



Chef Fundamentals

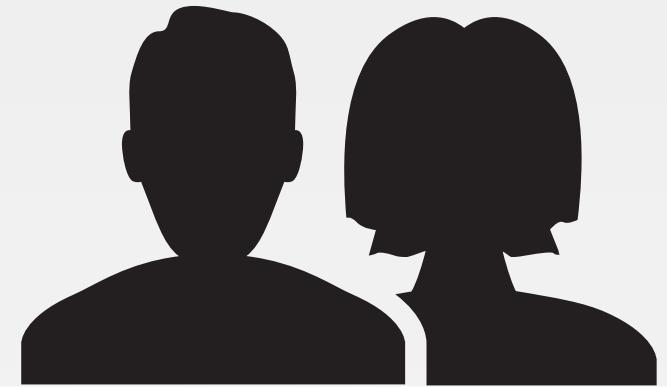
An Introduction to the ChefDK

v3.0.0



HELLO

Instructor Name



Titles, Accomplishments, Etc.

DISCUSSION

Introduce Yourselves



Name

Current job role

Previous job roles/background

Experience with Chef and/or config management

Favorite Text Editor

Our Expectations

You will leave this workshop with a basic understanding Chef's core components and ensure you have a workstation with all necessary tools installed.



Your Infrastructure

You bring with you your own domain expertise and problems. Chef is a framework for solving those problems. Our job is to teach you how to express solutions to your problems with Chef.

Learning the Language

Learning Chef is like learning a language. You will reach fluency very fast but it will take practice until you become comfortable.

To best way to learn Chef is to use Chef.

A Taste of Chef

Chef is a large set of tools that are able to be used on multiple platforms and in numerous configurations. We will have time to only explore some of its most fundamental pieces.

Ask Me Anything (AMA)

All of us are coming here with *unique* experiences and from *unique* teams that are using Chef in *unique* ways. It is important that we answer your questions and set you on the path to find more.

Break It!



If everything works the first time go back and make some changes. Break it! It's rare that you have a safe space like this to explore. Sometimes its more important to know what something looks like when it does not work than when it does work.



Agenda

v3.0.0



Day 1

Getting a Workstation
Using Resources
Building Cookbooks
Applying multiple recipes
Testing with Test Kitchen
Node Attributes
Templates
Workstation Setup

Day 2

Connecting To Chef Server
Managing Multiple Nodes
Community Cookbooks
Roles
Search
Environments



Getting a Workstation

v3.0.0



Using a Cloud Workstation

To ensure the "smoothest" setup experience we will be using a virtual machine with all the necessary tools installed.

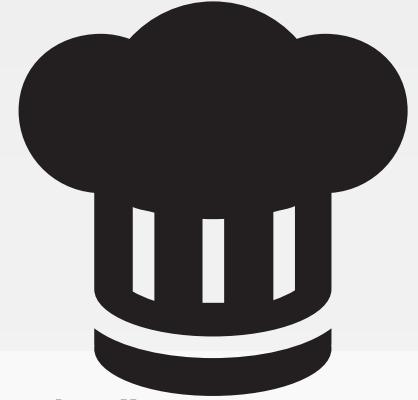
What About My Workstation?

At the end of the workshop we will have an
InstallFest!

During that time we will install all the necessary tools on your workstation and troubleshoot any installation issues you may experience.

LAB

Getting Your Workstation



"Let's all hope the WiFi holds up to the onslaught."
– Conference Attendee

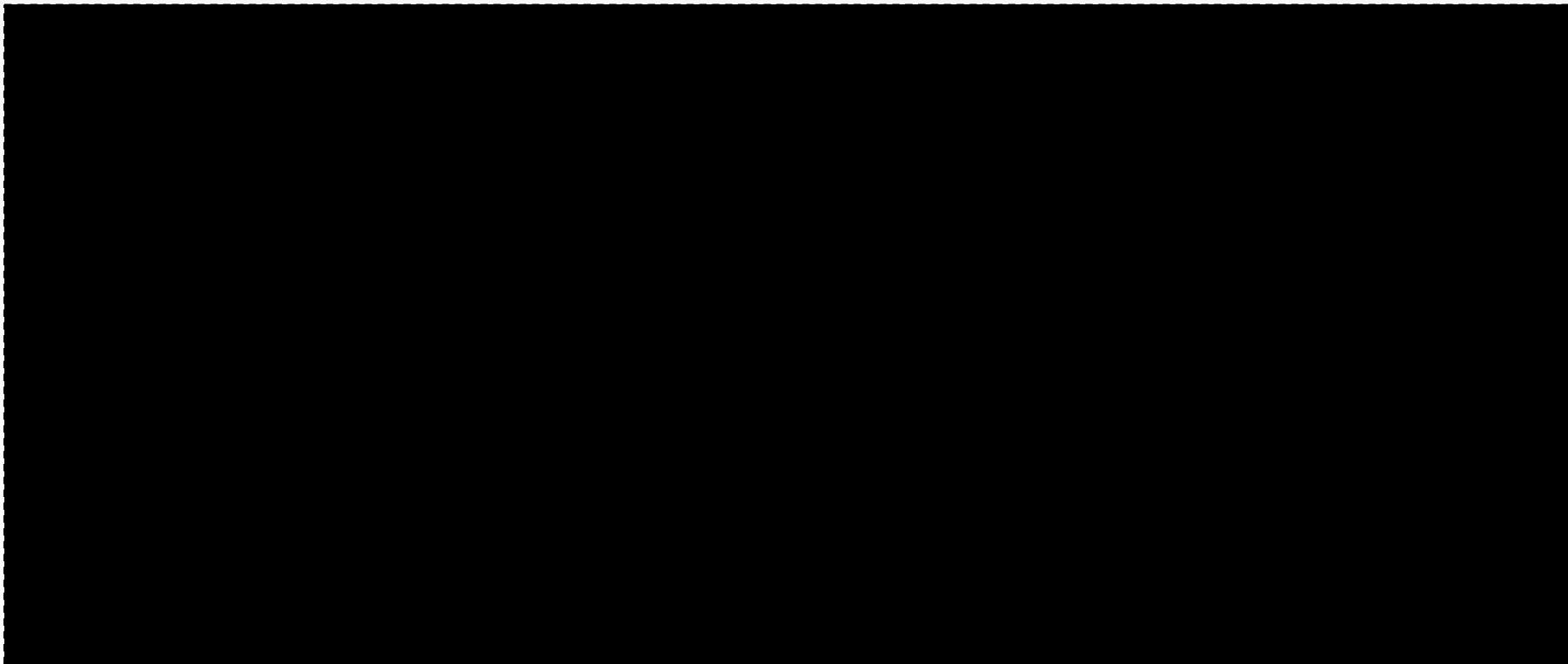
OBJECTIVE:

- ssh into the remote machine

SSH into the remote system



```
$ ssh ADDRESS -l chef
```



Amazon 2015.03 on EC2



The chef user has been granted password-less sudoers access.

The following software is installed:

Chef DK
Docker
kitchen-docker gem



Resources

Chef's Fundamental Building Blocks

v3.0.0



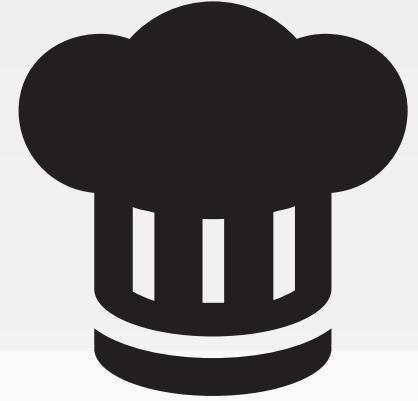
Welcome Email!

... listen, really sorry I didn't get a chance to finish setting up your workstation. There was another outage this morning!

Hey, also it would be awesome if you could test a config management tool to make it easier for the next few hires. Try chef-apply when you log in ...

LAB

How about an \$EDITOR?



Did she install anything another than some Chef tools? Is that all I'll need?

OBJECTIVE:

Install a command-line text editor (i.e. emacs, nano, vim).

- Pick an editor
- Install the editor through chef-apply

Choose an Editor

Emacs

Nano

Vim

EMACS

OPEN FILE \$ emacs FILENAME

WRITE FILE ctrl+x, ctrl+w

EXIT ctrl+x, ctrl+c

NANO

OPEN FILE \$ nano FILENAME

WRITE (WHEN EXITING)
 ctrl+x, y, ENTER
EXIT ctrl+x

VIM

OPEN FILE	\$ vim FILENAME
START EDITING	i
WRITE FILE	ESC, :w
EXIT	ESC, :q
EXIT (don't write)	ESC, :q!

How about nano?



```
$ nano
```

The program 'nano' is currently not installed. To run 'nano' please ask your administrator to install the package 'nano'

How about vim?



```
$ vim
```

The program 'vim' can be found in the following packages:

- * vim
- * vim-gnome
- * vim-tiny
- * vim-athena
- * vim-gtk
- * vim-nox

How about emacs?



```
$ which emacs
```

The program 'emacs' can be found in the following packages:

- * emacs24
- * emacs24-nox
- * e3
- * emacs23
- * emacs23-lucid
- * emacs23-nox
- * emacs24-lucid
- * jove

Ask your administrator to install one of them

Learning Chef

The best way to learn Chef is to use Chef.



CONCEPT

What is chef-apply?



An executable program that allows you to work with resources and recipe files.

What can chef-apply do?



```
$ sudo chef-apply --help
```

```
Usage: chef-apply [RECIPE_FILE] [-e RECIPE_TEXT] [-s]
      --[no-]color                      Use colored output, defaults to enabled
      -e, --execute RECIPE_TEXT          Execute resources supplied in a string
      -j JSON_ATTRIBS,                  Load attributes from a JSON file or URL
      --json-attributes
      -l, --log_level LEVEL            Set the log level (debug, info, warn, error, fatal)
      --minimal-ohai                    Only run the bare minimum ohai plugins chef need ...
      -s, --stdin                      Execute resources read from STDIN
      -v, --version                    Show chef version
      -W, --why-run                   Enable whyrun mode
      -h, --help                       Show this message
```

DOCS

Resources



A resource is a statement of configuration policy. It describes the desired state of an element of your infrastructure, along with the steps needed to bring that item to the desired state. Each resource statement includes the resource type (such as ...

Example: Package

```
package 'httpd'
```

The package named "httpd" is installed.

Example: Service

```
service 'ntp' do
  action [:start, :enable]
end
```

The service named "ntp" is enabled (start on reboot) and started.

Example: File

```
file '/etc/motd' do
  content 'This company is the property ...'
end
```

The file name "/etc/motd" is created with content "This company is the property ..."

Example: File

```
file '/etc/motd' do
  action :delete
end
```

The file name "/etc/motd" is deleted.

Let's try out execute



```
$ sudo chef-apply --help
```

```
Usage: chef-apply [RECIPE_FILE] [-e RECIPE_TEXT] [-s]
      --[no-]color           Use colored output, defaults to enabled
      -e, --execute RECIPE_TEXT Execute resources supplied in a string
      -j JSON_ATTRIBS,       Load attributes from a JSON file or URL
      --json-attributes
      -l, --log_level LEVEL Set the log level (debug, info, warn, error, fatal)
      --minimal-ohai         Only run the bare minimum ohai plugins chef need ...
      -s, --stdin            Execute resources read from STDIN
      -v, --version          Show chef version
      -W, --why-run          Enable whyrun mode
      -h, --help              Show this message
```

Example: Installing nano



```
$ sudo chef-apply -e "package 'nano'"
```

```
Recipe: (chef-apply cookbook)::(chef-apply recipe)
```

```
* yum_package[nano] action install
```

```
- install version 2.0.9-7.el6 of package nano
```

Did I install nano?



```
$ which nano
```

```
/usr/bin/nano
```

DISCUSSION

Test and Repair



What happens when I run the command again?

DISCUSSION

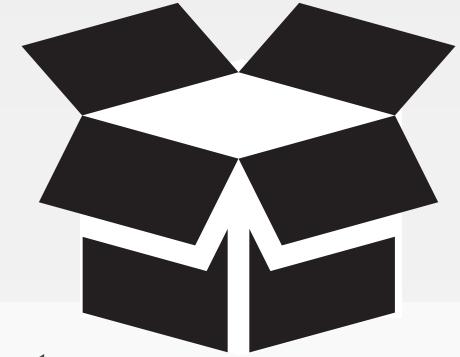
Test and Repair



What would happen if the package were to become uninstalled?

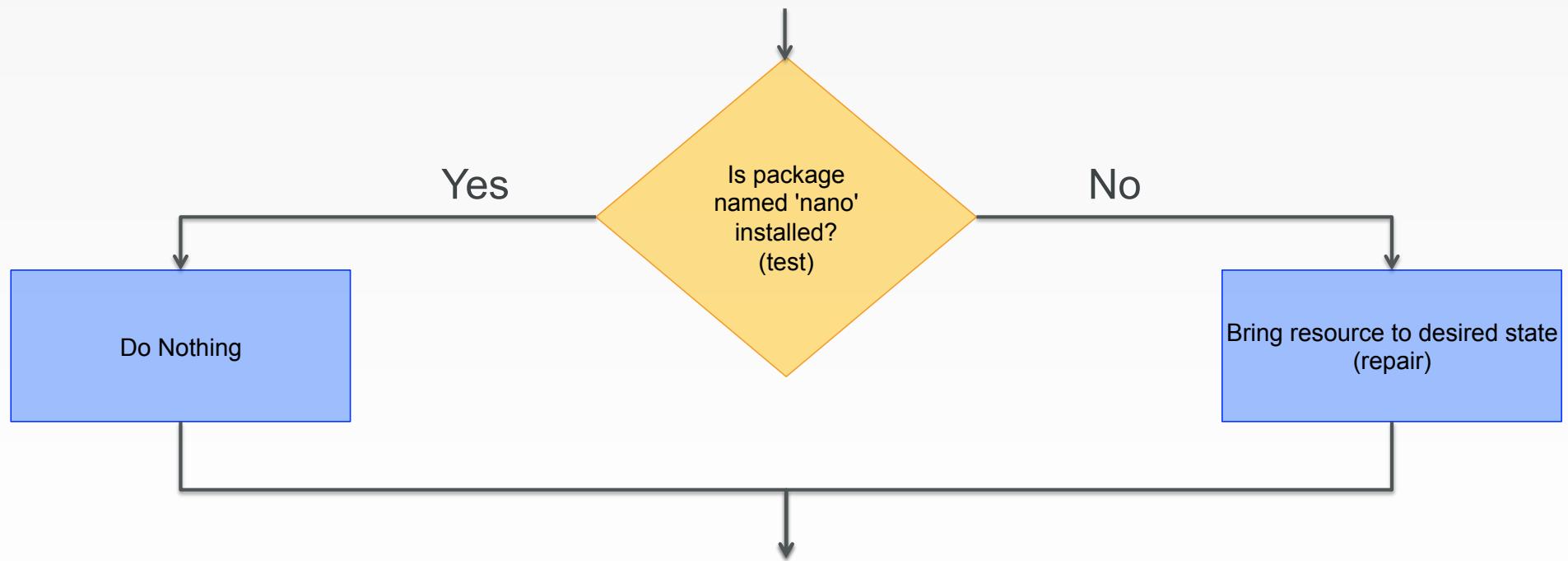
CONCEPT

Test and Repair



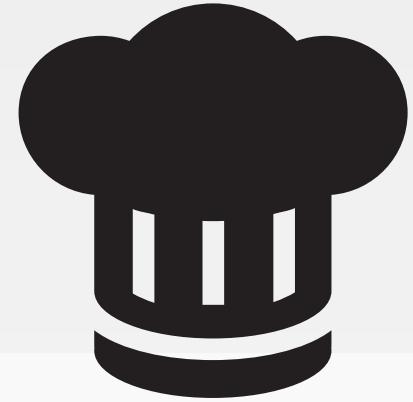
chef-apply takes action only when it needs to. Think of it as test and repair. Chef looks at the current state of each resource and takes action only when that resource is out of policy.

package 'nano'



LAB

Hello, World?



I heard Chef is written in Ruby. If that's the case its required that we write a quick "Hello, world!" application.

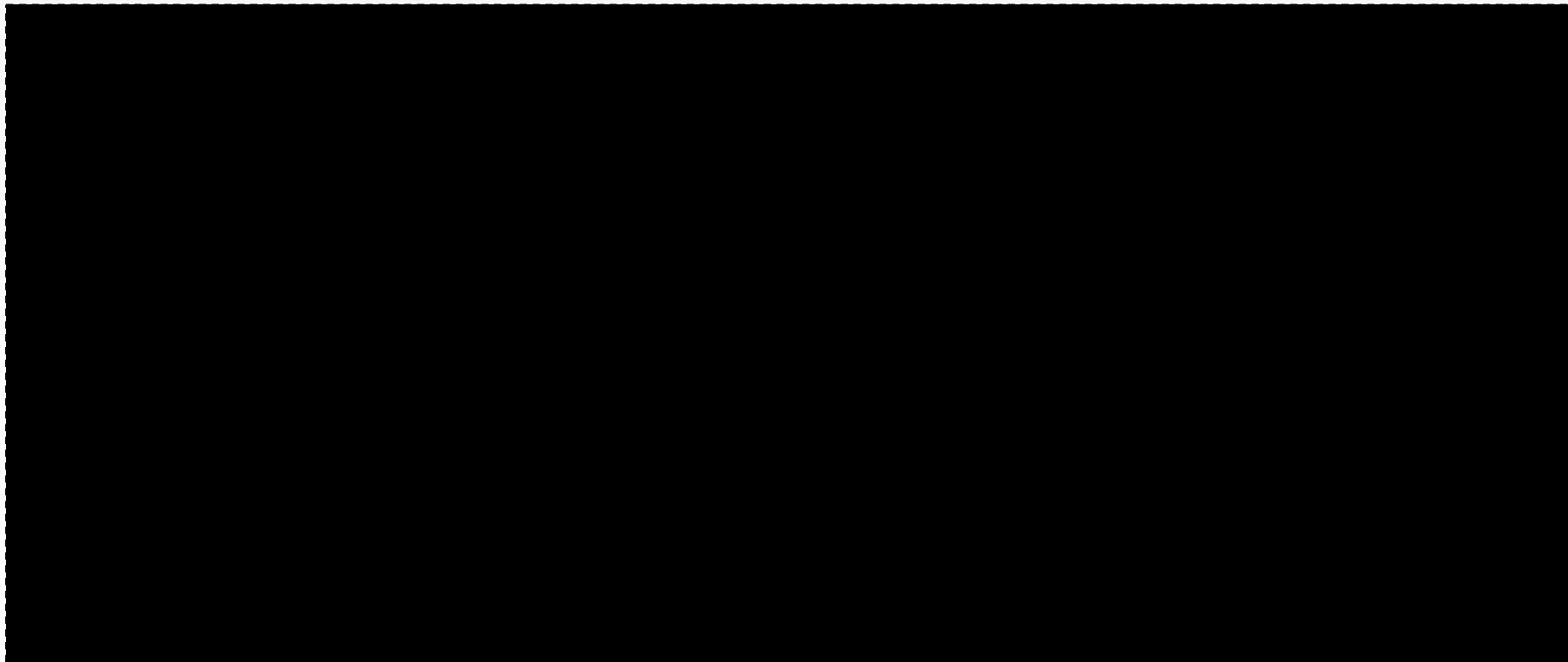
OBJECTIVE:

- Create a recipe file that defines the policy:
- The file named "hello.txt" is created with the content "Hello, world!".

Create and open a recipe file



```
$ nano hello.rb
```



v3.0.0

Creating a recipe file named hello.rb

```
~/.chef/recipes/hello.rb
```

```
file 'hello.txt' do
  content 'Hello, world!'
end
```

The file named "hello.txt" is created with the content "Hello, world!".

Can chef-apply run a recipe file?



```
$ sudo chef-apply --help
```

```
Usage: chef-apply [RECIPE_FILE] [-e RECIPE_TEXT] [-s]
      --[no-]color                         Use colored output, defaults to enabled
      -e, --execute RECIPE_TEXT            Execute resources supplied in a string
      -j JSON_ATTRIBS,                   Load attributes from a JSON file or URL
      --json-attributes
      -l, --log_level LEVEL              Set the log level (debug, info, warn, error, fatal)
      --minimal-ohai                      Only run the bare minimum ohai plugins chef need ...
      -s, --stdin                         Execute resources read from STDIN
      -v, --version                       Show chef version
      -W, --why-run                       Enable whyrun mode
      -h, --help                           Show this message
```

Example: Applying a recipe file



```
$ sudo chef-apply hello.rb
```

```
Recipe: (chef-apply cookbook)::(chef-apply recipe)
  * file[hello.txt] action create
    - create new file hello.txt
    - update content in file hello.txt from none to 315f5b
      --- hello.txt      2015-05-11 23:16:05.077570000 +0000
      +++ ./hello.txt20150511-1615-14378d5      2015-05-11 23:16:05.077570000 +0000
      @@ -1 +1,2 @@
      +Hello, world!
```

What does hello.txt say?



```
$ cat hello.txt
```

```
Hello, world!
```

DISCUSSION

Test and Repair



What happens when I run the command again?

DISCUSSION

Test and Repair



What happens when the file contents is modified?

DISCUSSION

Test and Repair

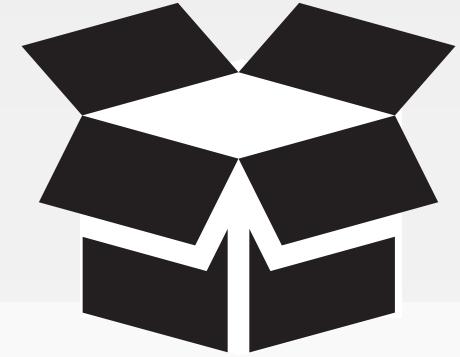


What happens when the file is removed?

CONCEPT

Resource Definition

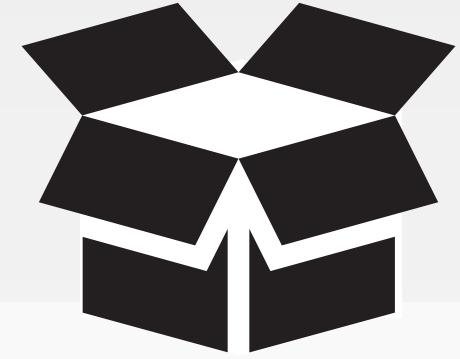
```
file 'hello.txt' do
  content 'Hello, world!'
end
```



The **TYPE** named **NAME** should be **ACTION'd** with **ATTRIBUTES**

CONCEPT

Resource Definition

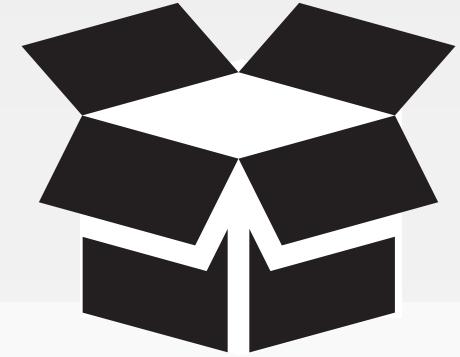


```
file 'hello.txt' do
  content 'Hello, world!'
end
```

The **TYPE** named **NAME** should be **ACTION'd** with **ATTRIBUTES**

CONCEPT

Resource Definition

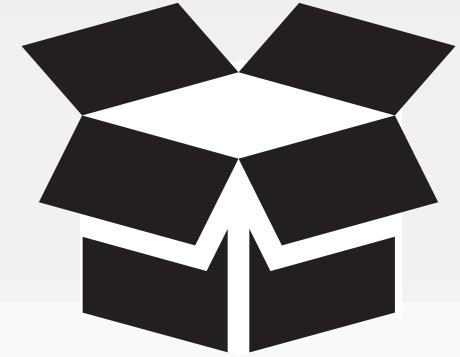


```
file 'hello.txt' do
  content 'Hello, world!'
end
```

The **TYPE** named **NAME** should be **ACTION'd** with **ATTRIBUTES**

CONCEPT

Resource Definition

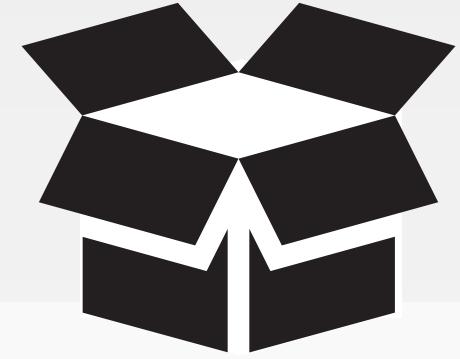


```
file 'hello.txt' do
  content 'Hello, world!'
end
```

The **TYPE** named **NAME** should be **ACTION'd** with **ATTRIBUTES**

CONCEPT

Resource Definition



```
file 'hello.txt' do
  content 'Hello, world!'
end
```

?

The **TYPE** named **NAME** should be **ACTION'd** with **ATTRIBUTES**

EXERCISE

The file resource



Read <http://docs.chef.io/chef/resources.html#file>

Discover the file resource's:

- default action
- default values for mode, owner, and group.

