfer
       Preparing dna.json
       Resolving cookbook dependencies with Berkshelf 3.2.3...
       Removing non-cookbook files before transfer
       Preparing validation.pem
       Preparing client.rb
-----> Chef Omnibus installation detected (install only if missing)
       sudo: /etc/sudoers.d/kitchen is mode 0644, should be 0440
       Transferring files to <default-ubuntu-1204>
       sudo: /etc/sudoers.d/kitchen is mode 0644, should be 0440
       Starting Chef Client, version 12.2.1
       [2015-04-15T22:02:01+00:00] WARN: Child with name 'dna.json' found in multiple directories: /tmp/
kitchen/dna.json and /tmp/kitchen/dna.json
       resolving cookbooks for run list: ["apache::default"]
```

Verify the kitchen

```
$ kitchen verify
```

```
 apache::default
          is awesome
          is installed
          is running
          responds to http requests

        Finished in 0.9764 seconds (files took 1.22 seconds to
load)
        4 examples, 0 failures

        Finished verifying <default-ubuntu-1204> (0m9.78s).
-----> Kitchen is finished. (0m11.23s)
```

Kitchen Workflow

- `kitchen create`
- `kitchen converge`
- `kitchen verify`
- `kitchen destroy`

- All at once with `kitchen test`

Chef Testing

- ✓ Did chef-client complete successfully?
- ✓ Did the recipe put the node in the desired state?
- Are the resources properly defined?
- Does the code following our style guide?

Even Faster Feedback

ChefSpec

CHEF

Chef Testing

- ✓ Did chef-client complete successfully?
- ✓ Did the recipe put the node in the desired state?
- Are the resources properly defined?
- Does the code following our style guide?

This is too slow!

- To test our code, we need to spin up a test kitchen, converge a node, execute some tests.
- Our simple test case takes about 2 minutes to fully execute.

Properly configured resources

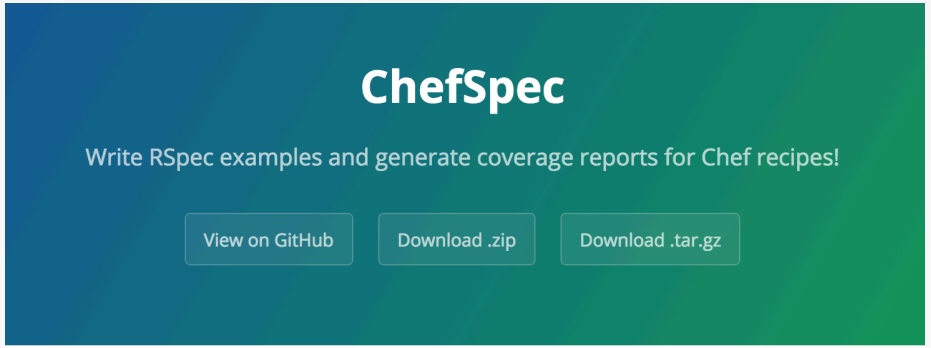
- We need a way to verify that the resources in our recipes are properly configured
- We want to get faster feedback

Lab – Verify the resources

- **Problem**: We should be able to catch errors before we need to converge a node
- **Success Criteria**: Catch a typo prior to converge

ChefSpec

- Test before you converge
- Get feedback on cookbook changes without the need for target servers



ChefSpec

gem v4.2.0 build passing

ChefSpec is a unit testing framework for testing Chef cookbooks. ChefSpec makes it easy to write examples and get fast feedback on cookbook changes without the need for virtual machines or cloud servers.

ChefSpec runs your cookbook(s) locally with Chef Solo without actually converging a node. This has two primary benefits:

- It's really fast!
- Your tests can vary node attributes, operating systems, and search results to assert behavior under varying conditions.

http://sethvargo.github.io/chefspec/

CHEF

Change to the apache cookbook directory

```
$ cd ~/chef-repo/cookbooks/apache
```

CHEF

Write a ChefSpec test

OPEN IN EDITOR: spec/unit/recipes/default_spec.rb