Update the file policy in "hello.rb" to:

The file named "hello.txt" should be created with the content "Hello, world!", mode "0644", owner is "root", and group is "root"

The updated file resource

```
~/hello.rb
```

```
file 'hello.txt' do
  content 'Hello, world!'
  mode '0644'
  owner 'root'
  group 'root'
  action :create
end
```

The default action is to create (not necessary to define it).

The default mode is "0777".

The default owner is the current user (could change).

The default group is the POSIX group (if available).

DISCUSSION

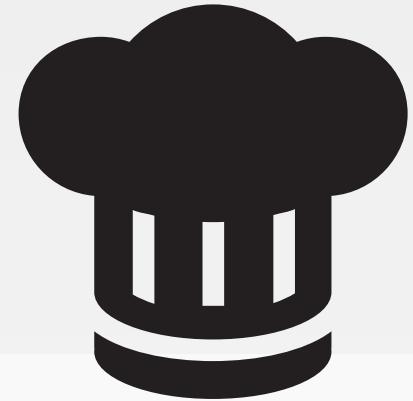
Questions



What questions can we answer for you?

LAB

Workstation Setup



Alright, it seems like I could create a recipe file to setup this workstation.

OBJECTIVE:

Create a recipe file named "setup.rb" that defines the policy:

1. Installs the \$EDITOR
2. Install the tree package
3. Setting up a customized Message of the Day (MOTD)

Installing our Editor



The package named "nano" is installed.

v3.0.0

<http://docs.chef.io/chef/resources.html#package>



Installing the tree package



The package named "tree" is installed.

v3.0.0

<http://docs.chef.io/chef/resources.html#package>



Setting up a customized MOTD



The file named "/etc/motd" is created with the content "Property of ...".

EXERCISE

Workstation Setup



Create a recipe file named "setup.rb" that defines the policy:

- The package named "nano" is installed.
- The package named "tree" is installed.
- The file named "/etc/motd" is created with the content "Property of ...".

Workstation setup recipe file

```
~/setup.rb
```

```
package 'nano'  
package 'vim'  
package 'emacs'  
  
package 'tree'  
  
file '/etc/motd' do  
  content 'Property of ...'  
  mode '0644'  
  owner 'root'  
  group 'root'  
end
```

The package named "\$EDITOR" is installed.

The package named tree is installed.

The file named "/etc/motd" is created with the content "Property of ...".

Apply the setup recipe



```
$ sudo chef-apply setup.rb
```

```
Recipe: (chef-apply cookbook)::(chef-apply recipe)
  * apt_package[vim] action install (up to date)
  * apt_package[tree] action install
    - install version 1.6.0-1 of package tree
  * file[/etc/motd] action create
    - create new file /etc/motd
    - update content in file /etc/motd from none to d100eb
      --- /etc/motd      2015-05-11 23:17:00.869570000 +0000
      +++ /etc/.motd20150511-1762-trppu1 2015-05-11 23:17:00.865570000 +0000
      @@ -1 +1,2 @@
+Property of ...
```

DISCUSSION

Let's Talk About Resources



Capture your answers because we're going to talk about them as a group.

DISCUSSION

Resources



What is a resource?

DISCUSSION

Resources



What are some other possible examples of resources?

DISCUSSION

Resources



How did the examples resources we wrote describe the desired state of an element of our infrastructure?

DISCUSSION

Resources



What does it mean for a resource to be a statement of configuration policy?

DISCUSSION

Discussion



What is a resource?

What are some other possible examples of resources?

How did the examples resources we wrote describe the desired state of an element of our infrastructure?

What does it mean for a resource to be a statement of configuration policy?

DISCUSSION

Discussion



What questions can we answer for you?

- chef-apply
- Resources
- Resource - default actions and default attributes
- Test and Repair



Cookbooks

Organizing our recipes

v3.0.0

Great First Day!

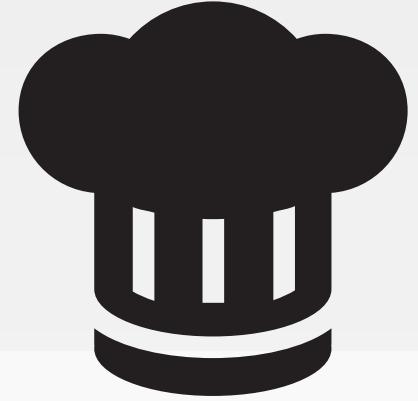


... at stand up you mentioned your workstation recipe - could you do something like that for a web server?

Also, is there a way to package up recipes you create with a version number (and maybe a README)? I think chef is able to generate something called a cookbook. Also, you really should start thinking about some version control. Don't want to lose all our hard work AGAIN ...

LAB

Versioning?



She has a point. How are we going to manage this file when I start to use it on more workstations?

OBJECTIVE:

- Choose a version control system

File Name Dot Bak

```
$ cp setup.rb setup.rb.bak
```

Save a copy of the original file as another filename.

Pros

Cons

File Name with Date and Time

```
$ cp foo{,.`date +%Y%m%d%H%M`}
```

Save a copy of the original file with the addition of the current date and time (YEARmonthDAYhourMINUTE).

Pros

Cons

File Name with Date Time and User

```
$ cp foo{,.`date +%Y%m%d%H%M`-$USER`}
```

Save a copy of the original file with the addition of the current date and time (YEARmonthDAYhourMINUTE) - current user.

Pros

Cons

How about a Wiki?

A wiki is an application, typically a web application, which allows collaborative modification, extension, or deletion of its content and structure.

Pros

Cons

How about Git?

git is a distributed revision control system with an emphasis on speed, data integrity, and support for distributed, non-linear workflows.

Pros

Cons

DISCUSSION

Version Control System



We will be using git throughout the rest of this workshop.

EXERCISE

Install git



- ❑ Add the additional policy to the setup.rb:

The package named "git" is installed.

- ❑ Then apply this recipe with chef-apply

Adding the git package

```
~/setup.rb
```

```
package "nano"  
package "vim"  
package "emacs"  
  
package "tree"  
+ package "git"  
  
file "/etc/motd" do  
  content "Property of ..."  
  mode "0644"  
  owner "root"  
  group "root"  
end
```

v3.0.0



Re-apply the setup recipe

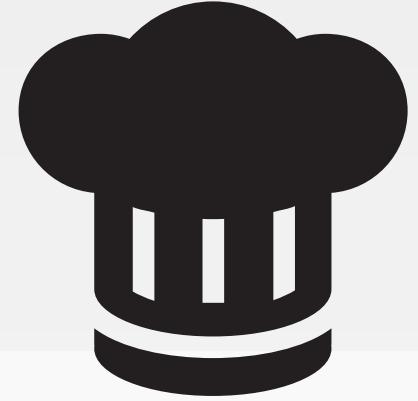


```
$ sudo chef-apply setup.rb
```

```
Recipe: (chef-apply cookbook)::(chef-apply recipe)
  * apt_package[vim] action install (up to date)
  * apt_package[tree] action install (up to date)
  * apt_package[git] action install (up to date)
  * file[/etc/motd] action create (up to date)
```

LAB

Versioning?



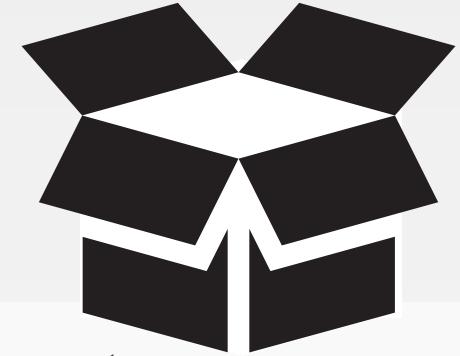
She has a point. How are we going to manage this file when I start to use it on more workstations?

OBJECTIVE:

- Use chef to generate a cookbook to store our setup recipe
- Add the "workstation" cookbook to version control

CONCEPT

What is chef?



An executable program that allows you generate cookbooks and cookbook components.

v3.0.0

 CHEF™
CODE CAN

What can chef do?



```
$ chef --help
```

Usage:

```
chef -h/--help  
chef -v/--version  
chef command [arguments...] [options...]
```

Available Commands:

exec	Runs the command in context of the embedded ruby
gem	Runs the `gem` command in context of the embedded ruby
generate	Generate a new app, cookbook, or component
shell-init	Initialize your shell to use ChefDK as your primary ruby
install	Install cookbooks from a Policyfile and generate a locked cookbook set
update	Updates a Policyfile.lock.json with latest run_list and cookbooks

v3.0.0

 CHEF™
CODE CAN

DOCS

Cookbooks



A cookbook is the fundamental unit of configuration and policy distribution. Each cookbook defines a scenario, such as everything needed to install and configure MySQL, and then it contains all of the components that are required to support that scenario . . .

What can chef generate do?



```
$ chef generate --help
```

```
Usage: chef generate GENERATOR [options]
```

```
Available generators:
```

app	Generate an application repo
cookbook	Generate a single cookbook
recipe	Generate a new recipe
attribute	Generate an attributes file
template	Generate a file template
file	Generate a cookbook file
lwrp	Generate a lightweight resource/provider
repo	Generate a Chef policy repository
policyfile	Generate a Policyfile for use with the install/push commands (experimental)

What can chef generate cookbook do?



```
$ chef generate cookbook --help
```

```
Usage: chef generate cookbook NAME [options]
```

-C, --copyright COPYRIGHT	Name of the copyright holder - defaults to 'The Authors'
-m, --email EMAIL	Email address of the author - defaults to 'you@exa...'
-a, --generator-arg KEY=VALUE	Use to set arbitrary attribute KEY to VALUE in the...
-I, --license LICENSE	all_rights, apache2, mit, gplv2, gplv3 - defaults ...
-g GENERATOR_COOKBOOK_PATH, --generator-cookbook	Use GENERATOR_COOKBOOK_PATH for the code_generator...

Lets create a cookbook!



```
$ chef generate cookbook workstation
```

```
Compiling Cookbooks...
Recipe: code_generator::cookbook
* directory[/home/chef/workstation] action create
  - create new directory /home/chef/workstation
* template[/home/chef/workstation/metadata.rb] action create_if_missing
  - create new file /home/chef/workstation/metadata.rb
  - update content in file /home/chef/workstation/metadata.rb from none to bd85d3
    (diff output suppressed by config)
* template[/home/chef/workstation/README.md] action create_if_missing
  - create new file /home/chef/workstation/README.md
  - update content in file /home/chef/workstation/README.md from none to 44d165
    (diff output suppressed by config)
* cookbook_file[/home/chef/workstation/chefignore] action create
```

v3.0.0

 CHEF™
CODE CAN

The cookbook has a README



```
$ tree workstation
```

```
workstation
├── Berksfile
├── README.md
├── chefignore
├── metadata.rb
├── README.md
└── recipes
    └── default.rb
└── spec
    ├── spec_helper.rb
    └── unit
        └── recipes
10 directories, 9 files
```

CONCEPT

README.md



The description of the cookbook's features written in Markdown.

v3.0.0

<http://daringfireball.net/projects/markdown/syntax>

 **CHEF**™
CODE CAN

The cookbook has some metadata



```
$ tree workstation
```

```
workstation
├── Berksfile
├── README.md
├── chefignore
└── metadata.rb
├── README.md
└── recipes
    └── default.rb
└── spec
    ├── spec_helper.rb
    └── unit
        └── recipes
10 directories, 9 files
```

DOCS

metadata.rb



Every cookbook requires a small amount of metadata. Metadata is stored in a file called `metadata.rb` that lives at the top of each cookbook's directory.

Lets take a look at the metadata



```
$ cat workstation/metadata.rb
```

```
name          'workstation'  
maintainer    'The Authors'  
maintainer_email 'you@example.com'  
license        'all_rights'  
description    'Installs/Configures workstation'  
long_description 'Installs/Configures workstation'  
version        '0.1.0'
```

The cookbook has a folder for recipes



```
$ tree workstation
```

```
workstation
├── Berksfile
├── README.md
├── chefignore
├── metadata.rb
├── README.md
└── recipes
    ├── default.rb
    └── spec
        ├── spec_helper.rb
        └── unit
            └── recipes
10 directories, 9 files
```

The cookbook has a default recipe



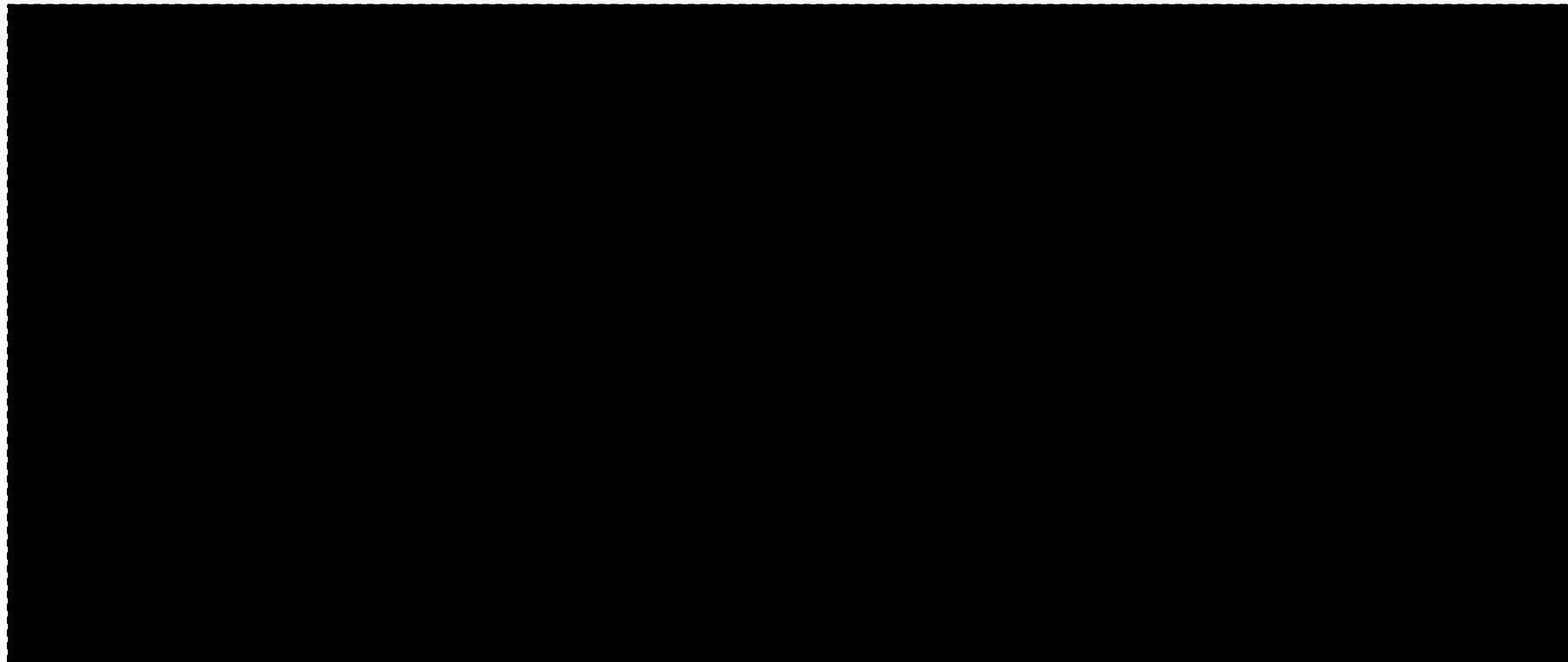
```
$ cat workstation/recipes/default.rb
```

```
# Cookbook Name:: workstation
# Recipe:: default
#
# Copyright (c) 2015 The Authors, All Rights Reserved.
```

Copy the recipe into the cookbook



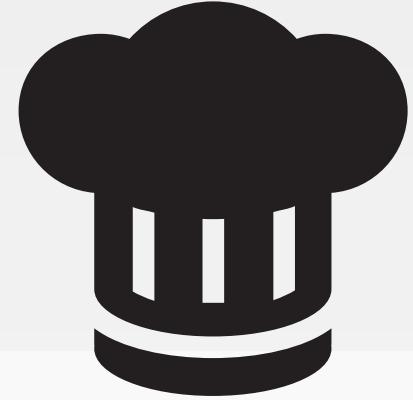
```
$ mv setup.rb workstation/recipes/setup.rb
```



v3.0.0

LAB

Version Control



This is probably a good point to capture the initial state of our cookbook.

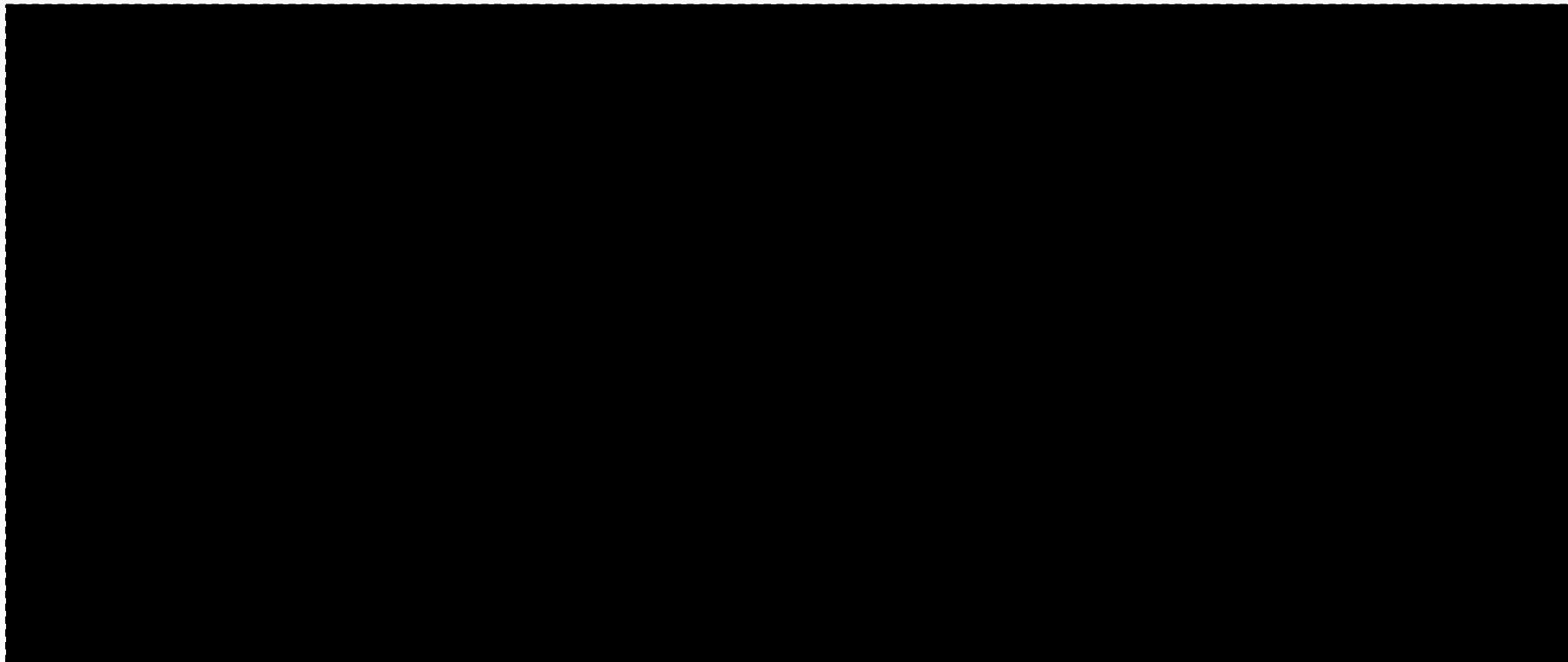
OBJECTIVE:

- ✓ Use chef to generate a cookbook to store our setup recipe
- ❑ Add the "workstation" cookbook to version control

Move into the cookbook directory



```
$ cd workstation
```



v3.0.0

Initialize it as a git repository



```
$ git init
```

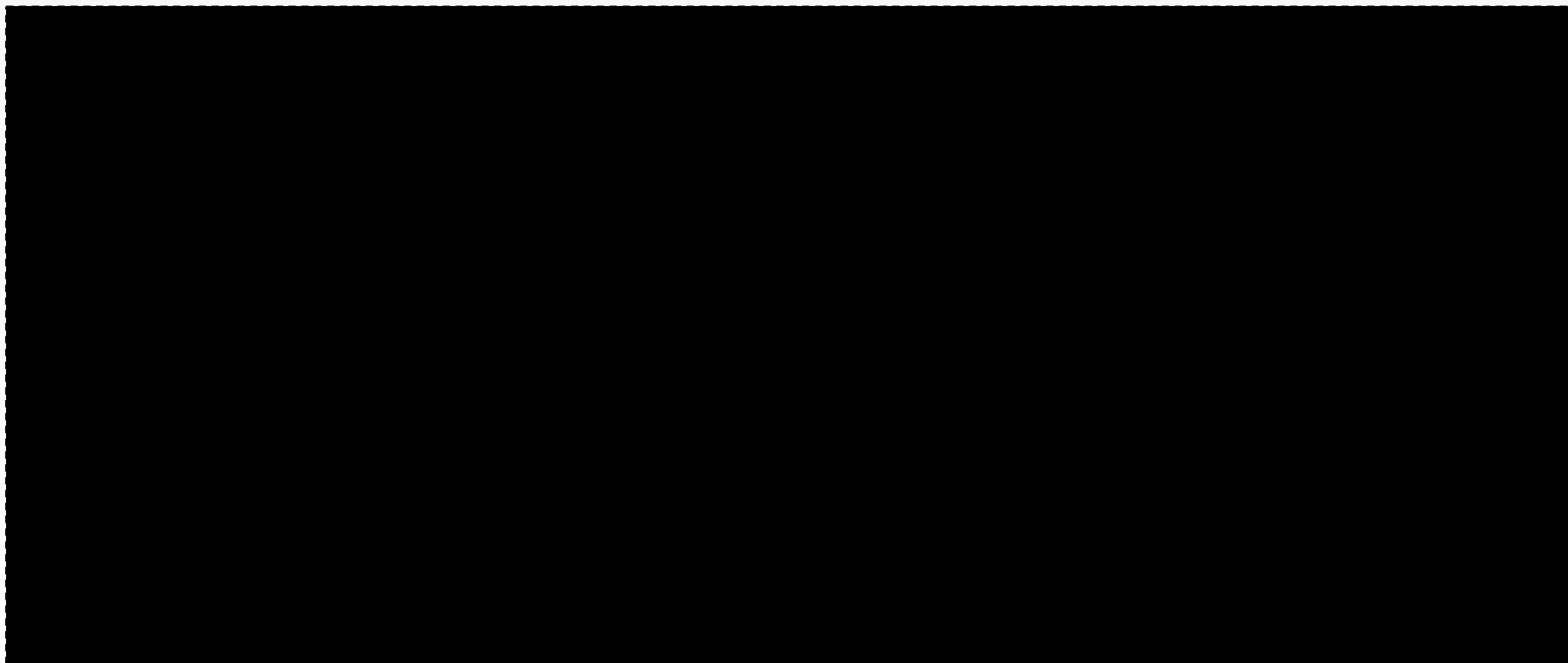
```
Reinitialized existing Git repository in /home/chef/workstation/.git/
```

v3.0.0

Use git add to stage files to be committed.



```
$ git add .
```



v3.0.0

CONCEPT

Staging Area



The staging area is a file, generally contained in your Git directory, that stores information about what will go into your next commit. It's sometimes referred to as the “index”, but it's also common to refer to it as the staging area.

Use git status to view the staged files



```
$ git status
```

```
On branch master
```

```
Initial commit
```

```
Changes to be committed:
```

```
(use "git rm --cached <file>..." to unstage)
```

```
new file: .gitignore
new file: .kitchen.yml
new file: Berksfile
new file: README.md
new file: chefignore
new file: metadata.rb
```

Use git commit to save the staged changes



```
$ git commit -m "Initial workstation cookbook"
```

```
master (root-commit) 9998472] Initial workstation cookbook
Committer: ChefDK User <chef@ip-172-31-59-191.ec2.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:
```

```
git config --global user.name "Your Name"
git config --global user.email you@example.com
```

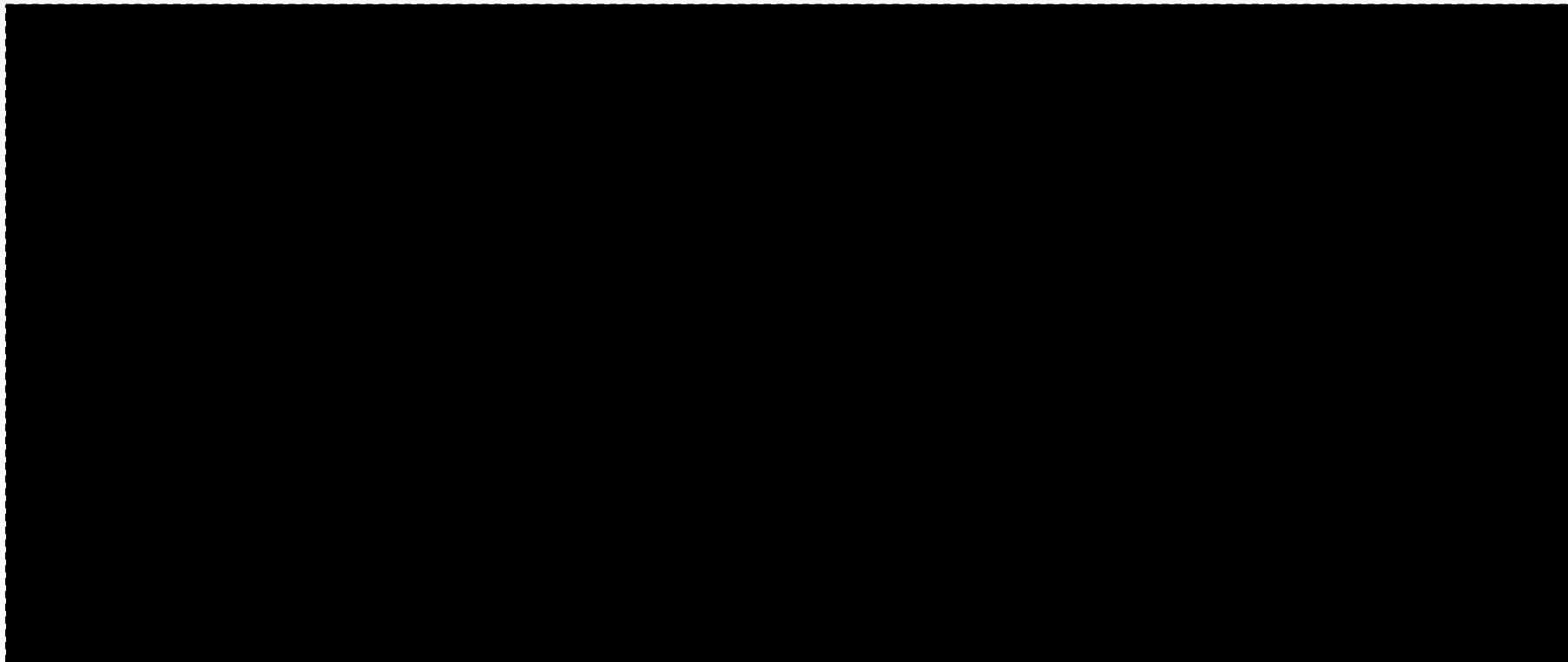
After doing this, you may fix the identity used for this commit with:

```
git commit --amend --reset-author
```

Git wants an Email



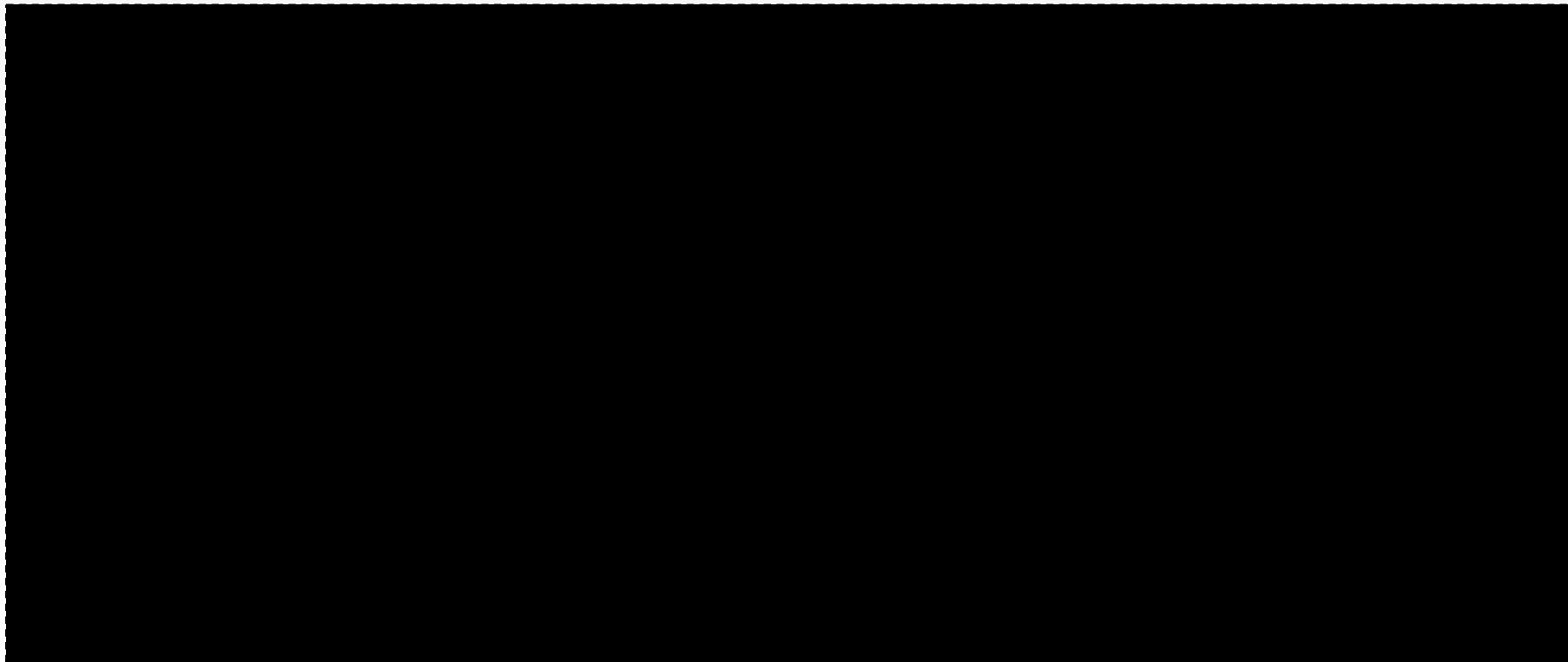
```
$ git config --global user.email "you@example.com"
```



Git wants a Username



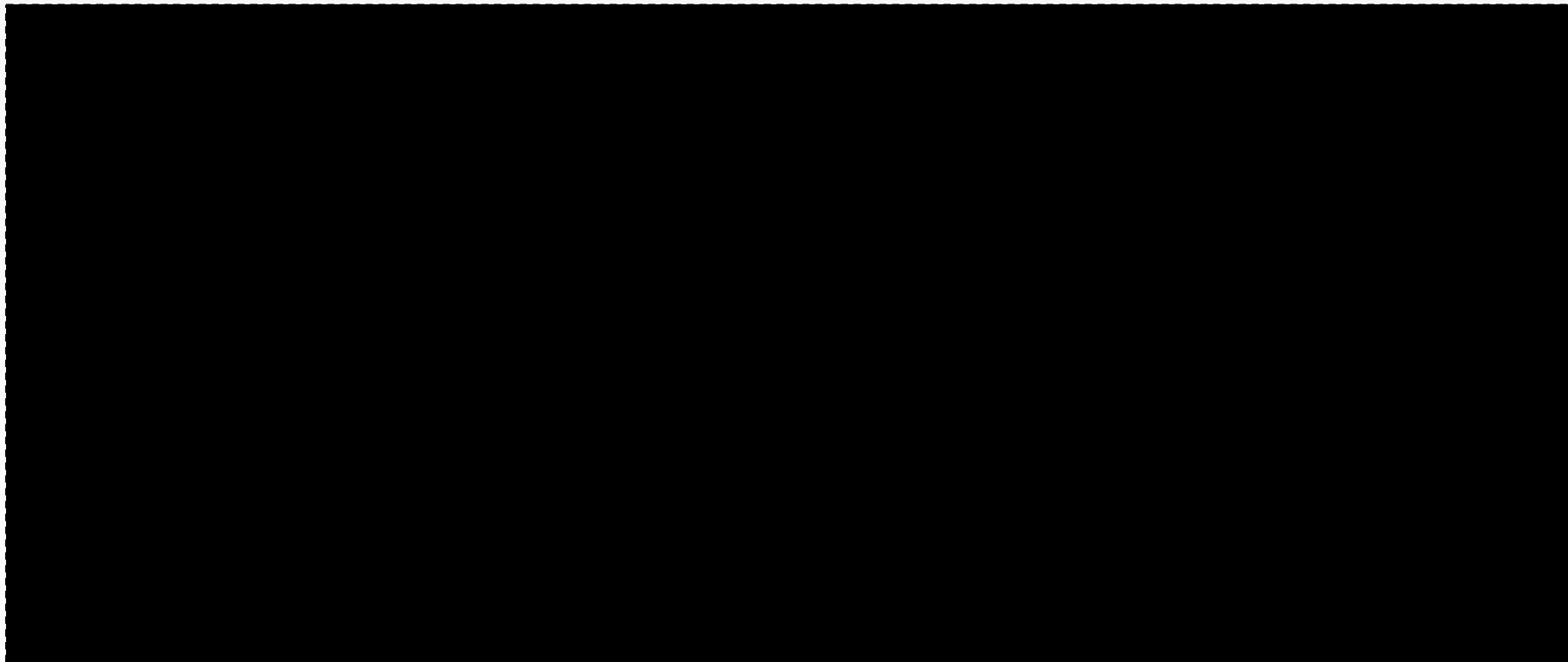
```
$ git config --global user.name "username"
```



Move out of the workstation cookbook



```
$ cd ~
```



v3.0.0

EXERCISE

Setting up a Web Server



- ❑ Use chef generate to create a cookbook named "apache".
- ❑ Write and apply a recipe named "server.rb" with the policy:

The package named "httpd" is installed.

The file named "/var/www/html/index.html" is created with the content "<h1>Hello, world!</h1>"

The service named "httpd" is started.

The service named "httpd" is enabled.

- ❑ Place the apache cookbook under version control

Lets create a cookbook!



```
$ chef generate cookbook apache
```

```
Compiling Cookbooks...
Recipe: code_generator::cookbook
  * directory[/home/chef/apache] action create
    - create new directory /home/chef
  * template[/home/chef/apache/metadata.rb] action create_if_missing
    - create new file /home/chef/apache/metadata.rb
    - update content in file /home/chef/apache/metadata.rb from none to bd85d3
      (diff output suppressed by config)
  * template[/home/chef/apache/README.md] action create_if_missing
    - create new file /home/chef/apache/README.md
    - update content in file /home/chef/apache/README.md from none to 44d165
      (diff output suppressed by config)
  * cookbook_file[/home/chef/apache/chefignore] action create
```

Apache Recipe

~/apache/recipes/server.rb

```
package "httpd"

file "/var/www/html/index.html" do
  content "<h1>Hello, world!</h1>"
end

service "httpd" do
  action :enable
end
service "httpd" do
  action :start
end
```

v3.0.0

 CHEF™
CODE CAN

Apache Recipe

```
~/apache/recipes/server.rb
```

```
package "httpd"

file "/var/www/html/index.html" do
  content "<h1>Hello, world!</h1>"
end

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

v3.0.0



Apply the server recipe



```
$ sudo chef-apply apache/recipes/server.rb
```

```
Recipe: (chef-apply cookbook)::(chef-apply recipe)
  * apt_package[apache2] action install
    - install version 2.4.7-1ubuntu4.4 of package apache2
  * file[/var/www/html/index.html] action create
    - update content in file /var/www/html/index.html from 538f31 to 17d291
      --- /var/www/html/index.html      2015-05-05 08:33:24.681723000 +0000
      +++ /var/www/html/.index.html20150505-2060-171d524 2015-05-05 08:33:31.989375000 +0000
      @@ -1,379 +1,2 @@
      -
      -<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
        -<html xmlns="http://www.w3.org/1999/xhtml">
          -  <!--
```

Verify the website is available



```
$ curl localhost
```

```
<h1>Hello, world!</h1>
```

COMMIT

Commit Your Work



```
$ cd apache  
$ git init  
$ git add .  
$ git commit -m "Initial Apache Cookbook"
```

DISCUSSION

Discussion



What questions can we answer for you?

- cookbooks
- versions
- version control



chef-client

Applying recipes from cookbooks

v3.0.0





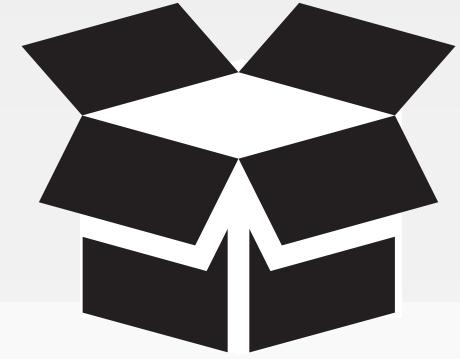
chef-apply

chef-apply is a great tool for applying resources (-e) and for individual recipes but it does not know how to apply a cookbook. This is why we need to specify the path to the recipe file.

A better tool for applying cookbooks is called chef-client.

CONCEPT

chef-client



A chef-client is an agent that runs locally on every node that is under management by Chef. When a chef-client is run, it will perform all of the steps that are required to bring the node into the expected state, including...

Using chef-client to locally apply recipes

```
$ sudo chef-client --local-mode -r "recipe[workstation::setup]"
```

Apply the following recipes locally:

- The 'setup' recipe from the 'workstation' cookbook

Using chef-client to locally apply recipes

```
$ sudo chef-client --local-mode -r "recipe[apache::server]"
```

Apply the following recipes locally:

- The 'server' recipe from the 'apache' cookbook

Using chef-client to locally apply recipes

```
$ sudo chef-client --local-mode -r \
"recipe[workstation::setup],recipe[apache::server]"
```

Apply the following recipes locally:

- The 'setup' recipe from the 'workstation' cookbook
- The 'server' recipe from the 'apache' cookbook

CONCEPT

--local-mode



chef-client's default mode attempts to contact a Chef Server and ask it for the recipes to run for the given node. We are overriding that behavior to have work in a local mode.

CONCEPT

`-r "recipe[COOKBOOK::RECIPE]"`



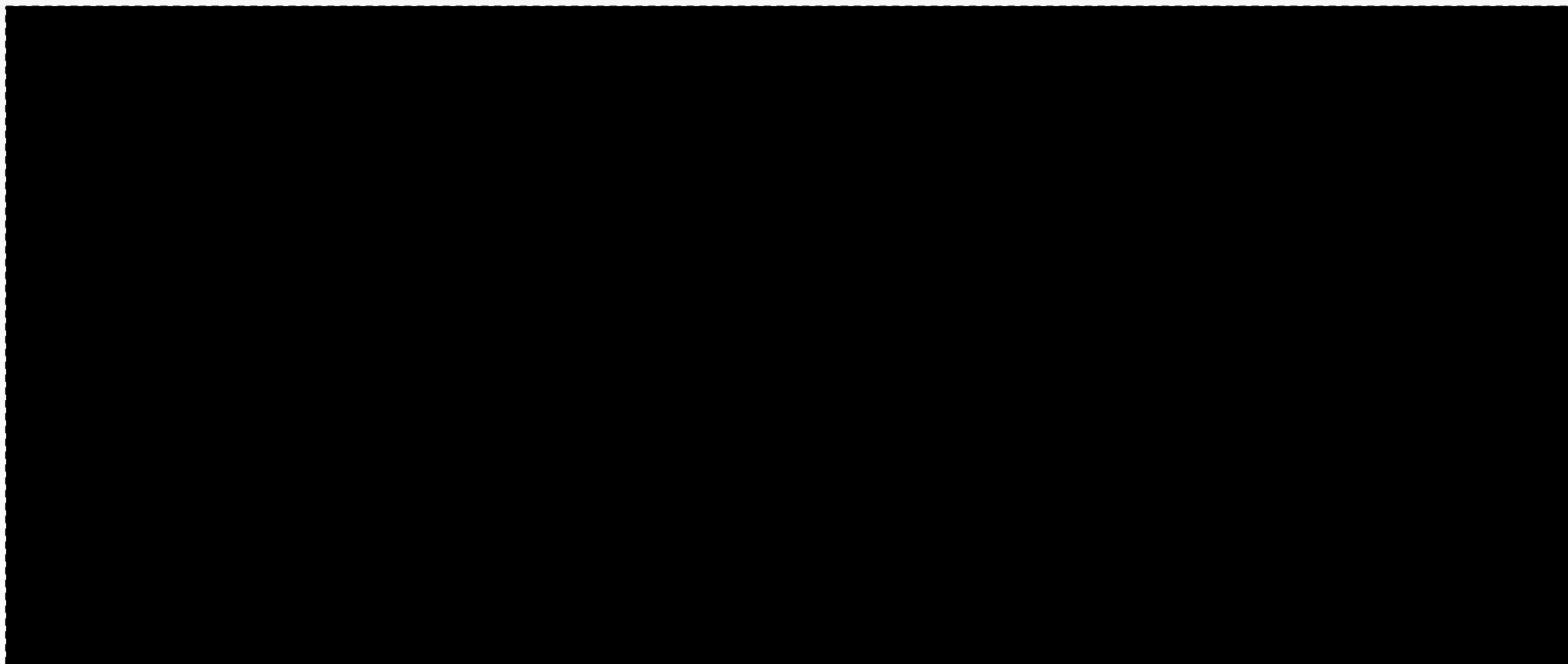
In local mode we need to provide a list of recipes to apply to the system. This is called a **run list**. A run list is an ordered collection of recipes to execute.

Each recipe in the run list must be addressed with the format `recipe[COOKBOOK::RECIPE]`.

Return home first



```
$ cd ~
```



v3.0.0

Applying the apache::server recipe locally



```
$ sudo chef-client --local-mode -r "recipe[apache::server]"
```

```
[2015-03-29T21:38:56-07:00] WARN: No config file found or specified on command line, using  
command line options.
```

```
Starting Chef Client, version 11.16.4
```

```
resolving cookbooks for run list: ["apache::server"]
```

```
=====
Error Resolving Cookbooks for Run List:
```

```
=====
Missing Cookbooks:
```

Create a cookbooks directory



```
$ mkdir cookbooks
```

v3.0.0

Move the workstation cookbook



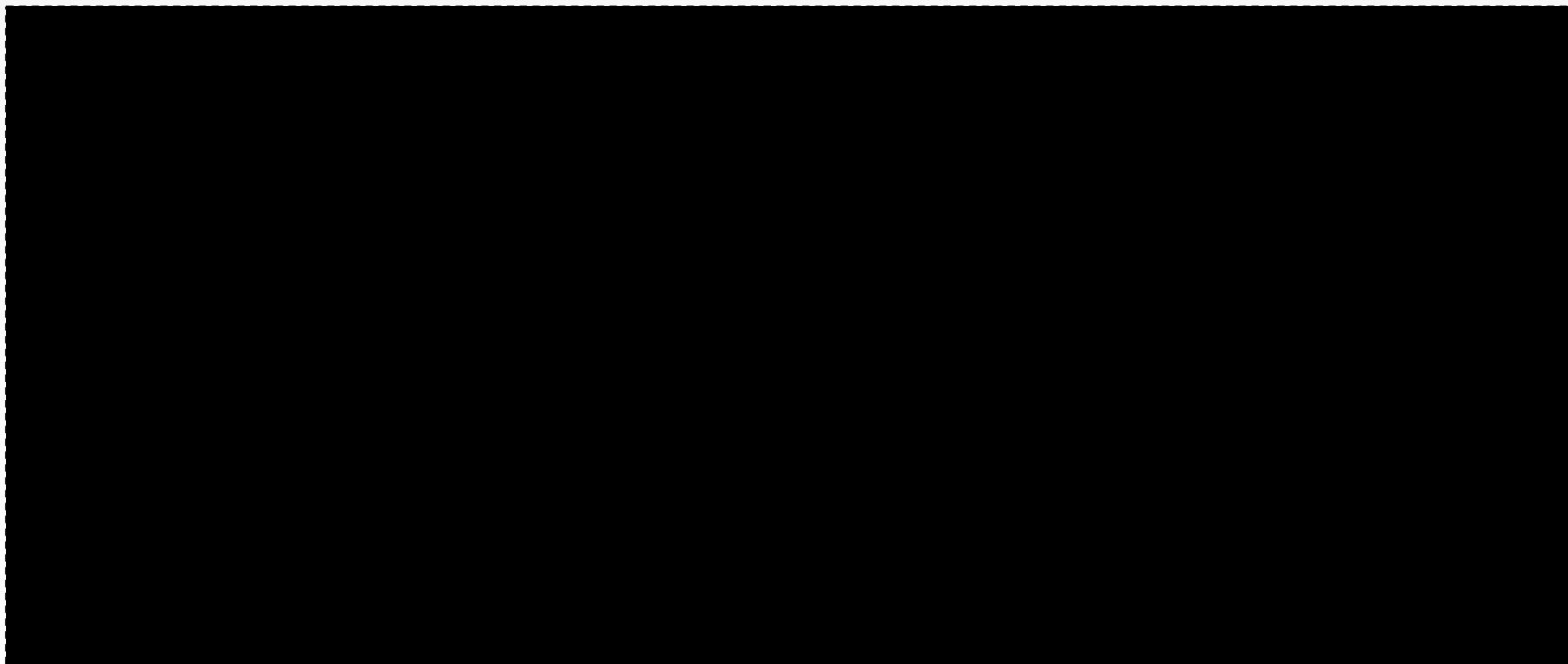
```
$ mv workstation cookbooks
```

v3.0.0

Move the apache cookbook



```
$ mv apache cookbooks
```



v3.0.0

Applying the cookbook recipe locally



```
$ sudo chef-client --local-mode -r "recipe[apache::server]"
```

```
[2015-03-29T21:38:08-07:00] WARN: No config file found or specified on command line, using command line options.
```

```
Starting Chef Client, version 11.16.4
```

```
resolving cookbooks for run list: ["apache::server"]
```

```
Synchronizing Cookbooks:
```

```
  - apache
```

```
Compiling Cookbooks...
```

```
Converging 3 resources
```

```
Recipe: apache::server
```

```
...
```

Applying the cookbook recipe locally



```
$ sudo chef-client --local-mode -r "recipe[workstation::setup]"
```

```
[2015-03-29T21:38:08-07:00] WARN: No config file found or specified on command line, using  
command line options.
```

```
Starting Chef Client, version 11.16.4
```

```
resolving cookbooks for run list: ["workstation::setup"]
```

```
Synchronizing Cookbooks:
```

```
  - workstation
```

```
Compiling Cookbooks...
```

```
Converging 6 resources
```

```
Recipe: workstation::setup
```

```
...
```

Applying both recipes locally



```
$ sudo chef-client --local-mode \
-r "recipe[apache::server],recipe[workstation::setup]"
```

```
[2015-03-29T21:38:08-07:00] WARN: No config file found or specified on command line, using
command line options.