```
require 'spec_helper'

describe 'apache::default' do

  context 'When all attributes are default, on an unspecified platform' do

    let(:chef_run) do
      runner = ChefSpec::ServerRunner.new
      runner.converge(described_recipe)
    end

    it 'converges successfully' do
      chef_run # This should not raise an error
    end

  end
end
```

SAVE FILE!

CHEF

Write a ChefSpec test

OPEN IN EDITOR: spec/unit/recipes/default_spec.rb

```
describe 'apache::default' do
  context 'When all attributes are default, on an unspecified platform' do

    let(:chef_run) do
      runner = ChefSpec::ServerRunner.new
      runner.converge(described_recipe)
    end

    it 'converges successfully' do
      chef_run # This should not raise an error
    end

    it 'installs apache' do
      expect(chef_run).to install_package 'apache2'
    end
  end
end
```

SAVE FILE!

CHEF

Run the ChefSpec tests

```
$ rspec spec
```

```
..

Finished in 1.19 seconds (files took 13.09 seconds to load)
2 examples, 0 failures
```

Break the cookbook

OPEN IN EDITOR: recipes/default.rb

```
package "apache"

service "apache2" do
  action :start
end
```

SAVE FILE!

CHEF

Run the ChefSpec tests

```
$ rspec spec
```

```
.F

Failures:

  1) apache::default When all attributes are default, on an unspecified platform installs apache
     Failure/Error: expect(chef_run).to install_package 'apache2'
       expected "package[apache2]" with action :install to be in Chef run. Other package
resources:

         package[apache]

     # ./spec/unit/recipes/default_spec.rb:23:in `block (3 levels) in <top (required)>'

Finished in 1.18 seconds (files took 12.7 seconds to load)
2 examples, 1 failure
```

Fix the cookbook

OPEN IN EDITOR: recipes/default.rb

```
package "apache2"

service "apache2" do
  action :start
end
```

SAVE FILE!

CHEF

Run the ChefSpec tests

```
$ rspec spec
```

```
..

Finished in 1.19 seconds (files took 13.09 seconds to load)
2 examples, 0 failures
```

Chef Testing

- ✓ Did chef-client complete successfully?
- ✓ Did the recipe put the node in the desired state?
- ✓ Are the resources properly defined?
- Does the code following our style guide?

Clean code

Follow best practices, avoid mistakes

CHEF

Foodcritic

- Check cookbooks for common problems
- Style, correctness, deprecations, etc.
- Included with ChefDK

http://www.foodcritic.io/

CHEF

Change our recipe

OPEN IN EDITOR: recipes/default.rb

```
package_name = "apache2"

package "#{package_name}"