Starting Chef Client, version 11.16.4
resolving cookbooks for run list: ["apache::server", "workstation::setup"]
Synchronizing Cookbooks:
  - apache
  - workstation

Compiling Cookbooks...
Converging 9 resources
Recipe: apache::server
...
...
```

CONCEPT

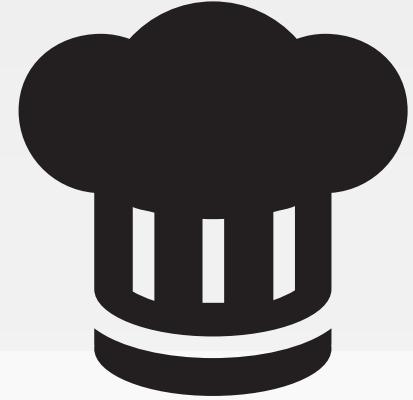
`-r "recipe[COOKBOOK(: :default)]"`



When you are referencing the default recipe within a cookbook you may optionally specify only the name of the cookbook. chef-client understands that you mean to apply the default recipe from within that cookbook.

LAB

Setting a default in our cookbook



*It seems silly to type "recipe[workstation::setup]".
Typing out "recipe[workstation]" also seems clearer.*

OBJECTIVE:

- Update the default recipe to use `include_recipe` to include the setup recipe.
- Run chef-client and locally apply the run_list: `"recipe[workstation]"`

DOCS

include_recipe



A recipe can include one (or more) recipes located in cookbooks by using the `include_recipe` method. When a recipe is included, the resources found in that recipe will be inserted (in the same exact order) at the point where the `include_recipe` keyword is located.

Including a recipe

```
include_recipe "workstation::setup"
```

Include the "setup" recipe from the "workstation" cookbook in this recipe

Including a recipe

```
include_recipe "apache::server"
```

Include the "server" recipe from the "apache" cookbook in this recipe

The default recipe includes the setup recipe

```
~/cookbooks/workstation/recipes/default.rb
```

```
#  
# Cookbook Name:: workstation  
# Recipe:: default  
#  
# Copyright (c) 2015 The Authors, All Rights Reserved.
```

```
include_recipe "workstation::setup"
```

Applying cookbook's default recipe



```
$ sudo chef-client --local-mode -r "recipe[workstation]"
```

```
[2015-03-29T21:38:08-07:00] WARN: No config file found or specified on command line, using  
command line options.
```

```
Starting Chef Client, version 11.16.4
```

```
resolving cookbooks for run list: ["workstation"]
```

```
Synchronizing Cookbooks:
```

```
  - workstation
```

```
Compiling Cookbooks...
```

```
Converging 6 resources
```

```
Recipe: workstation::default
```

```
...
```

EXERCISE

Update the apache cookbook



- ❑ Update the "apache" cookbook's "default" recipe to:

Include the "server" recipe from the "apache" cookbook

- ❑ Run chef-client and locally apply the run_list:
"recipe[apache]"

The default recipe includes the apache recipe

```
cookbooks/apache/recipes/default.rb
```

```
#  
# Cookbook Name:: apache  
# Recipe:: default  
  
#  
# Copyright (c) 2015 The Authors, All Rights Reserved.
```

```
include_recipe "apache::server"
```

Applying apache's default recipe



```
$ sudo chef-client --local-mode -r "recipe[apache]"
```

```
[2015-03-29T21:38:08-07:00] WARN: No config file found or specified on command line, using command line options.
```

```
Starting Chef Client, version 11.16.4
```

```
resolving cookbooks for run list: ["apache"]
```

```
Synchronizing Cookbooks:
```

```
  - apache
```

```
Compiling Cookbooks...
```

```
Converging 3 resources
```

```
Recipe: apache::default
```

```
...
```

DISCUSSION

Discussion



What questions can we help you answer?

- chef-client
- local mode
- run list
- include_recipe



Testing Cookbooks

Validating our recipes in virtual environments

v3.0.0



Can We Test Cookbooks?



I'm going to miss our 1-on-1 this week because I was up way too late dealing with a deployment issue for that new event-queuing system. Apparently, no one had actually tested the deployment scripts on the actual platforms in production.

Jennifer and I did a quick post mortem and she mentioned that we can verify cookbooks with Test Kitchen and Docker. Is that something you could check out?

DISCUSSION

Mandating Testing



What steps would it take to test one of the cookbooks that we have created?

EXERCISE

Time for Testing



Estimate how long it would take to accomplish testing the cookbook.

DISCUSSION

Testing Interval



How often do you test your cookbook?

DISCUSSION

What's the Risk?



How often do you think changes will occur?

DISCUSSION

Is this dangerous?



What happens when the rate of cookbook changes exceed the time interval it takes to verify the cookbook?

CONCEPT

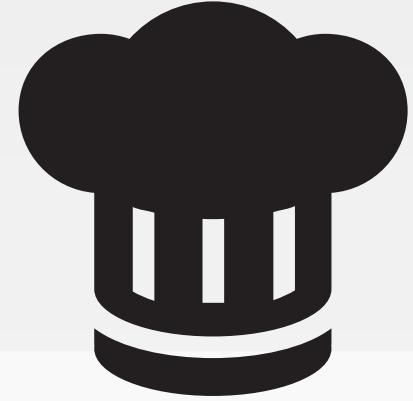
Code Testing



Automated way to ensure code accomplishes the intended goal and help the team understand its intent

LAB

Test Configuration



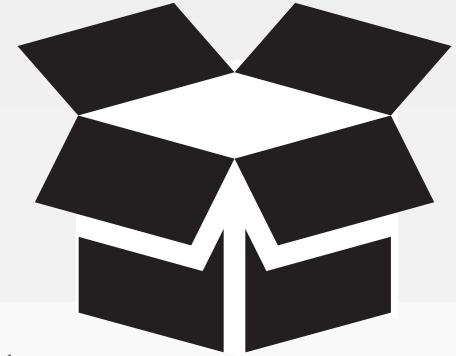
What are we running in production? Maybe I could test the cookbook against a virtual machine.

OBJECTIVE:

- Configure the "workstation" cookbook's .kitchen.yml to use the Docker driver and Ubuntu 14.04 platform
- Use kitchen converge to apply the recipe on a virtual machine

CONCEPT

Test Kitchen



Test Kitchen is a test harness tool to execute your configured code on one or more platforms in isolation. A driver plugin architecture is used which lets you run your code on various cloud providers and virtualization technologies such as . . .

What can kitchen do?



```
$ kitchen --help
```

Commands:

```
kitchen console                      # Kitchen Console!
kitchen converge [INSTANCE|REGEXP|all]   # Converge one or more instances
kitchen create [INSTANCE|REGEXP|all]      # Create one or more instances
kitchen destroy [INSTANCE|REGEXP|all]     # Destroy one or more instances
...
kitchen help [COMMAND]                  # Describe available commands or one specif...
kitchen init                           # Adds some configuration to your cookbook...
kitchen list [INSTANCE|REGEXP|all]       # Lists one or more instances
kitchen setup [INSTANCE|REGEXP|all]       # Setup one or more instances
kitchen test [INSTANCE|REGEXP|all]        # Test one or more instances
kitchen verify [INSTANCE|REGEXP|all]      # Verify one or more instances
kitchen version                        # Print Kitchen's version information
```

v3.0.0

 CHEF™
CODE CAN

What can kitchen init do?



```
$ kitchen help init
```

Usage:

```
kitchen init  
-D, [--driver=one two three]          # One or more Kitchen Driver gems ...  
                                      # Default: kitchen-vagrant  
-P, [--provisioner=PROVISIONER]        # The default Kitchen Provisioner to use  
                                      # Default: chef_solo  
[--create-gemfile], [--no-create-gemfile] # Whether or not to create a Gemfi ...
```

Description:

Init will add Test Kitchen support to an existing project for convergence integration testing. A default `.kitchen.yml` file (which is intended to be customized) is created in the project's root directory and one or more gems will be added to the project's Gemfile.

Do we have a .kitchen.yml?



```
$ tree cookbooks/workstation -a -I .git
```

```
workstation
├── Berksfile
├── chefignore
├── .gitignore
└── .kitchen.yml
├── metadata.rb
├── README.md
├── recipes
│   ├── default.rb
│   └── setup.rb
└── spec
    ├── spec_helper.rb
    └── unit
```

What is inside.kitchen.yml?



```
$ cat cookbooks/workstation/.kitchen.yml
```

```
---
```

```
driver:
```

```
  name: vagrant
```

```
provisioner:
```

```
  name: chef_solo
```

```
platforms:
```

```
  - name: ubuntu-12.04
```

```
  - name: centos-6.4
```

```
suites:
```

```
  - name: default
```

CONCEPT

.kitchen.yml



When chef generates a cookbook a default .kitchen.yml is created. It contains kitchen configuration for the driver, provisioner, platform, and suites.

The kitchen driver



~/cookbooks/workstation/.kitchen.yml

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-12.04
  - name: centos-6.5

...
```

The driver is responsible for creating a machine that we'll use to test our cookbook.

Example Drivers:

- docker
- vagrant

The kitchen provisioner



~/cookbooks/workstation/.kitchen.yml

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-12.04
  - name: centos-6.5

...
```

This tells Test Kitchen how to run Chef, to apply the code in our cookbook to the machine under test.

The default and simplest approach is to use `chef_zero`.

The kitchen platforms



```
~/cookbooks/workstation/.kitchen.yml
```

```
---
```

```
driver:
```

```
  name: vagrant
```



```
provisioner:
```

```
  name: chef_zero
```



```
platforms:
```

```
  - name: ubuntu-12.04
```

```
  - name: centos-6.5
```



```
...
```

This is a list of operation systems on which we want to run our code.

The kitchen suites



~/cookbooks/workstation/.kitchen.yml

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

This section defines what we want to test. It includes the Chef run-list of recipes that we want to test.

We define a single suite named "default".

The kitchen suites



~/cookbooks/workstation/.kitchen.yml

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

The suite named "default" defines a `run_list`.

Run the "workstation" cookbook's "default" recipe file.

CONCEPT

Kitchen Test Matrix



Kitchen defines a list of instances, or test matrix, based on the platforms multiplied by the suites.

PLATFORMS x SUITES

Running kitchen list will show that matrix.

Example: Kitchen Test Matrix



```
$ kitchen list
```

Instance	Driver	Provisioner	Last Action
default-ubuntu-1204	Vagrant	ChefZero	<Not Created>
default-centos-65	Vagrant	ChefZero	<Not Created>

suites:

```
- name: default
  run_list:
    - recipe[workstation::default]
  attributes:
```

platforms:

```
- name: ubuntu-12.04
- name: centos-6.5
```

View the Kitchen Test Matrix



```
$ kitchen list
```

Instance	Driver	Provisioner	Last Action
default-ubuntu-1204	Vagrant	ChefZero	<Not Created>
default-centos-65	Vagrant	ChefZero	<Not Created>

suites:

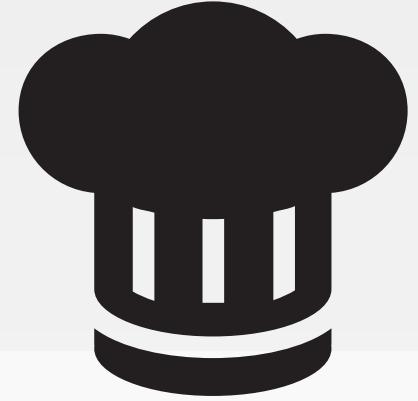
```
- name: default
  run_list:
    - recipe[workstation::default]
  attributes:
```

platforms:

```
- name: ubuntu-12.04
- name: centos-6.5
```

LAB

Test Configuration



What are we running in production? Maybe I could test the cookbook against a virtual machine.

OBJECTIVE:

- Configure the "workstation" cookbook's .kitchen.yml to use the Docker driver and Ubuntu 14.04 platform
- Use kitchen converge to apply the recipe on a virtual machine

Move into the Cookbook's Directory



```
$ cd cookbooks/workstation
```

Setting the driver to Docker

```
~/cookbooks/workstation/.kitchen.yml
```

```
---
driver:
  name: docker

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-14.04

suites:
  - name: default
    run_list:
```



v3.0.0

<https://github.com/portertech/kitchen-docker>

 CHEF™
CODE CAN

Setting the platform to Ubuntu 14.04



~/cookbooks/workstation/.kitchen.yml

```
---
driver:
  name: docker

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-14.04

suites:
  - name: default
    run_list:
```



v3.0.0

<https://github.com/portertech/kitchen-docker>

CHEF™
CODE CAN

Look at the Test Matrix



```
$ kitchen list
```

Instance	Driver	Provisioner	Verifier	Transport	Last Action
default-ubuntu-1404	Docker	ChefZero	Busser	Ssh	<Not Created>

LAB

Converging a Cookbook



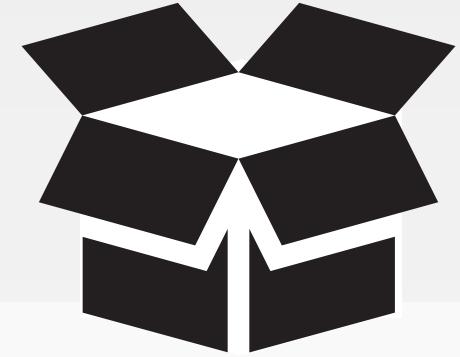
Before I add features it really would be nice to test these cookbooks against the environments that resemble production.

OBJECTIVE:

- ✓ Configure the "workstation" cookbook's .kitchen.yml to use the Docker driver and Ubuntu 14.04 platform
- ❑ Use kitchen converge to apply the recipe on a virtual machine

CONCEPT

Kitchen Create



kitchen create

kitchen
converge

kitchen verify

```
$ kitchen create [INSTANCE|REGEXP|all]
```

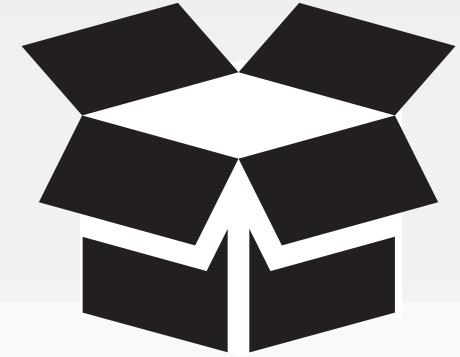
Create one or more instances.

v3.0.0

 CHEF™
CODE CAN

CONCEPT

Kitchen Converge



```
$ kitchen converge [INSTANCE|REGEXP|all]
```

Create the instance (if necessary) and then apply the run list to one or more instances.

v3.0.0

 CHEF™
CODE CAN

Converge the Cookbook



```
$ kitchen converge
```

```
----> Starting Kitchen (v1.4.0)
----> Converging <default-ubuntu-1404>...
$$$$$ Running legacy converge for 'Docker' Driver
      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 curl...
      Download complete.
```

EXERCISE

Converge the Recipe



- We want to validate that our run-list installs correctly.
- Within the "apache" cookbook use kitchen converge for the default suite on the CentOS 6.6 platform.

Configuring Test Kitchen for Apache

```
~/cookbooks/apache/.kitchen.yml
```

```
---
```

```
driver:
```

```
  name: docker
```

```
provisioner:
```

```
  name: chef_zero
```

```
platforms:
```

```
  - name: ubuntu-14.04
```

```
suites:
```

```
  - name: default
```

```
    run_list:
```

v3.0.0

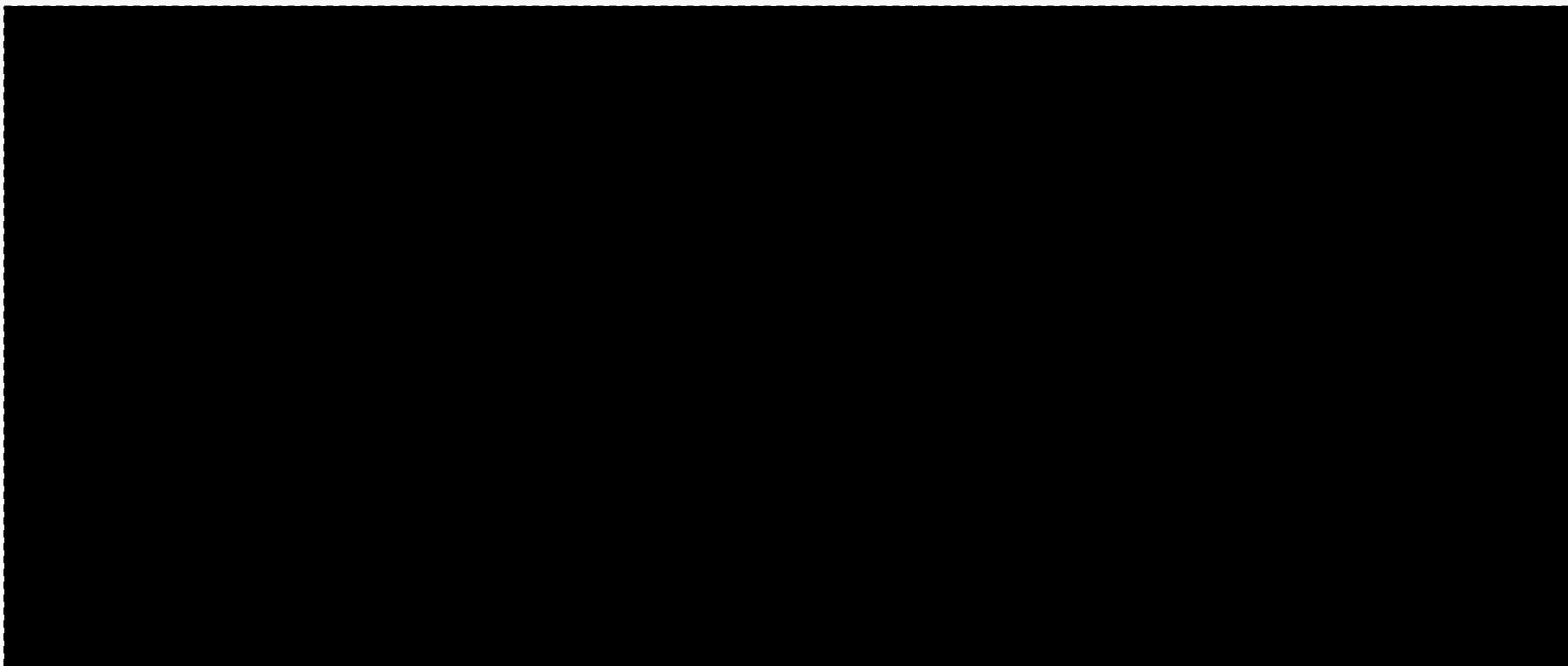
<https://github.com/portertech/kitchen-docker>



Return home



```
$ cd ~
```

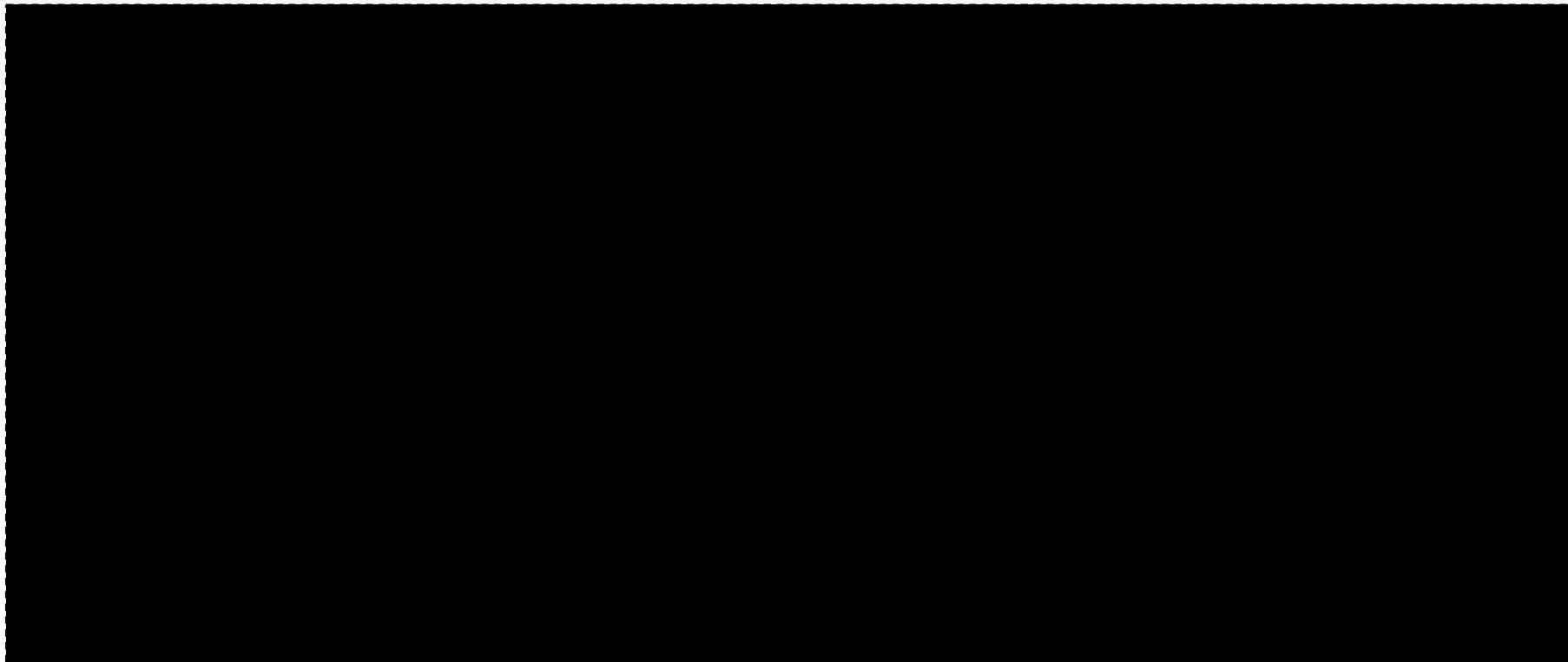


v3.0.0

Move into the Cookbook



```
$ cd cookbooks/apache
```



v3.0.0

Converge the Cookbook



```
$ kitchen converge
```

```
----> Starting Kitchen (v1.4.0)
----> Converging <default-ubuntu-1404>...
$$$$$ Running legacy converge for 'Docker' Driver
      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 curl...
      Download complete.
```

DISCUSSION

Test Kitchen



What does this test when kitchen converges a recipe?

DISCUSSION

Test Kitchen



What does it NOT test when kitchen converges a recipe?

DISCUSSION

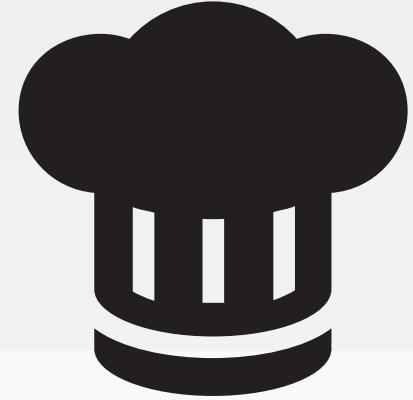
Test Kitchen



What is left to validate to ensure that the cookbook successfully applied the policy defined in the recipe?

LAB

The First Test



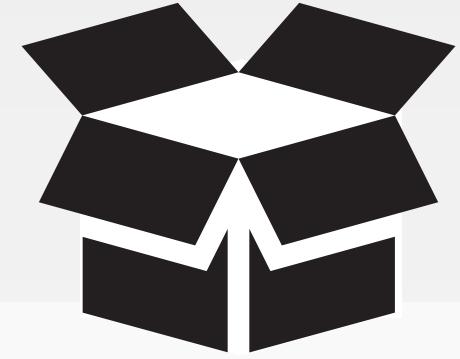
Converging seems to validate that the recipe runs successfully. But does it assert what actually is installed?

OBJECTIVE:

- Write and execute a test that asserts that the tree package is installed when the "workstation" cookbook's default recipe is applied.

CONCEPT

Kitchen Verify



kitchen create

kitchen converge

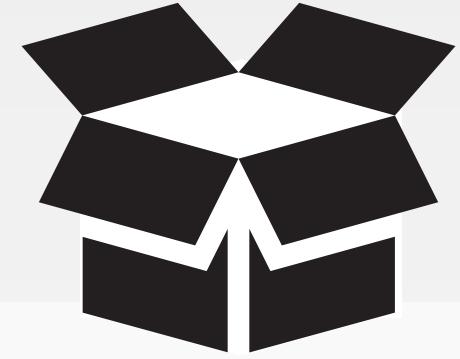
kitchen verify

```
$ kitchen verify [INSTANCE|REGEXP|all]
```

Create, converge, and verify one or more instances.

CONCEPT

Kitchen Destroy



```
$ kitchen destroy [INSTANCE|REGEXP|all]
```

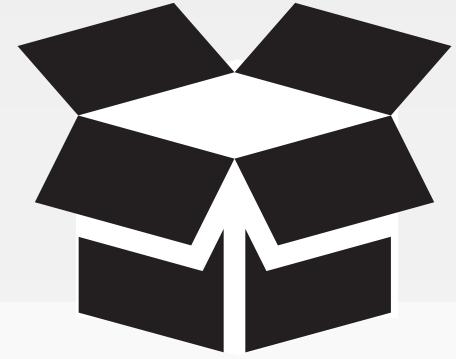
Destroys one or more instances.

v3.0.0

 CHEF™
CODE CAN

CONCEPT

Kitchen Test



```
$ kitchen test [INSTANCE|REGEXP|all]
```

Destroys (for clean-up), creates, converges, verifies and then destroys one or more instances.

v3.0.0

 CHEF™
CODE CAN

CONCEPT

ServerSpec



Serverspec tests your servers' actual state by executing command locally, via SSH, via WinRM, via Docker API and so on. So you don't need to install any agent softwares on your servers and can use any configuration management tools, Puppet, Chef, CFEngine, Itamae and so on.

v3.0.0

<http://serverspec.org>

 CHEF™
CODE CAN

Is the tree package installed?

```
describe package('tree') do
  it { should be_installed }
end
```

I expect the package tree should be installed.

Our assertion in a spec file

```
~/cookbooks/workstation/test/integration/default/serverspec/default_spec.rb
```

```
require "spec_helper"

describe "workstation::default" do

  describe package('tree') do
    it { should be_installed }
  end

end
```

Loads a helper file with that name in the same directory.

Our assertion in a spec file

```
~/cookbooks/workstation/test/integration/default/serverspec/default_spec.rb
```

```
require "spec_helper"

describe "workstation::default" do

  describe package('tree') do
    it { should be_installed }
  end

end
```

Describing a body of tests for the "workstation" cookbook's default recipe.

v3.0.0

http://serverspec.org/resource_types.html#package



Our assertion in a spec file

```
~/cookbooks/workstation/test/integration/default/serverspec/default_spec.rb
```

```
require "spec_helper"

describe "workstation::default" do

  describe package('tree') do
    it { should be_installed }
  end

end
```

When we converge the workstation cookbook's default recipe we expect the package named tree to be installed.

CONCEPT

Where do Tests Live?



workstation/test/integration/default/serverspec/default_spec.rb

Test Kitchen will look for tests to run under this directory. It allows you to put unit or other tests in test/unit, spec, acceptance, or wherever without mixing them up. This is configurable, if desired.

v3.0.0

<http://kitchen.ci/docs/getting-started/writing-test>

 CHEF™
CODE CAN

CONCEPT

Where do Tests Live?



workstation/test/integration/**default**/serverspec/default_spec.rb

This corresponds exactly to the Suite name we set up in the .kitchen.yml file. If we had a suite called "server-only", then you would put tests for the server only suite under

CONCEPT

Where do Tests Live?



workstation/test/integration/default/**serverspec**/default_spec.rb

This tells Test Kitchen (and Busser) which Busser runner plugin needs to be installed on the remote instance.

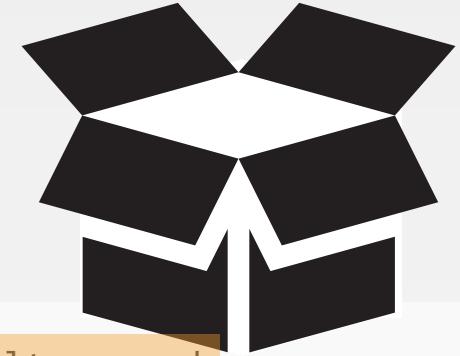
v3.0.0

<http://kitchen.ci/docs/getting-started/writing-test>

 CHEF™
CODE CAN

CONCEPT

Where do Tests Live?



workstation/test/integration/default/serverspec/**default_spec.rb**

All test files (or specs) are named after the recipe they test and end with the suffix "`_spec.rb`". A spec missing that will not be found when executing `kitchen verify`.

v3.0.0

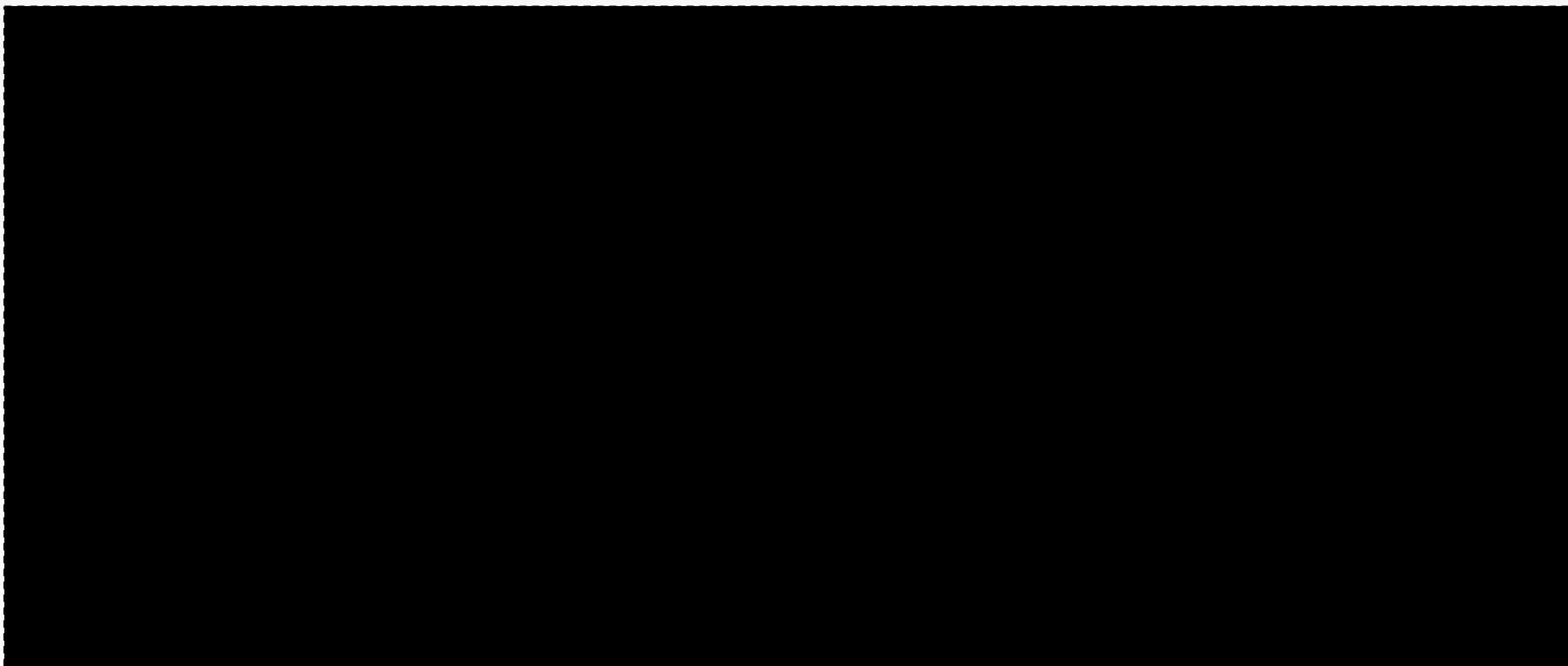
<http://kitchen.ci/docs/getting-started/writing-test>

 CHEF™
CODE CAN

Return home



```
$ cd ~
```

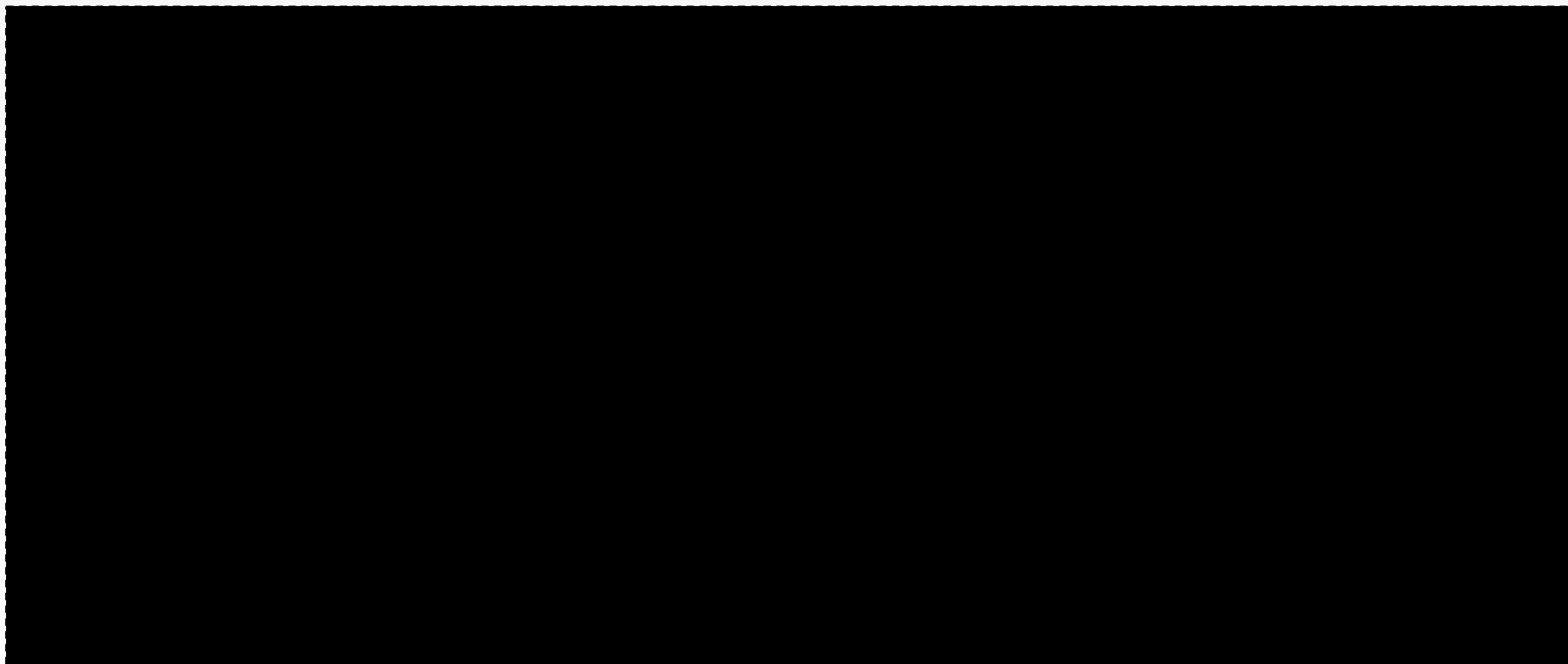


v3.0.0

Move into the Cookbook



```
$ cd cookbooks/workstation
```



v3.0.0

Running the Specification

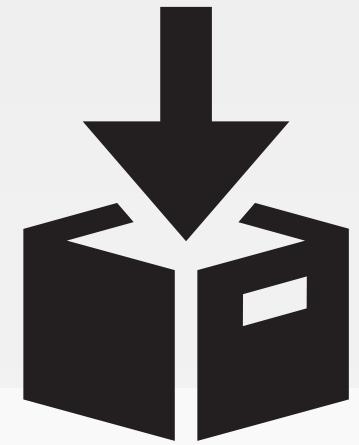


```
$ kitchen verify
```

```
----> Starting Kitchen (v1.4.0)
----> Setting up <default-ubuntu-1404>...
$$$$$ Running legacy setup for 'Docker' Driver
----> Installing Busser (busser)
Fetching: thor-0.19.0.gem (100%)
  Successfully installed thor-0.19.0
Fetching: busser-0.7.1.gem (100%)
  Successfully installed busser-0.7.1
  2 gems installed
----> Setting up Busser
  Creating BUSSER_ROOT in /tmp/verifier
  Creating busser binstub
  Installing Busser plugins: busser-serverspec
```

COMMIT

Commit Your Work



```
$ cd ~/cookbooks/workstation  
$ git add .  
$ git status  
$ git commit -m "Added first test for the  
default recipe"
```

DISCUSSION

More Tests



What are other resources within the recipe that we could test?

CONCEPT

Testing a File



ServerSpec can help us assert different characteristics about files on the file system. Like if it is a file, directory, socket or symlink. The mode, owner, or group. If it is readable, writeable, or executable. Even the data it contains.

The file contains data

```
describe file("/etc/passwd") do
  it { should be_file }
end
```

I expect the file named "/etc/passwd" to be a file (as opposed to a directory, socket or symlink).

The file contains specific content

```
describe file("/etc/httpd/conf/httpd.conf") do
  its(:content) { should match /ServerName www.example.jp/ }
end
```

I expect the file named "/etc/httpd/conf/httpd.conf" to have content that matches "ServerName www.example.jp"

The file is owned by a particular user

```
describe file("/etc/sudoers") do
  it { should be_owned_by "root" }
end
```

I expect the file named "/etc/sudoers" to be owned by the "root" user.

EXERCISE

More Tests



- Add tests that validate that the remaining package resources have been installed
- Add tests that validate the file resource
- Run `kitchen verify` to validate the test meets the expectations that you defined

http://serverspec.org/resource_types.html#package
http://serverspec.org/resource_types.html#file

Our assertion in a spec file

```
~/cookbooks/workstation/test/integration/default/serverspec/default_spec.rb
```

```
require "spec_helper"

describe "workstation::default" do

  describe package("tree") do
    it { should be_installed }
  end

  describe package("git") do
    it { should be_installed }
  end

end
```

The package named "git" is installed.

Our assertion in a spec file

```
~/cookbooks/workstation/test/integration/default/serverspec/default_spec.rb
```

```
...
describe package("git") do
  it { should be_installed }
end

describe file("/etc/motd") do
  it { should be_owned_by "root" }
end

end
```

The file named "/etc/motd" should be owned by "root".

COMMIT

Commit Your Work



```
$ cd ~/cookbooks/workstation  
$ git add .  
$ git status  
$ git commit -m "Added additional tests for  
default recipe"
```

DISCUSSION

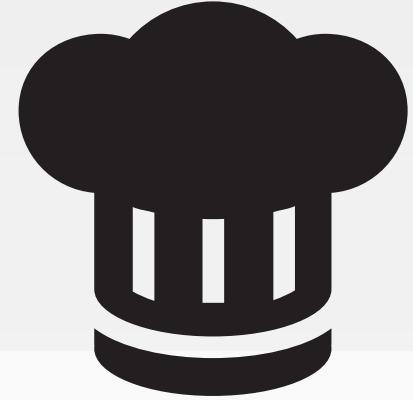
Testing



What questions can we help you answer?

LAB

Testing Our Webserver



I would love to know that the webserver is installed and running correctly.

OBJECTIVE:

- Discuss and decide what should be tested with the apache cookbook

DISCUSSION

Testing



What are some things we could test to validate our web server has deployed correctly?

DISCUSSION

Testing



What manual tests do we use now to validate a working web server?

EXERCISE

Testing Apache



- Create a test file for the "apache" cookbook's default recipe
- Add tests that validate a working web server
- Run kitchen verify

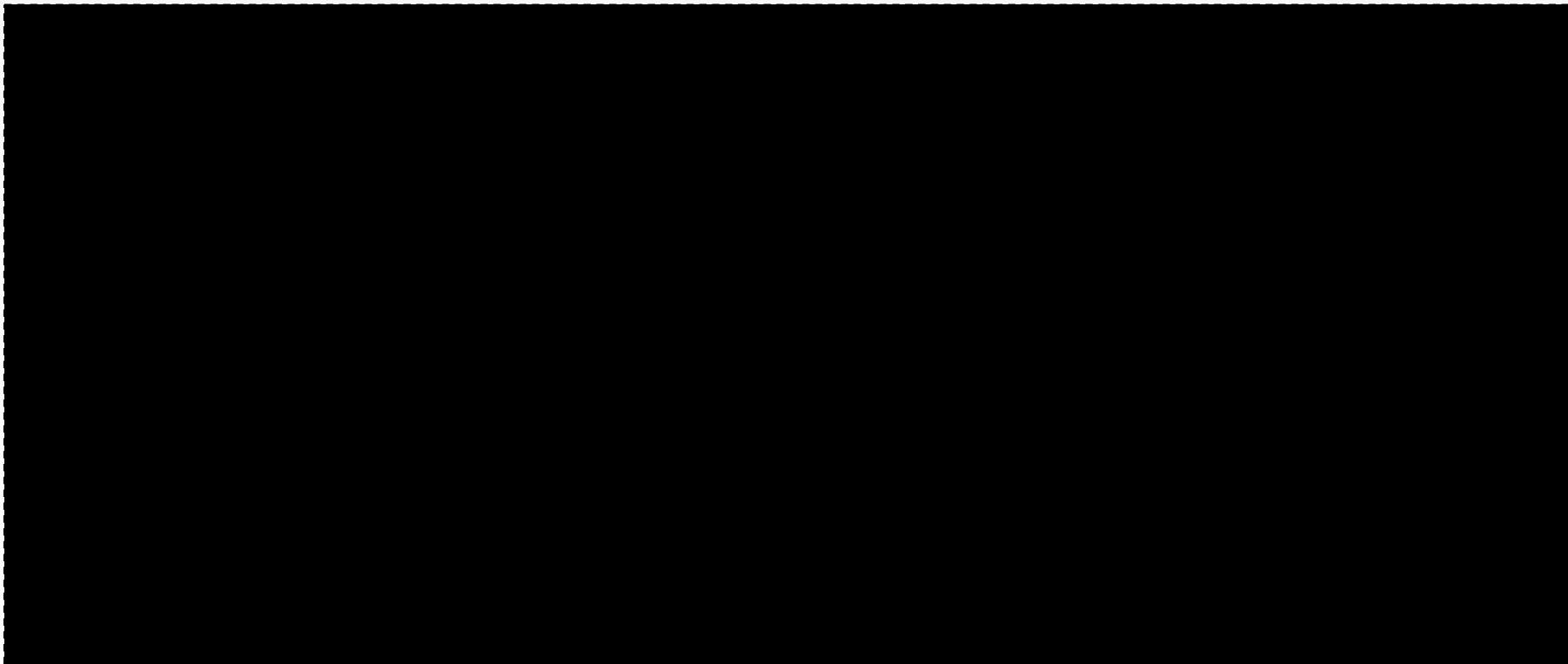
http://serverspec.org/resource_types.html#port

http://serverspec.org/resource_types.html#command

Return home



```
$ cd ~
```



v3.0.0

Move in to the apache cookbook



```
$ cd cookbooks/apache
```

v3.0.0

What does the webserver say?

```
~/cookbooks/apache/test/integration/default/serverspec/default_spec.rb
```

```
require "spec_helper"

describe "apache::default" do
  describe port(80) do
    it { should be_listening }
  end

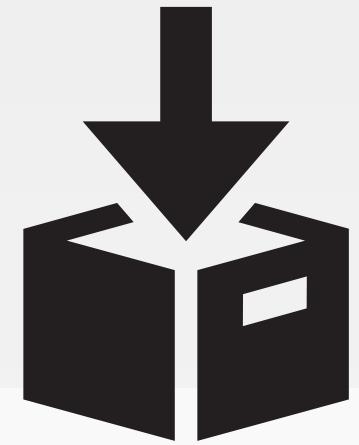
  describe command("curl http://localhost") do
    its(:stdout) { should match /Hello, world!/ }
  end
end
```

The port 80 should be listening.

The standard out from the command "curl http://localhost" should match "Hello, world!"

COMMIT

Commit Your Work



```
$ cd ~/cookbooks/apache  
$ git add .  
$ git status  
$ git commit -m "Added tests for the default  
recipe"
```

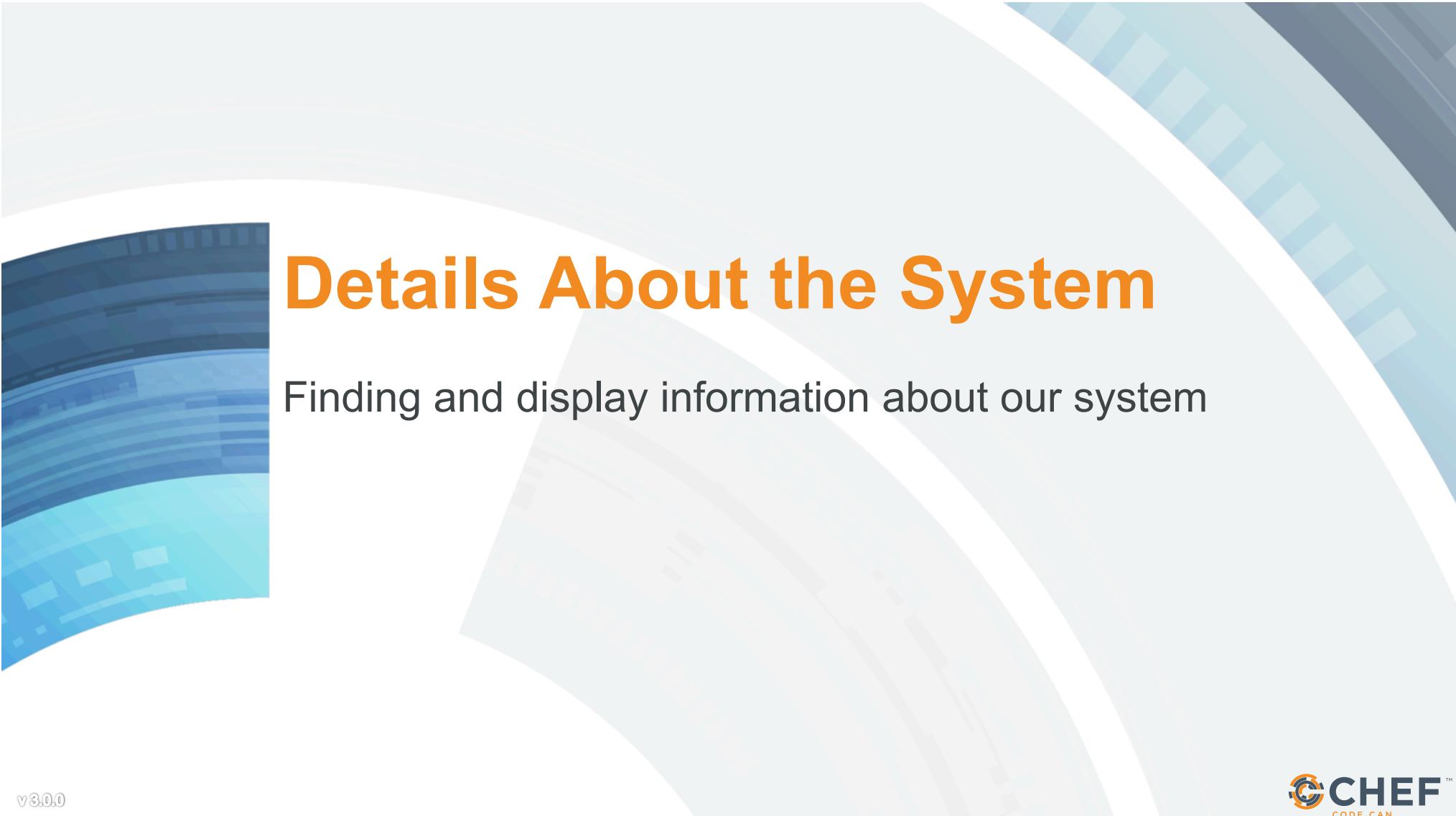
DISCUSSION

Questions



What questions can we help you answer?

- Test Kitchen
- ServerSpec
- Testing



Details About the System

Finding and display information about our system

v3.0.0

New Features!

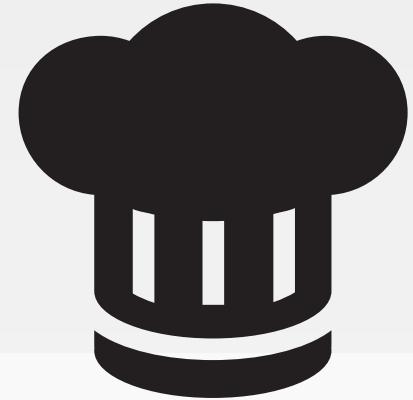


Finally got a chance to read through the cookbooks. Awesome! I shared them with the team over chat and I think it got them excited. George has already submitted a few feature requests:

He thought the Message of the Day and the web page could write out some information about the system ...

LAB

Details about the Node



Displaying system details in the MOTD definitely sounds useful.

OBJECTIVE:

- Update the MOTD file contents, in the "workstation" cookbook, to include node details

Some Useful System Data

- IP Address
- hostname
- memory
- CPU - MHz

Discover the ipaddress



```
$ hostname -I
```

```
104.236.192.102 172.17.42.1
```

v3.0.0

Discover the ipaddress



```
$ ifconfig
```

```
docker0  Link encap:Ethernet  HWaddr 00:00:00:00:00:00
          inet addr:172.17.42.1  Bcast:0.0.0.0  Mask:255.255.0.0
              inet6 addr: fe80::e081:d7ff:fe71:f146/64 Scope:Link
                  UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
                  RX packets:14540 errors:0 dropped:0 overruns:0 frame:0
                  TX packets:17427 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 txqueuelen:0
                  RX bytes:872692 (852.2 KiB)  TX bytes:201605955 (192.2 MiB)
```

```
eth0      Link encap:Ethernet  HWaddr 04:01:3D:E7:09:01
          inet addr:104.236.192.102  Bcast:104.236.255.255  Mask:255.255.192.0
              inet6 addr: fe80::601:3dff:fee7:901/64 Scope:Link
                  UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

Adding the ipaddress

~/cookbooks/workstation/recipes/setup.rb

```
file "/etc/motd" do
  content "Property of ...
  IPADDRESS: 104.236.192.102
  "
  mode "0644"
  owner "root"
  group "root"
end
```

v3.0.0



Discover the hostname



```
$ hostname
```

```
banana-stand
```

v3.0.0

Adding the hostname

~/cookbooks/workstation/recipes/setup.rb

```
file "/etc/motd" do
  content "Property of ...
  IPADDRESS: 104.236.192.102
  HOSTNAME : banana-stand
  "
  mode "0644"
  owner "root"
  group "root"
end
```

v3.0.0



Discover the memory



```
$ cat /proc/meminfo
```

MemTotal:	502272 kB
MemFree:	118384 kB
Buffers:	141156 kB
Cached:	165616 kB
SwapCached:	0 kB
Active:	303892 kB
Inactive:	25412 kB
Active(anon):	22548 kB
Inactive(anon):	136 kB
Active(file):	281344 kB
Inactive(file):	25276 kB
Unevictable:	0 kB
Mlocked:	0 kB

Adding the memory

~/cookbooks/workstation/recipes/setup.rb

```
file "/etc/motd" do
  content "Property of ...
  IPADDRESS: 104.236.192.102
  HOSTNAME : banana-stand
  MEMORY    : 502272 kB
  "
  mode "0644"
  owner "root"
  group "root"
end
```

Discover the cpu - MHz



```
$ cat /proc/cpuinfo
```

```
processor      : 0
vendor_id     : GenuineIntel
cpu family    : 6
model         : 62
model name    : Intel(R) Xeon(R) CPU E5-2630L v2 @ 2.40GHz
stepping       : 4
cpu MHz       : 2399.998
cache size    : 15360 KB
fpu           : yes
fpu_exception : yes
cpuid level   : 13
wp            : yes
flags          : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36
```

Adding the CPU

```
~/cookbooks/workstation/recipes/setup.rb
```

```
file "/etc/motd" do
  content "Property of ...
  IPADDRESS: 104.236.192.102
  HOSTNAME : banana-stand
  MEMORY   : 502272 kB
  CPU       : 2399.998 MHz
  "
  mode "0644"
  owner "root"
  group "root"
end
```



Introducing a Change

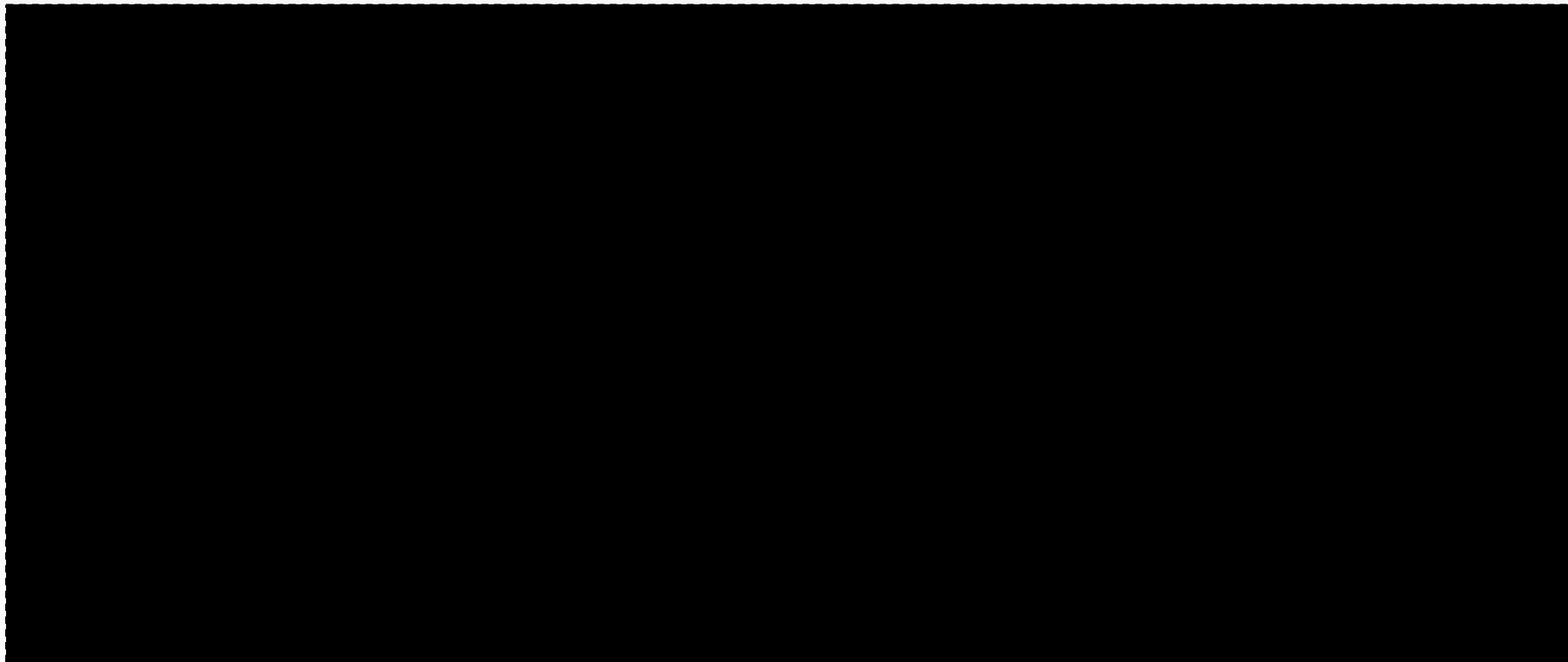
By creating a change we have introduced risk.
There is a chance that the code we wrote will not
work when we attempt to deploy it.