service "apache2" do
  action :start
end
```

SAVE FILE!

CHEF

Run Foodcritic

```
$ foodcritic .
```

```
FC002: Avoid string interpolation
where not required: ./recipes/
default.rb:7
```

Chef Testing

- ✓ Did chef-client complete successfully?
- ✓ Did the recipe put the node in the desired state?
- ✓ Are the resources properly defined?
- ✓ Does the code following our style guide?

Next steps

- Install ruby on the system
- Install and configure passenger
- Install and configure a database
- Deploy the application
- Demo the working solution

Building the Rails App

CHEF

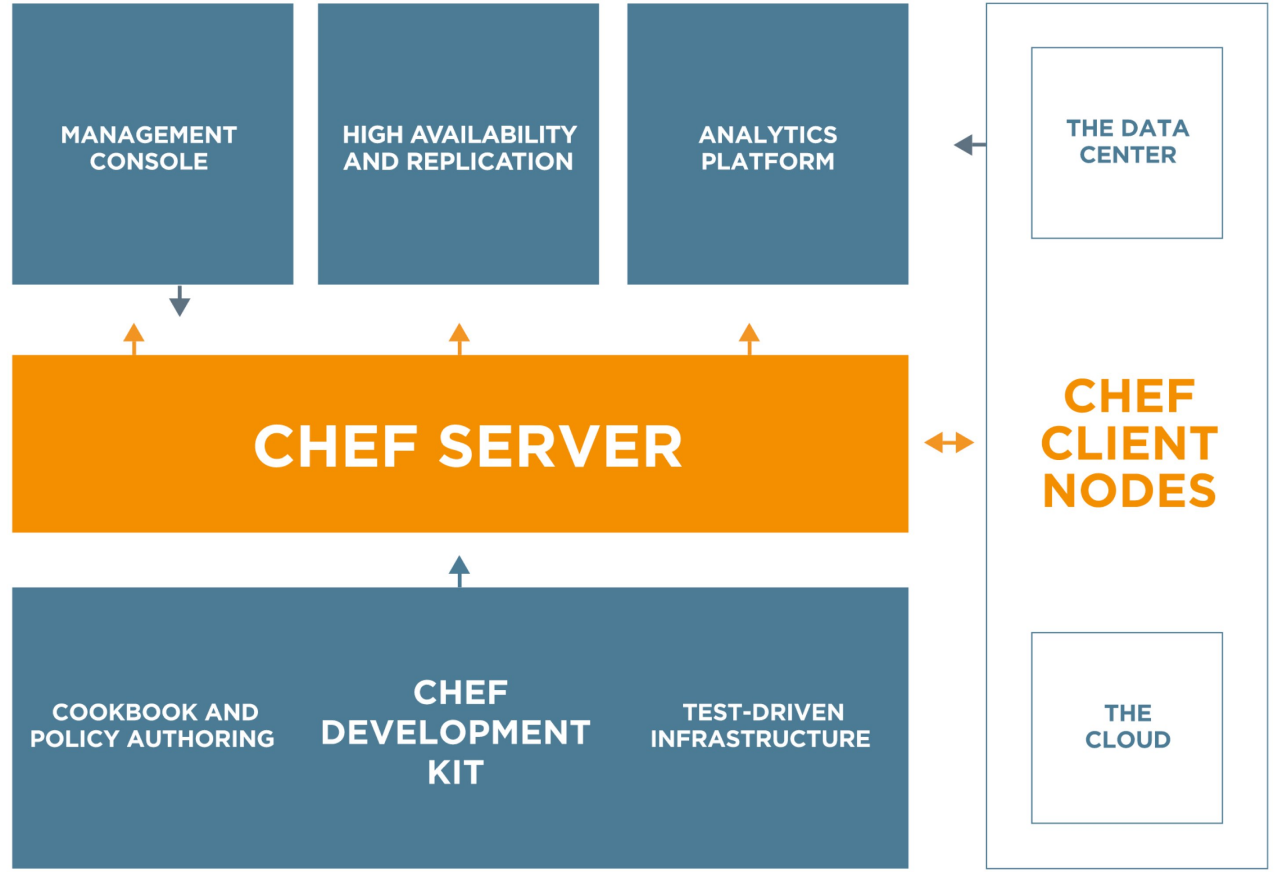
Launch the Kitchen

- From `chef-repo/cookbooks/widget_world_application`
- `kitchen converge`
- Open http://localhost:8080

Wrapping Up

CHEF

We've only scratched the surface



https://www.chef.io/chef/

CHEF

Build Anything

- Simple internal applications
- Complex external applications
- Workstations
- Hadoop clusters
- IaaS infrastructure
- PaaS infrastructure
- SaaS applications
- Storage systems
- You name it

http://www.flickr.com/photos/hyku/245010680/

CHEF

And Manage it Simply

http://www.flickr.com/photos/helico/404640681/

- Automatically reconfigure everything
- Linux, Windows, Unixes, BSDs
- Load balancers
- Metrics collection systems
- Monitoring systems
- Cloud migrations become trivial

CHEF

What questions do you have?

- Ask me anything!
- @nathenharvey
- Thank you!