Lets run our cookbook tests before we apply the
updated recipe.

Change into our cookbook



```
$ cd ~/cookbooks/workstation
```



v3.0.0

Run our Tests



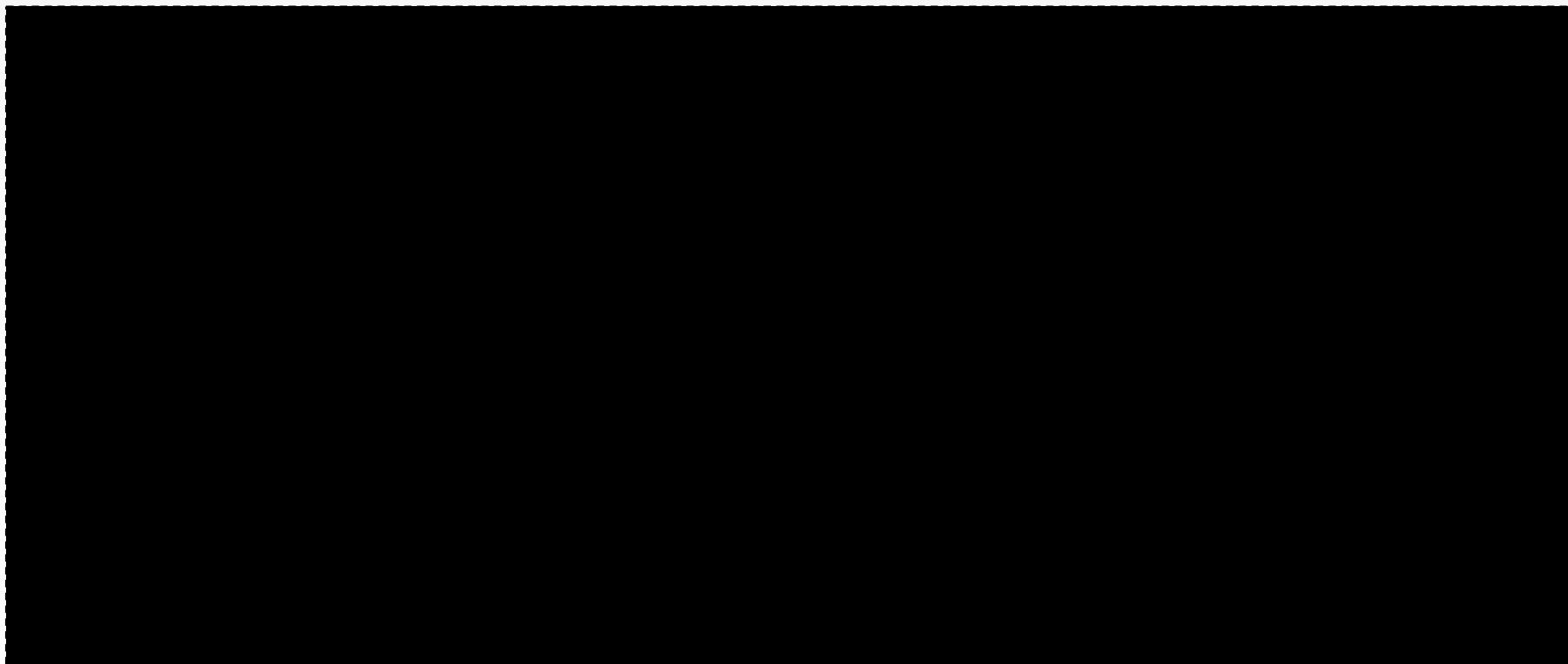
```
$ kitchen test
```

```
----> Starting Kitchen (v1.4.0)
----> Setting up <default-ubuntu-1404>...
$$$$$ Running legacy setup for 'Docker' Driver
----> Installing Busser (busser)
Fetching: thor-0.19.0.gem (100%)
  Successfully installed thor-0.19.0
Fetching: busser-0.7.1.gem (100%)
  Successfully installed busser-0.7.1
  2 gems installed
----> Setting up Busser
  Creating BUSSER_ROOT in /tmp/verifier
  Creating busser binstub
  Installing Busser plugins: busser-serverspec
```

Return Home



```
$ cd ~
```



v3.0.0

Apply the workstation cookbook



```
$ sudo chef-client --local-mode -r "recipe[workstation]"
```

```
resolving cookbooks for run list: ["workstation"]
```

```
Synchronizing Cookbooks:
```

```
  - workstation
```

```
Compiling Cookbooks...
```

```
Converging 5 resources
```

```
Recipe: setup::default
```

```
  * apt_package[nano] action install (up to date)
  * apt_package[vim] action install (up to date)
  * apt_package[emacs] action install (up to date)
  * apt_package[tree] action install (up to date)
  * file[/etc/motd] action create (up to date)
```

```
Running handlers:
```

v3.0.0

Verify that the /etc/motd has been updated



```
$ cat /etc/motd
```

```
Property of ...
```

```
IPADDRESS: 104.236.192.102  
HOSTNAME : banana-stand  
MEMORY    : 502272 kB  
CPU       : 2399.998 MHz
```

DISCUSSION

Capturing System Data



What are the limitations of the way we captured this data?

DISCUSSION

Capturing System Data



How accurate will our MOTD be when we deploy it
on other systems?

DISCUSSION

Capturing System Data



Are these values we would want to capture in our tests?

Hard Coded Values



The values that we have derived at this moment may not be the correct values when we deploy this recipe again even on the same system!

DISCUSSION

Data In Real Time



How could we capture this data in real-time?

CONCEPT

Ohai!

Ohai is a tool that already captures all the data that we similarly demonstrated finding.



Ohai!



```
$ ohai
```

```
{  
  "kernel": {  
    "name": "Linux",  
    "release": "2.6.32-431.1.2.0.1.el6.x86_64",  
    "version": "#1 SMP Fri Dec 13 13:06:13 UTC 2013",  
    "machine": "x86_64",  
    "os": "GNU/Linux",  
    "modules": {  
      "veth": {  
        "size": "5040",  
        "refcount": "0"  
      },  
      "ipt_addrtype": {  
        "size": "5040",  
        "refcount": "0"  
      }  
    }  
  }  
}
```

v3.0.0

 CHEF™
CODE CAN

CONCEPT

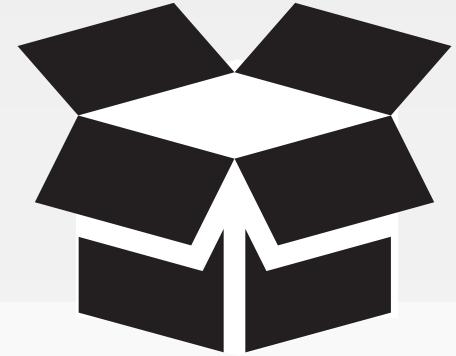
All About The System



Ohai queries the operating system with a number of commands, similar to the ones demonstrated. The data is presented in JSON (JavaScript Object Notation).

CONCEPT

ohai + chef-client = <3



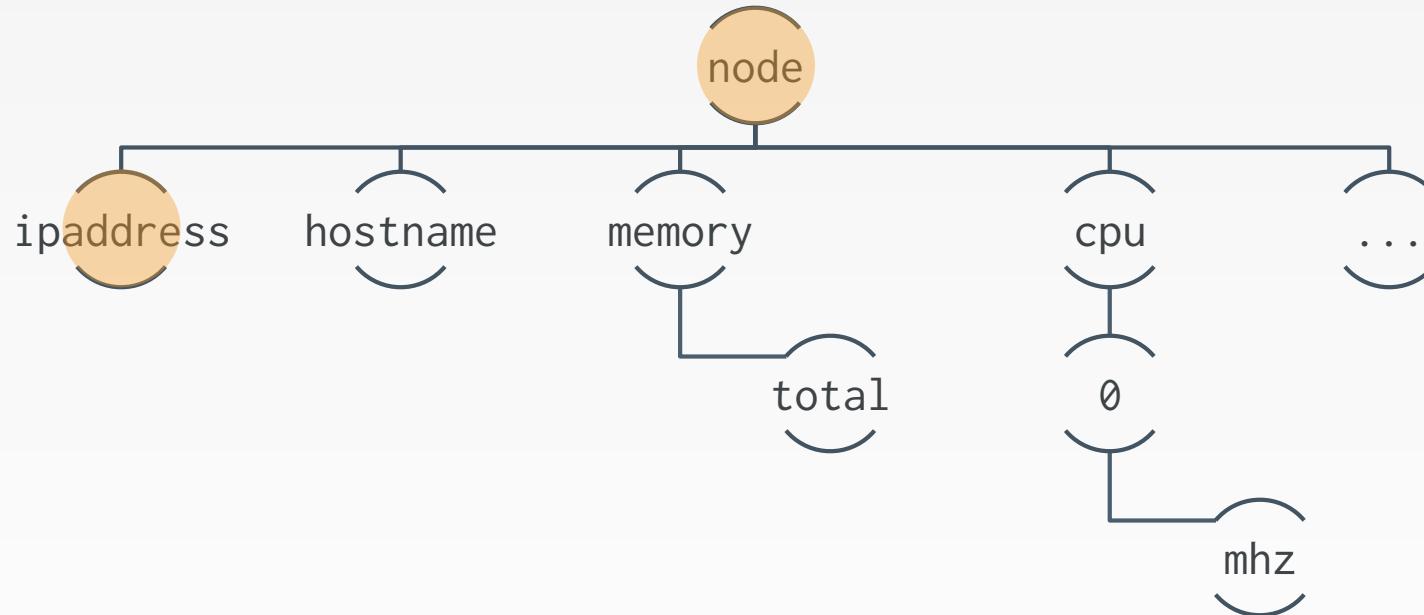
chef-client and chef-apply automatically executes ohai and stores the data about the node in an object we can use within the recipes named node.

CONCEPT

The node



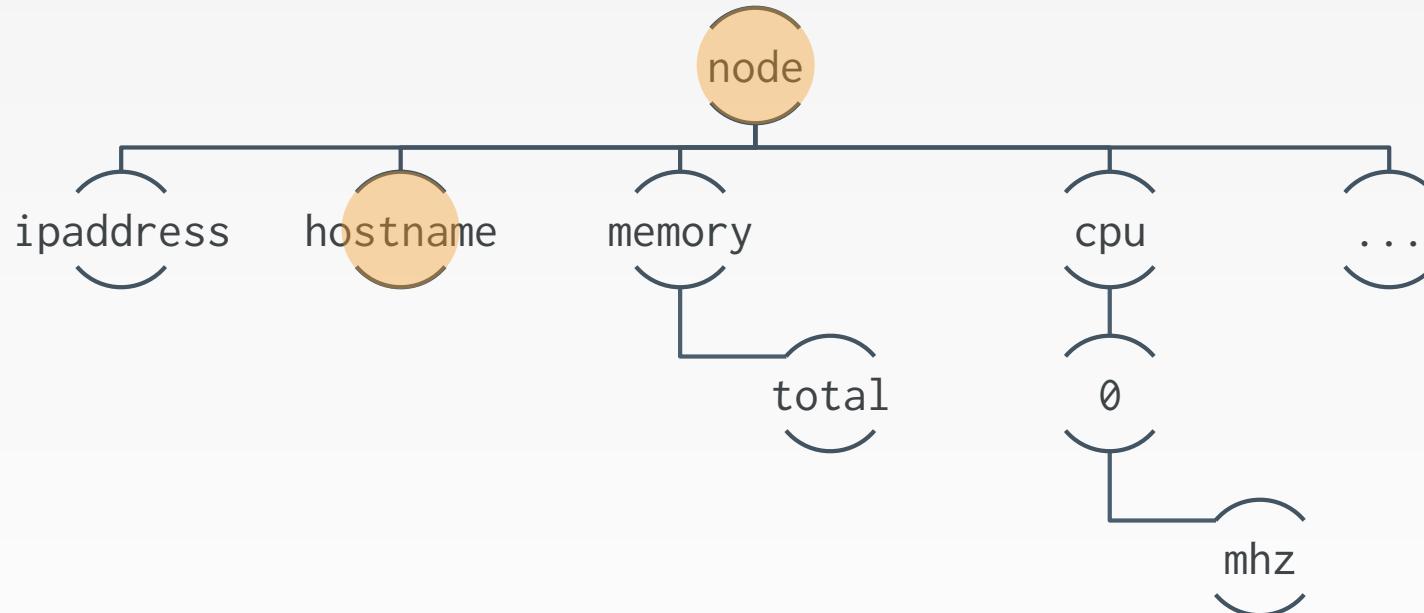
An attribute is a specific detail about a node, such as an IP address, a host name, a list of loaded kernel modules, the version(s) of available programming languages that are available, and so on.



IPADDRESS: 104.236.192.102

"IPADDRESS: #{node["ipaddress"]}"

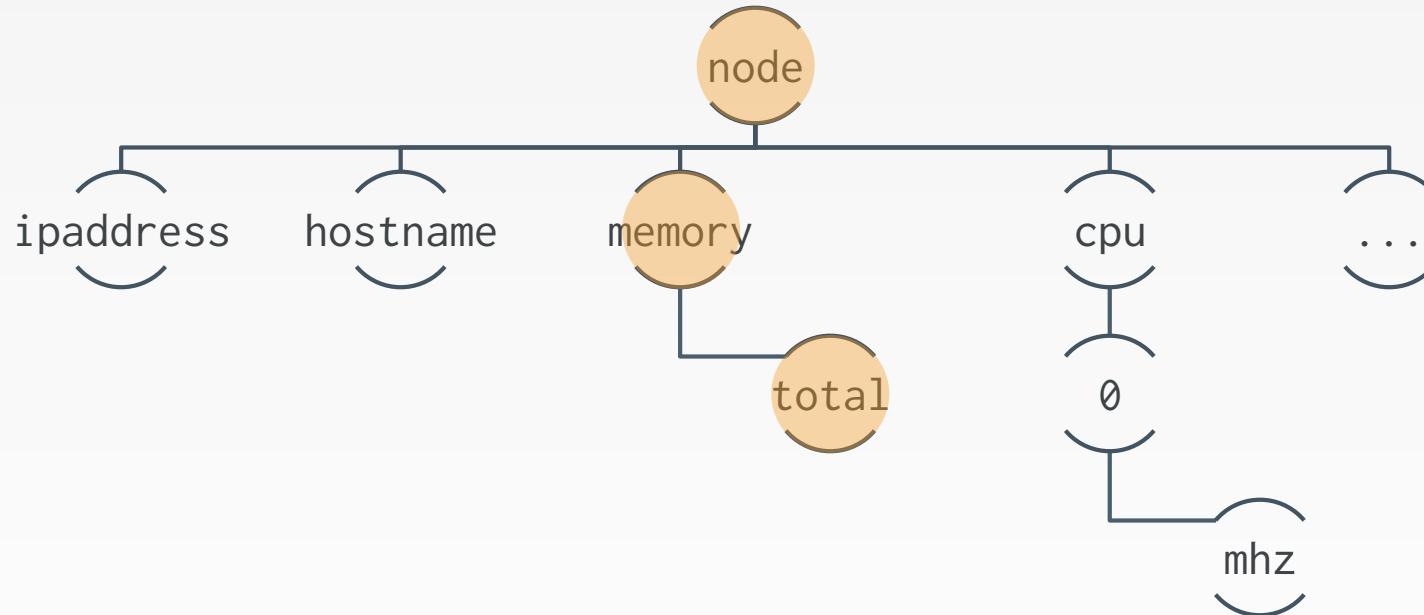
v3.0.0



HOSTNAME: banana-stand

"HOSTNAME: #{node["hostname"]}"

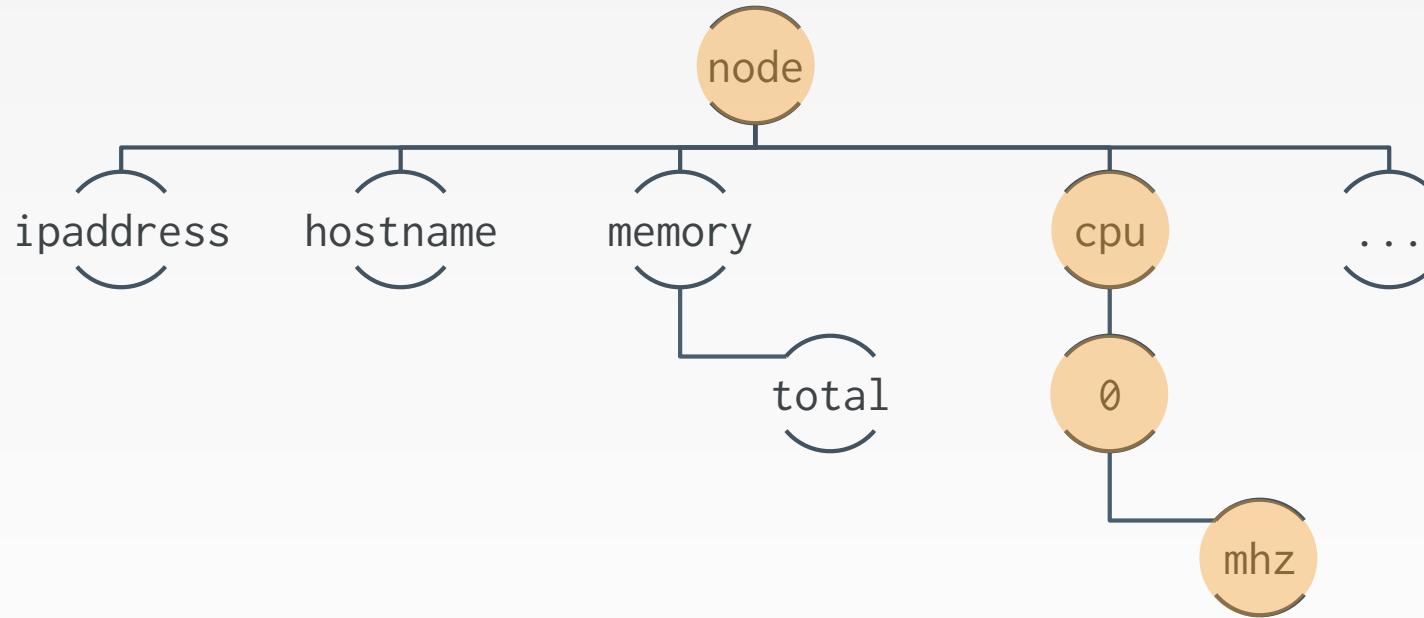
v3.0.0



MEMORY: 502272kB

"Memory: #{node["memory"]["total"]}"

v3.0.0



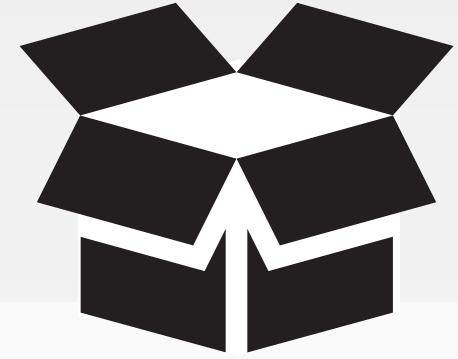
CPU: 2399.998MHz

```
"CPU: #{node["cpu"]["0"]["mhz"]}"
```

v3.0.0

CONCEPT

String Interpolation

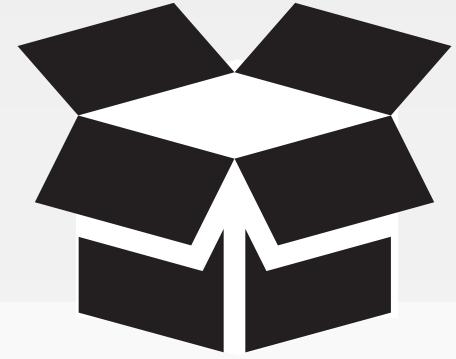


I have 4 apples

```
apple_count = 4  
puts "I have #{apple_count} apples"
```

CONCEPT

String Interpolation



I have 4 apples

```
apple_count = 4  
puts "I have #{apple_count} apples"
```

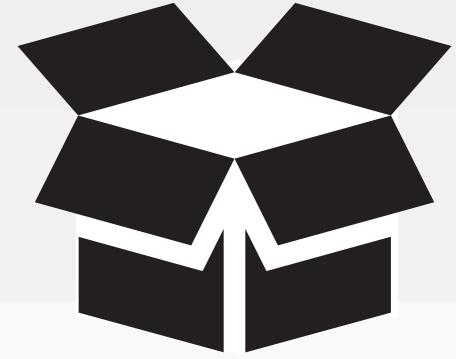
http://en.wikipedia.org/wiki/String_interpolation#Ruby

v3.0.0

 **CHEF**TM
CODE CAN

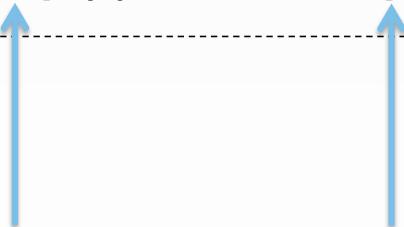
CONCEPT

String Interpolation



I have 4 apples

```
apple_count = 4  
puts "I have #{apple_count} apples"
```



Using the node's IP Address

~/cookbooks/workstation/recipes/setup.rb

```
file "/etc/motd" do
  content "Property of ...
  IPADDRESS: #{node["ipaddress"]}
  HOSTNAME : banana-stand
  MEMORY    : 502272 kB
  CPU       : 2399.998 MHz
  "
  mode "0644"
  owner "root"
  group "root"
end
```

v3.0.0



Using the node's hostname

```
[ ] ~/cookbooks/workstation/recipes/setup.rb
```

```
file "/etc/motd" do
  content "Property of ...
  IPADDRESS: #{node["ipaddress"]}
  HOSTNAME : #{node["hostname"]}
  MEMORY    : 502272 kB
  CPU        : 2399.998 MHz
  "
  mode "0644"
  owner "root"
  group "root"
end
```

Using the node's total memory

```
[/] ~/cookbooks/workstation/recipes/setup.rb
```

```
file "/etc/motd" do
  content "Property of ...
  IPADDRESS: #{node["ipaddress"]}
  HOSTNAME : #{node["hostname"]}
  MEMORY    : #{node["memory"]["total"]}
  CPU       : 2399.998 MHz
  "
  mode "0644"
  owner "root"
  group "root"
end
```

Using the node's CPU MHz

```
[ ] ~/cookbooks/workstation/recipes/setup.rb
```

```
file "/etc/motd" do
  content "Property of ...
  IPADDRESS: #{node["ipaddress"]}
  HOSTNAME : #{node["hostname"]}
  MEMORY    : #{node["memory"]["total"]}
  CPU        : #{node["cpu"]["0"]["mhz"]}
  "
  mode "0644"
  owner "root"
  group "root"
end
```

EXERCISE

Verify the Changes



- Change directory into the "workstation" cookbook's directory
- Run kitchen test for the "workstation" cookbook
- Change directory into the home directory
- Run chef-client locally to verify the "workstation" cookbook's default recipe.

LAB

Changes Mean a New Version



Lets bump the version number and check in the code to source control.

OBJECTIVE:

- Update the version of the "workstation" cookbook
- Commit the changes to the "workstation" cookbook to version control

CONCEPT

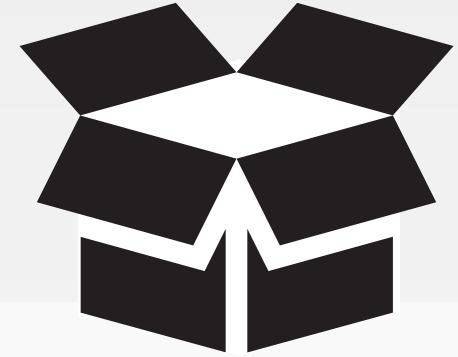
Cookbook Versions



A cookbook version represents a set of functionality that is different from the cookbook on which it is based. A version may exist for many reasons, such as ensuring the correct use of a third-party component, updating a bug fix, or adding an improvement.

CONCEPT

Semantic Versions



Given a version number **MAJOR.MINOR.PATCH**,
increment the:

- **MAJOR** version when you make incompatible API changes
- **MINOR** version when you add functionality in a backwards-compatible manner
- **PATCH** version when you make backwards-compatible bug fixes

DISCUSSION

Major, Minor, or Patch?



What kind of changes did you make to the cookbook?

Update the Cookbook Version

```
~/cookbooks/workstation/metadata.rb
```

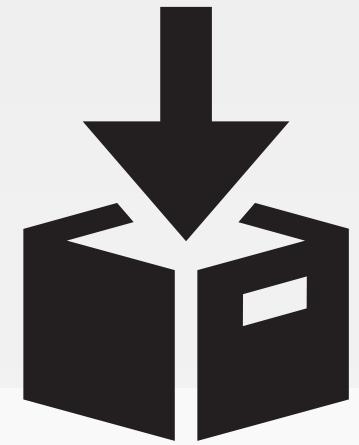
```
name          'workstation'  
maintainer    'The Authors'  
maintainer_email 'you@example.com'  
license        'all_rights'  
description    'Installs/Configures workstation'  
long_description 'Installs/Configures workstation'  
version        '0.2.0'
```

v3.0.0



COMMIT

Commit Your Work



```
$ cd ~/cookbooks/workstation  
$ git add .  
$ git status  
$ git commit -m "Version 0.2.0 - Added Node  
Details in MOTD"
```

EXERCISE

Node Details in the Webserver



- ❑ The file resource named "/var/www/html/index.html" is created with the content that includes the node details:
 - ipaddress
 - hostname
- ❑ Run kitchen test for the "apache" cookbook
- ❑ Run chef-client to locally apply the "apache" cookbook's default recipe.
- ❑ Update the version of the "apache" cookbook
- ❑ Commit the changes to the "apache" cookbook to version control

Apache Recipe



~/cookbooks/apache/recipes/server.rb

...

```
file "/var/www/html/index.html" do
  content "<h1>Hello, world!</h1>
<h2>ipaddress: #{node["ipaddress"]}</h2>
<h2>hostname: #{node["hostname"]}</h2>
"
end
```

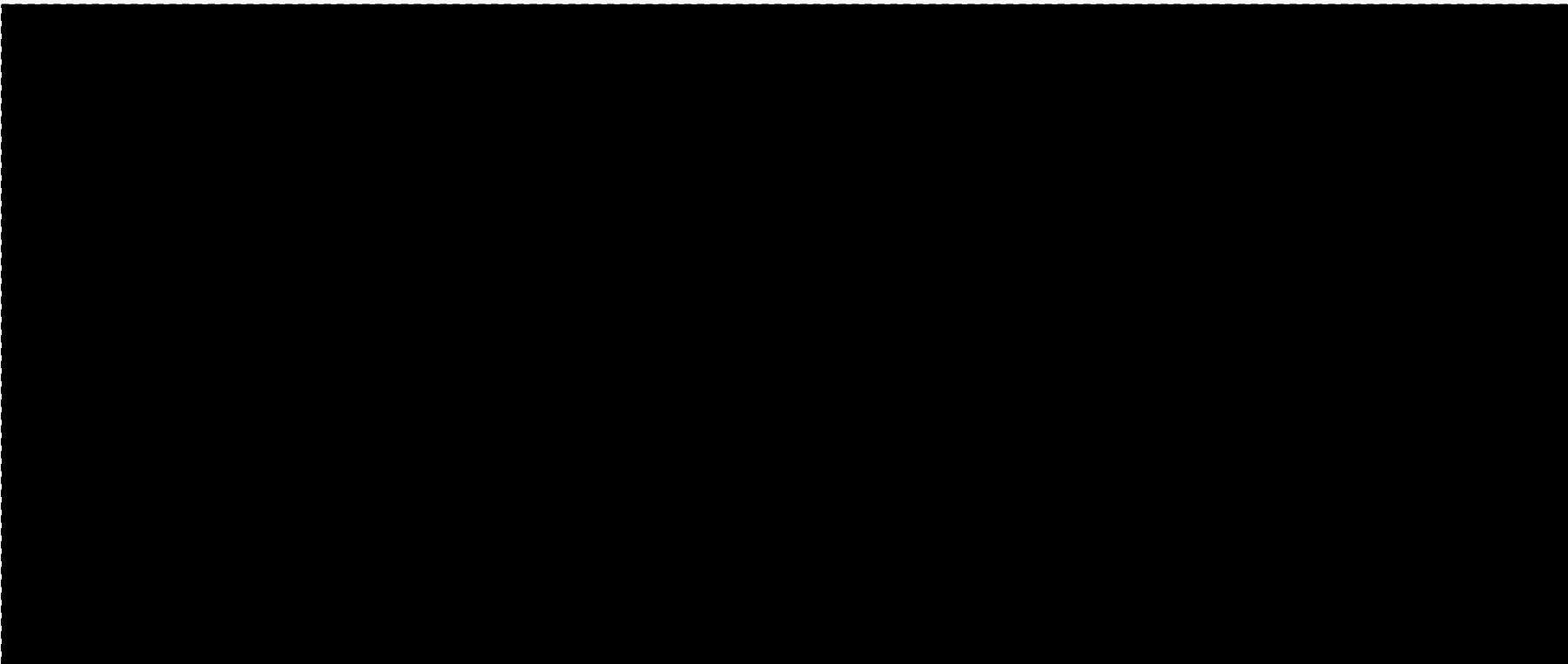
...

v3.0.0

Change into the Apache Cookbook



```
$ cd cookbooks/apache
```



v3.0.0

Test the apache cookbook's default recipe



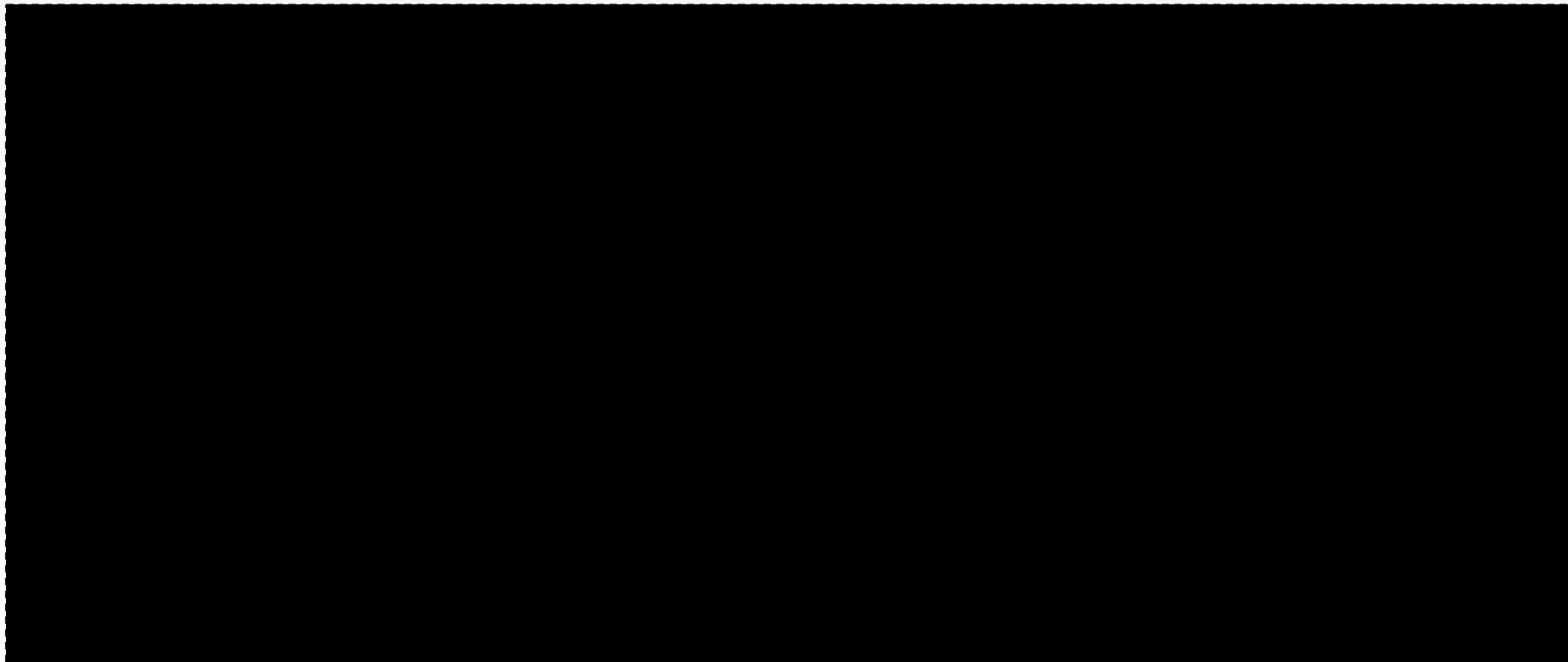
```
$ kitchen test
```

```
----> Starting Kitchen (v1.4.0)
----> Setting up <default-ubuntu-1404>...
$$$$$ Running legacy setup for 'Docker' Driver
----> Installing Busser (busser)
Fetching: thor-0.19.0.gem (100%)
  Successfully installed thor-0.19.0
Fetching: busser-0.7.1.gem (100%)
  Successfully installed busser-0.7.1
  2 gems installed
----> Setting up Busser
  Creating BUSSER_ROOT in /tmp/verifier
  Creating busser binstub
  Installing Busser plugins: busser-serverspec
```

Return to the home directory



```
$ cd ~
```



Run chef-client to apply the apache cookbook



```
$ sudo chef-client --local-mode -r "recipe[apache]"
```

```
[2015-05-05T08:09:08+00:00] WARN: No config file found or specified on command line, using  
command line options.  
Starting Chef Client, version 12.3.0  
resolving cookbooks for run list: ["apache"]  
Synchronizing Cookbooks:  
  - apache  
Compiling Cookbooks...  
Converging 3 resources  
Recipe: apache::server  
  * apt_package[apache2] action install  
    - install version 2.4.7-1ubuntu4.4 of package apache2  
  * service[apache2] action enable (up to date)  
  * service[apache2] action start (up to date)
```

Update the Cookbook Version

```
~/cookbooks/apache/metadata.rb
```

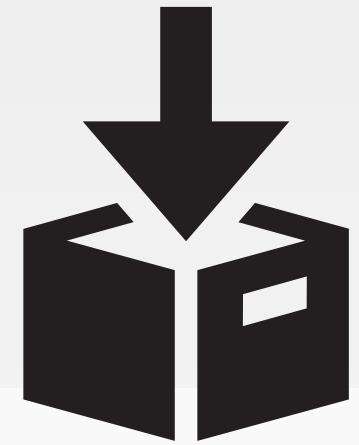
```
name          'apache'
maintainer    'The Authors'
maintainer_email 'you@example.com'
license        'all_rights'
description    'Installs/Configures apache'
long_description 'Installs/Configures apache'
version        '0.2.0'
```

v3.0.0



COMMIT

Commit Your Work



```
$ cd ~/cookbooks/apache  
$ git add .  
$ git status  
$ git commit -m "Version 0.2.0 - Added Node  
Details in Index Page"
```

DISCUSSION

Questions



What questions can we help you answer?

- Ohai
- Node Object
- Node Attributes
- String Interpolation
- Semantic Versions



Desired State and Data

Extracting the content for clarity

Cleaner recipes?

This message has no body.



Apache Recipe



```
~/cookbooks/apache/recipes/server.rb
```

```
package "apache2"

file "/var/www/html/index.html" do
  content "<h1>Hello, world!</h1>
<h2>ipaddress: #{node["ipaddress"]}</h2>
<h2>hostname: #{node["hostname"]}</h2>
"
end

service "apache2" do
  action [ :enable, :start ]
end
```

v3.0.0

Double Quotes close Double Quotes



Double quoted strings are terminated by double quotes.

```
"<h1 style="color: red;">Hello, World!</h1>"
```



CONCEPT

Backslash



We can use double-quotes as long as we prefix them with a backslash.

```
"<h1 style=\"color: red;\">Hello, World!</h1>"
```





Backslash

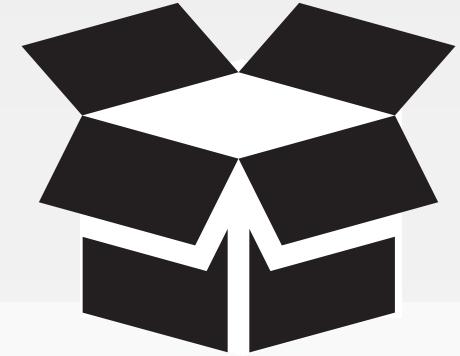
Backslashes are reserved characters. So to use them you need to use a backslash.

"My feelings :\\"



CONCEPT

Backslash



Backslashes are reserved characters. So to use them you need to use a backslash.

```
"My feelings :\\\"
```



Unexpected Formatting

```
file "/etc/motd" do
  content "This is the first line of the file.
    This is the second line. If I try and line it up..."
```

Don't even think about pasting ASCII ART in here!

```
""
end
```

This is the first line of the file.

This is the second line. If I try and line it up...

Don't even think about pasting ASCII ART in here!



Copy Pasta

This process is definitely error prone. Especially because a human has to edit the file again before it is deployed.

CONCEPT

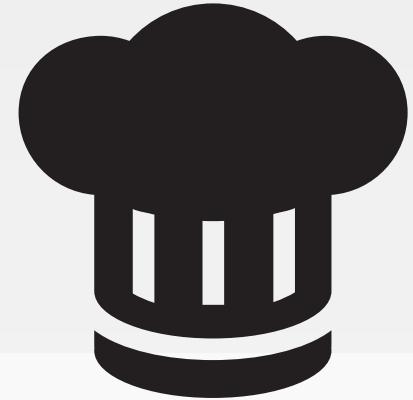
What We Need



the ability to store the data in another file which is in the native format of the file we are writing out but that still allows us to insert ruby code -- specifically the node attributes we have defined.

LAB

Cleaner Recipes



Adding the node attributes to our recipes did make it harder to read.

OBJECTIVE:

- Decide which resource will help us address this issue

DOCS

Let's Check the Docs...



Use the file resource to manage files directly on a node.

Use the **cookbook_file** resource to copy a file from a cookbook's /files directory. Use the **template** resource to create a file based on a template in a cookbook's / templates directory. And use the **remote_file** resource to transfer a file to a node from a remote location.

DOCS

remote_file



Use the **remote_file** resource to transfer a file from a remote location using file specificity. This resource is similar to the `file` resource.

DOCS

cookbook_file



Use the **cookbook_file** resource to transfer files from a sub-directory of COOKBOOK_NAME/files/ to a specified path located on a host that is running the chef-client.

cookbook_file's source match up

```
$ tree cookbooks/apache/files/default  
files/default
```

```
└── index.html
```

```
0 directories, 1 file
```

```
cookbook_file "/var/www/index.html" do
```

```
  source "index.html"
```

```
end
```



DOCS

template



A cookbook template is an Embedded Ruby (ERB) template that is used to generate files ... Templates may contain Ruby expressions and statements and are a great way to... Use the template resource to add cookbook templates to recipes; place the corresponding Embedded Ruby (ERB) template in a cookbook's /templates directory.

template file's source matches up

```
$ tree cookbooks/apache/templates/default  
templates/default
```

```
└── index.html.erb
```

```
0 directories, 1 file
```

```
template "/var/www/index.html" do
```

```
  source "index.html.erb"
```

```
end
```

DOCS

template



To use a template, two things must happen:

1. A template resource must be added to a recipe
2. An Embedded Ruby (ERB) template must be added to a cookbook

DISCUSSION

Replacement Resource



What resource could be used in this situation?

DISCUSSION

Which Resource?



What resource will allow us to insert our node data into the file that it copies to the target system?

DISCUSSION

The Template Resource



Why is using the template resource the best choice
in this situation?

LAB

Cleaner Apache Recipe



Adding the node attributes to the index page did make it harder to read the recipe.

OBJECTIVE:

- Create a template with chef generate
- Define the contents of the ERB template
- Change the file resource to the template resource in the 'apache' cookbook

CONCEPT

What is chef?



An executable program that allows you generate cookbooks and cookbook components.

v3.0.0

 CHEF™
CODE CAN

What can chef do?



```
$ chef --help
```

Usage:

```
chef -h/--help  
chef -v/--version  
chef command [arguments...] [options...]
```

Available Commands:

exec	Runs the command in context of the embedded ruby
gem	Runs the 'gem' command in context of the embedded ruby
generate	Generate a new app, cookbook, or component
shell-init	Initialize your shell to use ChefDK as your primary ruby
install	Install cookbooks from a Policyfile and generate a locked cookbook set
update	Updates a Policyfile.lock.json with latest run_list and cookbooks

v3.0.0

 CHEF™
CODE CAN

What can chef generate do?



```
$ chef generate --help
```

```
Usage: chef generate GENERATOR [options]
```

```
Available generators:
```

- | | |
|------------|---|
| app | Generate an application repo |
| cookbook | Generate a single cookbook |
| recipe | Generate a new recipe |
| attribute | Generate an attributes file |
| template | Generate a file template |
| file | Generate a cookbook file |
| lwrp | Generate a lightweight resource/provider |
| repo | Generate a Chef policy repository |
| policyfile | Generate a Policyfile for use with the install/push commands (experimental) |

What can chef generate template do?



```
$ chef generate template --help
```

```
Usage: chef generate template [path/to/cookbook] NAME [options]
```

- | | |
|---|--|
| -C, --copyright COPYRIGHT | Name of the copyright holder - defaults to 'The Authors' |
| -m, --email EMAIL | Email address of the author - defaults to ... |
| -a, --generator-arg KEY=VALUE | Use to set arbitrary attribute KEY to VALUE in the
all_rights, apache2, mit, gplv2, gplv3 - defaults to |
| -I, --license LICENSE | all_rights, apache2, mit, gplv2, gplv3 - defaults to |
| -s, --source SOURCE_FILE | Copy content from SOURCE_FILE |
| -g GENERATOR_COOKBOOK_PATH,
--generator-cookbook | Use GENERATOR_COOKBOOK_PATH for the code_generator |

Use chef to generate a template



```
$ chef generate template cookbooks/apache index.html
```

```
Compiling Cookbooks...
Recipe: code_generator::template
  * directory[cookbooks/apache/templates/default] action create
    - create new directory cookbooks/apache/templates/default
  * template[cookbooks/apache/templates/default/index.html.erb] action create
    - create new file cookbooks/apache/templates/default/index.html.erb
    - update content in file cookbooks/apache/templates/default/index.html.erb from none to
e3b0c4
  (diff output suppressed by config)
```

Lets look at the template file



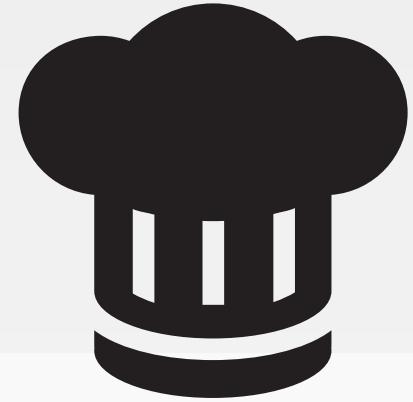
```
$ tree cookbooks/apache/templates
```

```
cookbooks/apache/templates/
└── default
    └── index.html.erb
```

```
1 directory, 1 file
```

LAB

Cleaner Recipes



Adding the node attributes to the default page did make it harder to read the recipe.

OBJECTIVE:

- ✓ Create a template with chef generate
- ❑ Define the contents of the ERB template
- ❑ Change the file resource to the template resource in the 'apache' cookbook

CONCEPT

ERB



An Embedded Ruby (ERB) template allows Ruby code to be embedded inside a text file within specially formatted tags. Ruby code can be embedded using expressions and statements.

Text within an ERB template

```
<% if (50 + 50) == 100 %>
50 + 50 = <%= 50 + 50 %>
<% else %>
At some point all of MATH I learned in school changed.
<% end %>
```

Each ERB tag has a beginning tag and a matched ending tag.

Text within an ERB template

```
<% if (50 + 50) == 100 %>
50 + 50 = <%= 50 + 50 %>
<% else %>
At some point all of MATH I learned in school changed.
<% end %>
```

Each ERB tag has a beginning tag and a matched ending tag.

Text within an ERB template

```
<% if (50 + 50) == 100 %>
```

```
50 + 50 = <%= 50 + 50 %>
```

```
<% else %>
```

At some point all of MATH I learned in school changed.

```
<% end %>
```

Each ERB tag has a beginning tag and a matched ending tag.

Text within an ERB template

```
<% if (50 + 50) == 100 %>
```

```
50 + 50 = <%= 50 + 50 %>
```

```
<% else %>
```

```
At some point all of MATH I learned in school changed.
```

```
<% end %>
```

Executes the ruby code within the brackets and do not display the result.

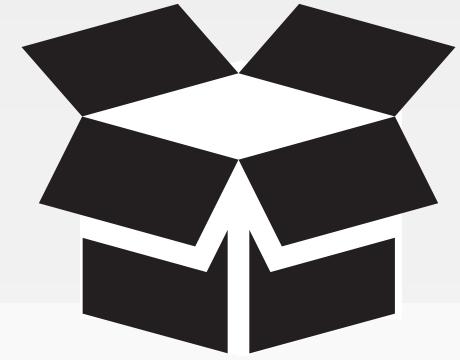
Text within an ERB template

```
<% if (50 + 50) == 100 %>
50 + 50 = <%= 50 + 50 %>
<% else %>
At some point all of MATH I learned in school changed.
<% end %>
```

Executes the ruby code within the brackets and display the results.

CONCEPT

The Angry Squid



<%=

v3.0.0

Move our source to the template

```
~/cookbooks/apache/templates/default/index.html.erb
```

```
<html>
  <body>
    <h1>Hello, world!</h1>
    <h2>ipaddress: #{node["ipaddress"]}</h2>
    <h2>hostname: #{node["hostname"]}</h2>
  </body>
</html>
```

Replace string interpolation with ERB

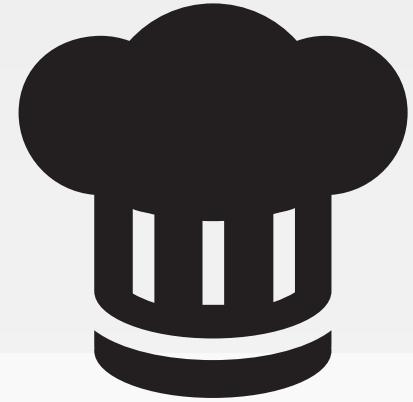


```
~/cookbooks/apache/templates/default/index.html.erb
```

```
<html>
  <body>
    <h1>Hello, world!</h1>
    <h2>ipaddress: <%= node["ipaddress"] %></h2>
    <h2>hostname: <%= node["hostname"] %></h2>
  </body>
</html>
```

LAB

Cleaner Recipes



Adding the node attributes to the default page did make it harder to read the recipe.

OBJECTIVE:

- ✓ Create a template with chef generate
- ✓ Define the contents of the ERB template
- ❑ Change the file resource to the template resource in the 'apache' cookbook

Remove the existing content attribute

```
~/cookbooks/apache/recipes/server.rb
```

```
file "/var/www/html/index.html" do
  content "<h1>Hello, world!</h1>
<h2>IPADDRESS: #{node["ipaddress"]}</h2>
<h2>HOSTNAME : #{node["hostname"]}</h2>
"
end
```

Change the file resource to a template

```
~/cookbooks/apache/recipes/server.rb
```

```
template "/var/www/html/index.html" do
```

```
end
```

v3.0.0

What to specify as the source?



```
~/cookbooks/apache/recipes/server.rb
```

```
template "/var/www/html/index.html" do
  source "????????????????????"
end
```

v3.0.0

Viewing the partial path to the template



```
$ tree cookbooks/apache/templates/default
```

```
cookbooks/apache/templates/default/
```

```
    └── index.html.erb
```

```
0 directories, 1 file
```

Change the file resource to a template



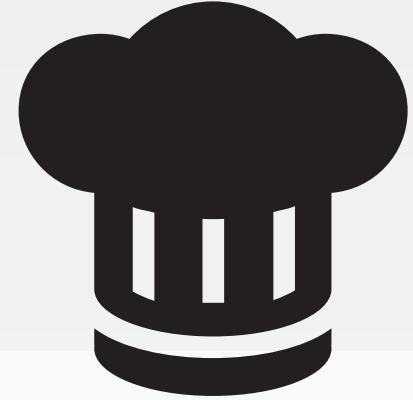
~/cookbooks/apache/recipes/server.rb

```
template "/var/www/html/index.html" do
  source "index.html.erb"
end
```

v3.0.0

LAB

Cleaner Recipes



Adding the node attributes to the default page did make it harder to read the recipe.

OBJECTIVE:

- ✓ Create a template with chef generate
- ✓ Define the contents of the ERB template
- ✓ Change the file resource to the template resource in the 'apache' cookbook

EXERCISE

Update the Version

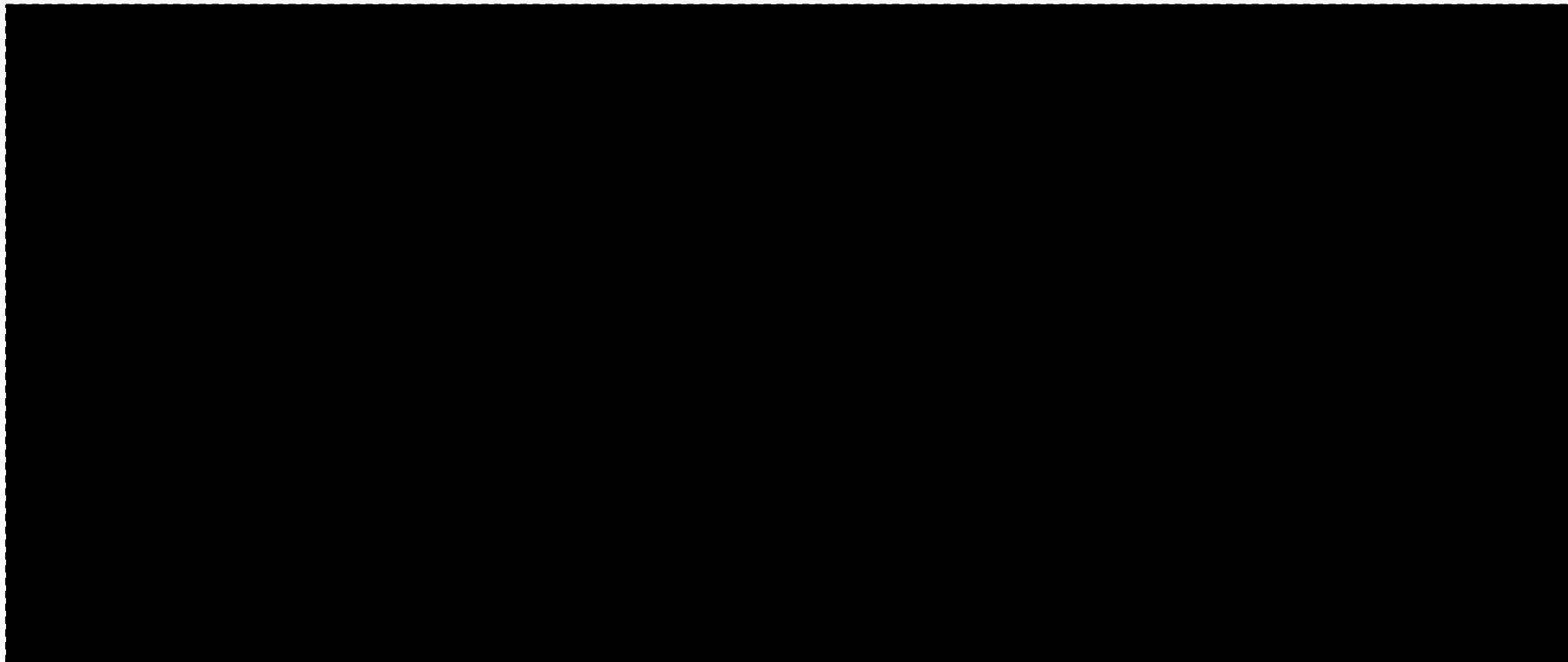


- Use kitchen test on the "apache" cookbook
- Use chef-client to apply the "apache" cookbook's "default" recipe
- Update the "apache" cookbook's version for this patch
- Commit the changes to the "apache" cookbook to version control

Move into the cookbook



```
$ cd ~/cookbooks/apache
```



v3.0.0

Test the cookbook



```
$ kitchen test
```

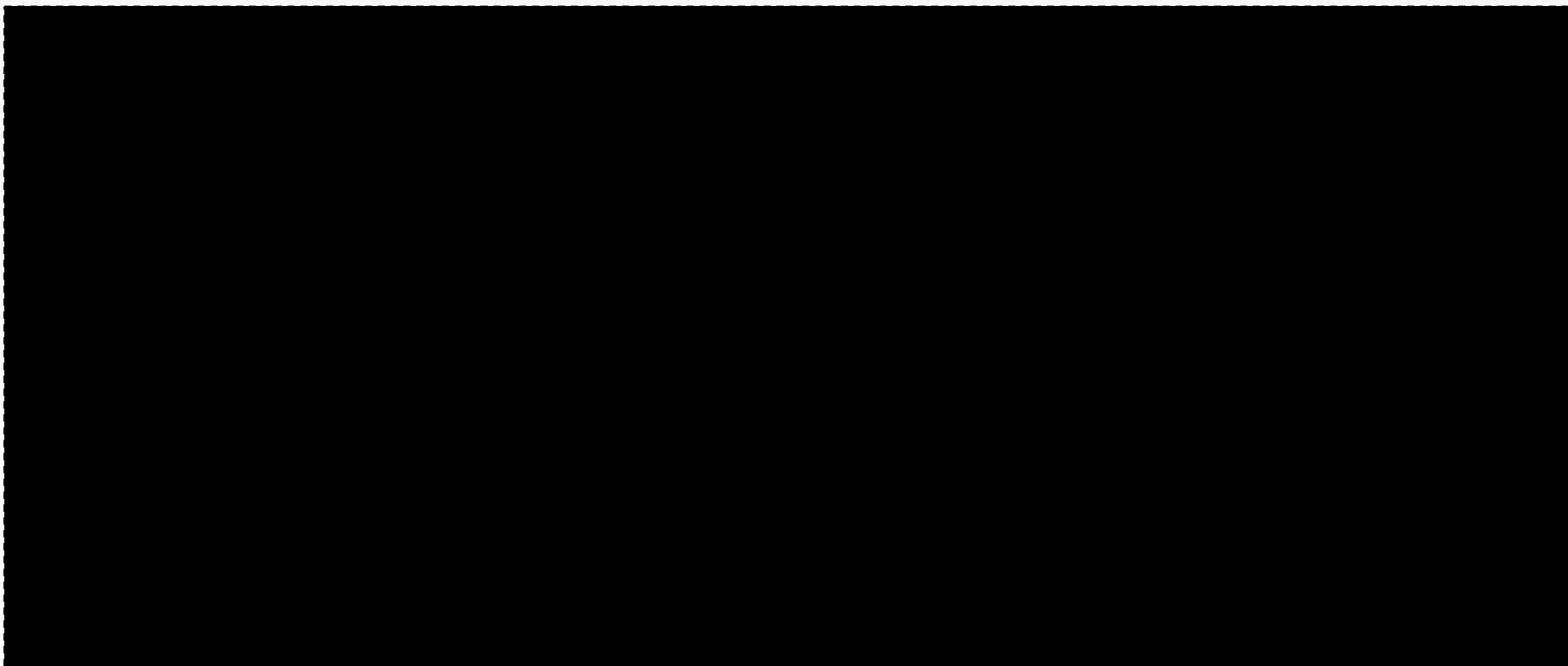
```
----> Starting Kitchen (v1.4.0)
----> Cleaning up any prior instances of <default-ubuntu-1404>
----> Destroying <default-ubuntu-1404>...
      Finished destroying <default-ubuntu-1404> (0m0.00s).
----> Testing <default-ubuntu-1404>
----> Creating <default-ubuntu-1404>...
      Sending build context to Docker daemon  2.56 kB
      Sending build context to Docker daemon
      Step 0 : FROM ubuntu:14.04
      ...

```

Return Home



```
$ cd ~
```



v3.0.0

Apply the cookbook



```
$ sudo chef-client --local-mode -r "recipe[apache]"
```

```
[2015-05-12T05:09:50+00:00] WARN: No config file found or specified on command line, using  
command line options.  
Starting Chef Client, version 12.3.0  
resolving cookbooks for run list: ["apache"]  
Synchronizing Cookbooks:  
  - apache  
Compiling Cookbooks...  
Converging 3 resources  
Recipe: apache::server  
  * apt_package[apache2] action install  
  
  * template[/var/www/html/index.html] action create  
    - update content in file /var/www/html/index.html from 317f72 to 7bc72d
```

Update the cookbook's patch number

```
~/cookbooks/apache/metadata.rb
```

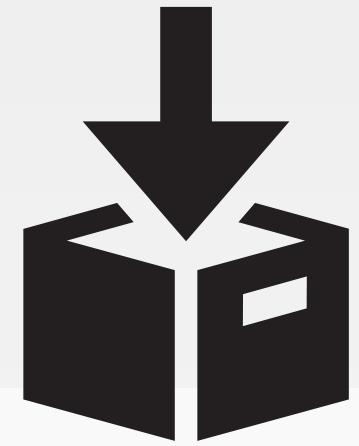
```
name          'apache'
maintainer    'The Authors'
maintainer_email 'you@example.com'
license        'all_rights'
description    'Installs/Configures apache'
long_description 'Installs/Configures apache'
version        '0.2.1'
```

v3.0.0



COMMIT

Commit the Changes



```
$ cd ~/cookbooks/apache  
$ git add .  
$ git status  
$ git commit -m "Changed file resource to  
template resource and defined a template"
```

EXERCISE

Use the Template



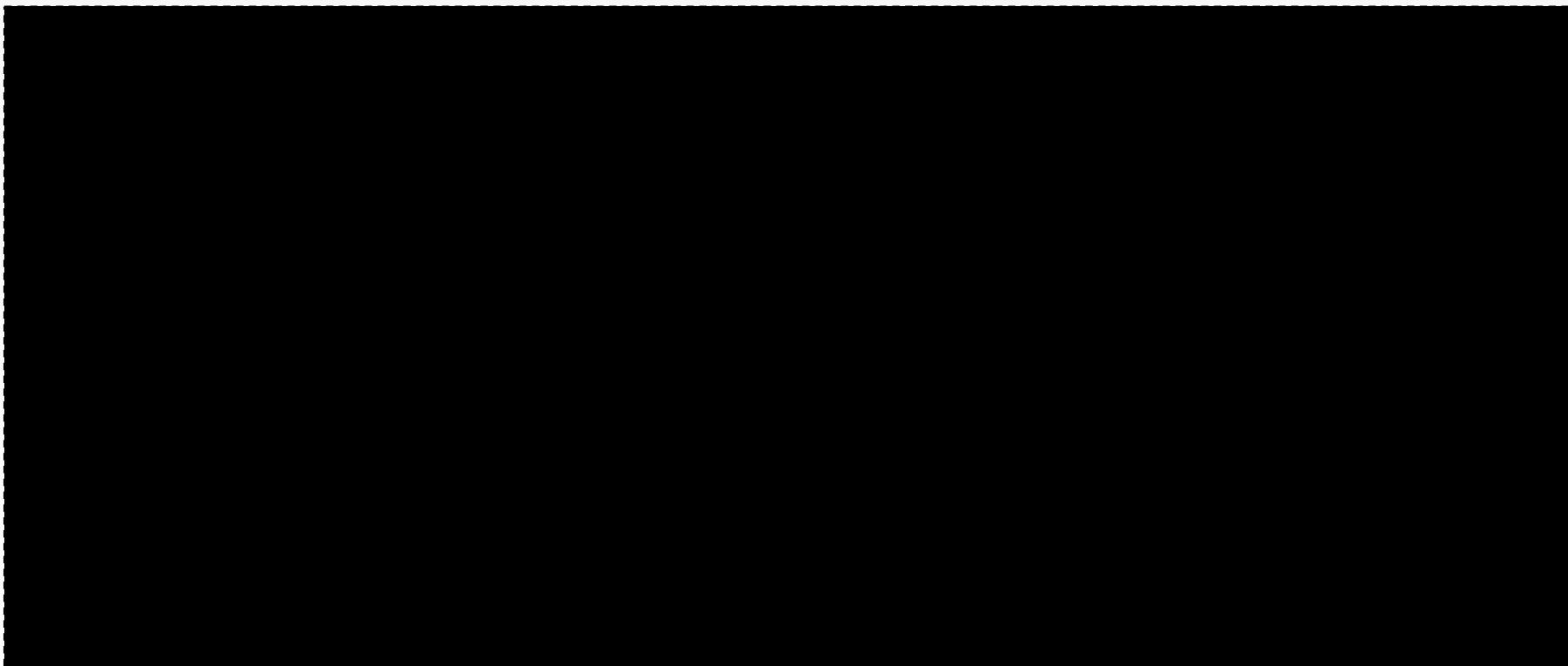
For the "workstation" cookbook:

- Use chef generate to create a template named "motd.erb".
- Copy the source attribute from the file named "/etc/motd" into the template file "motd.erb"
- Remove a resource: The file named "/etc/motd"
- Add a resource: The template named "/etc/motd" is created with the source "motd.erb"
- Use kitchen test to test it and chef-client to locally apply the default recipe.

Return Home



```
$ cd ~
```



v3.0.0

Generate the template



```
$ chef generate cookbook template cookbooks/workstation motd
```

```
Compiling Cookbooks...
Recipe: code_generator::template
  * directory[cookbooks/workstation/templates/default] action create
    - create new directory cookbooks/workstation/templates/default
  * template[cookbooks/workstation/templates/default/motd.erb] action create
    - create new file cookbooks/workstation/templates/default/motd.erb
    - update content in file cookbooks/workstation/templates/default/motd.erb from non
e to e3b0c4
  (diff output suppressed by config)
```

Copy the existing source into the template

```
~/cookbooks/workstation/templates/default/motd.erb
```

Property of ...

```
IPADDRESS: #{node["ipaddress"]}  
HOSTNAME : #{node["hostname"]}  
MEMORY    : #{node["memory"]["total"]}  
CPU       : #{node["cpu"]["0"]["mhz"]}
```

Update the motd.erb to use ERB

```
[ ] ~/cookbooks/workstation/templates/default/motd.erb
```

Property of ...

```
IPADDRESS: <%= node["ipaddress"] %>  
HOSTNAME : <%= node["hostname"] %>  
MEMORY   : <%= node["memory"]["total"] %>  
CPU       : <%= node["cpu"]["0"]["mhz"] %>
```

Remove the file resource

```
~/cookbooks/workstation/recipes/setup.rb
```

```
file "/etc/motd" do
  content "Property of ...
  IPADDRESS: #{node["ipaddress"]}
  HOSTNAME : #{node["hostname"]}
  MEMORY    : #{node["memory"]["total"]}
  CPU        : #{node["cpu"]["0"]["mhz"]}
  "
  mode "0644"
  owner "root"
  group "root"
end
```

Replace it with the template resource

```
~/cookbooks/workstation/recipes/setup.rb
```

```
template "/etc/motd" do
  source "motd.erb"
  mode "0644"
  owner "root"
  group "root"
end
```

EXERCISE

Update the Version



- Update the "workstation" cookbook's version for this patch

- Commit the changes to the "workstation" cookbook to version control

Update the cookbook's patch number

```
~/cookbooks/workstation/metadata.rb
```

```
name          'workstation'  
maintainer    'The Authors'  
maintainer_email 'you@example.com'  
license        'all_rights'  
description    'Installs/Configures workstation'  
long_description 'Installs/Configures workstation'  
version        '0.2.1'
```

v3.0.0



COMMIT

Commit the Changes



```
$ cd ~/cookbooks/workstation  
$ git add .  
$ git status  
$ git commit -m "Changed file resource to  
template resource and defined a template"
```

DISCUSSION

Questions



What questions can we help you answer?

- resources (file, cookbook_file, template, and remote_file)
- templates
- ERB
- Angry Squids



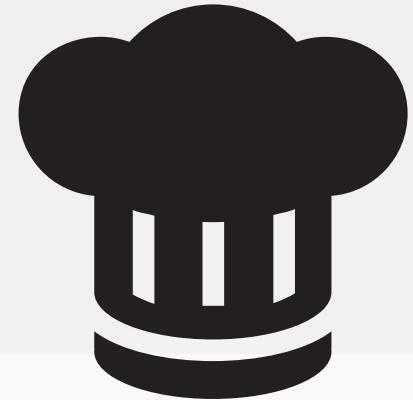
Workstation Installation

v3.0.0



LAB

Install the ChefDK



Now it's time to install the tools on your system

OBJECTIVE:

- Install the ChefDK
- Open a Terminal / Command Prompt
- Execute a series of commands to ensure everything is installed
- Download a repository of cookbooks
- Install git (optional)
- Install a text editor (optional)

CONCEPT

ChefDK



The omnibus installer is used to set up the Chef development kit on a workstation, including the chef-client itself, an embedded version of Ruby, RubyGems, OpenSSL, key-value stores, parsers, libraries, command line utilities, and community tools such as Kitchen, Berkshelf, and ChefSpec.

v3.0.0

<https://downloads.chef.io/chef-dk/>

 CHEF™
CODE CAN

EXERCISE

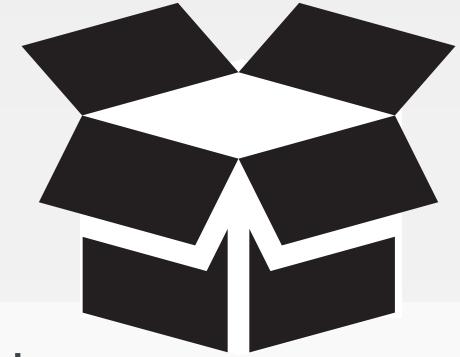
Run All These Commands



```
$ chef --version  
$ chef-client --version  
$ knife --version  
$ ohai --version  
$ berks --version  
$ kitchen --version  
$ foodcritic --version  
$ rubocop --version
```

CONCEPT

Download Your Work



A copy of the work that you completed in this workshop can be found on GitHub.

v3.0.0

<https://github.com/chef-training/chefdk-fundamentals-repo>

 CHEF™
CODE CAN

CONCEPT

git



Git is a distributed revision control system with an emphasis on speed, data integrity, and support for distributed, non-linear workflows. Git was initially designed and developed by Linus Torvalds for Linux kernel development in 2005, and has since become the most widely adopted version control system for software development.

CONCEPT

Sublime Text



Sublime Text is a sophisticated text editor for code, markup and prose.

You'll love the slick user interface, extraordinary features and amazing performance.

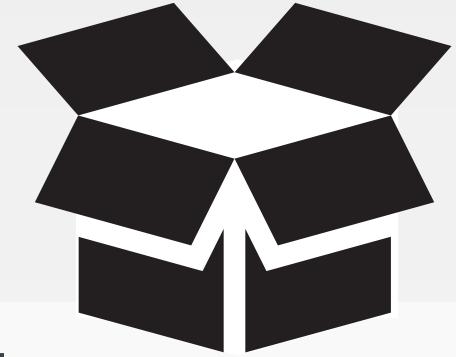
v3.0.0

<http://www.sublimetext.com>

 **CHEF**TM
CODE CAN

CONCEPT

ATOM Editor



At GitHub, we're building the text editor we've always wanted. A tool you can customize to do anything, but also use productively on the first day without ever touching a config file. Atom is modern, approachable, and hackable to the core. We can't wait to see what you build with it.

v3.0.0

<https://atom.io>

 CHEF™
CODE CAN



Review Day One



Chef Fundamentals by [Chef Software, Inc.](#) is licensed under a [Creative Commons Attribution-ShareAlike 4.0 International License](#).

v3.0.0



DISCUSSION

What questions can we answer for you?



- resources
- test-and-repair
- chef-apply
- recipes
- cookbooks
- version control with git
- chef-client in local mode
- chef command

DISCUSSION

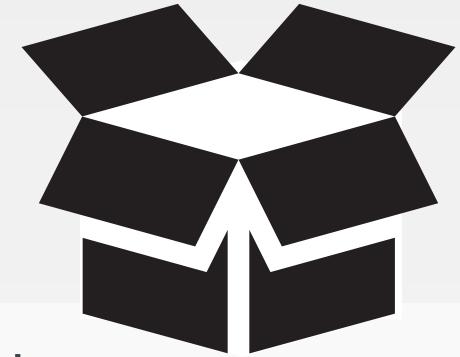
What questions can we answer for you?



- Test Kitchen
- Serverspec
- ohai
- node attributes
- String interpolation
- ERB
- Semantic versioning

CONCEPT

Download Your Work



A copy of the work that you completed in this workshop can be found on GitHub.

 [chef-training / chefdk-fundamentals-repo](#) Watch 18

The cookbooks created after the first day of completing the Chef DK Fundamentals

16 commits 2 branches 0 releases 1 contributor

 Branch: **rhel**  [chefdk-fundamentals-repo / +](#) 

An orange arrow points from the text "rhel branch!" to the "Branch: rhel" dropdown menu on the GitHub repository page.

v3.0.0

<https://github.com/chef-training/chefdk-fundamentals-repo>

 **CHEF**TM
CODE CAN



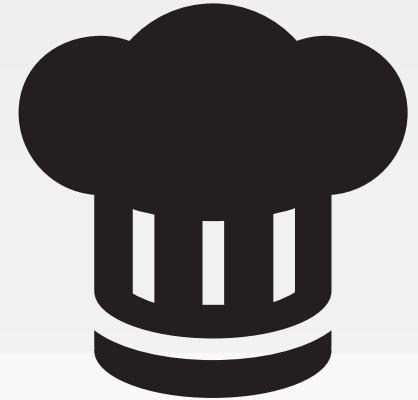
The Chef Server

v3.0.0



LAB

Hosted Chef



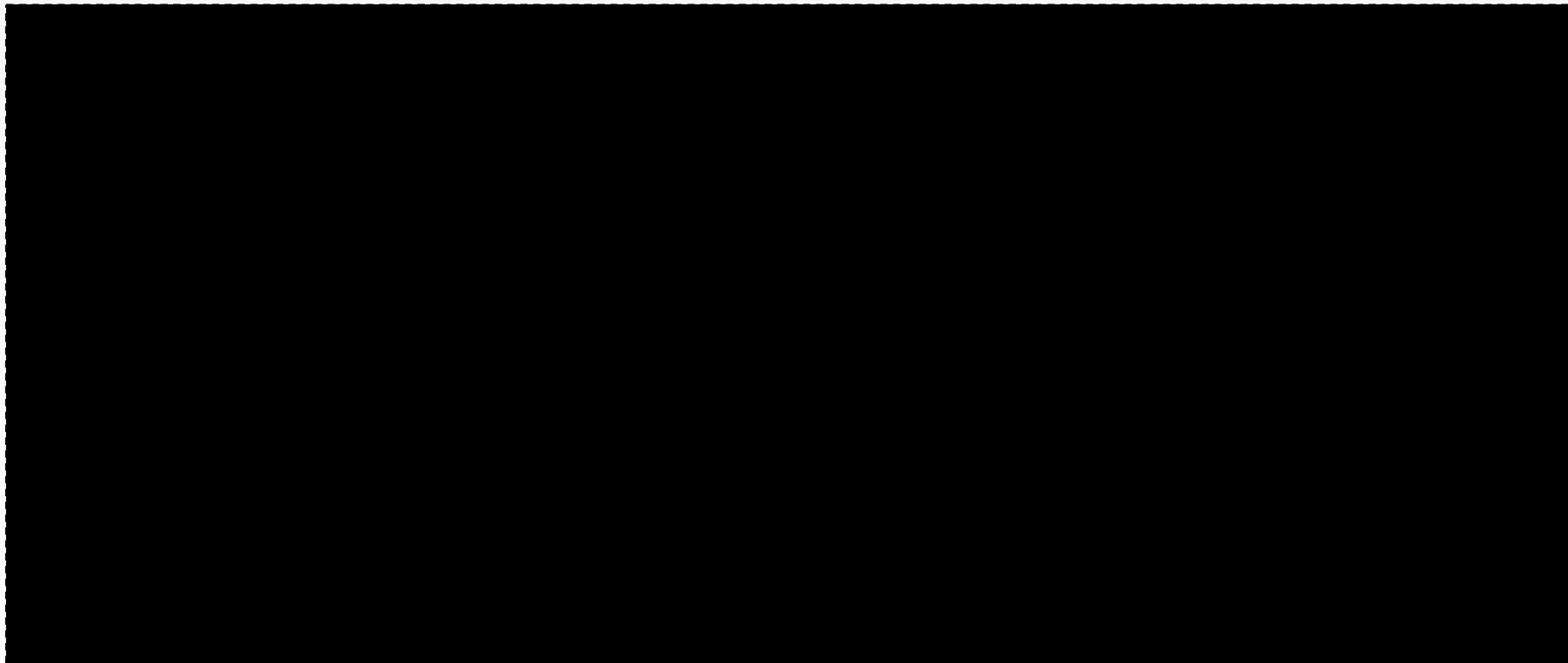
More easily manage multiple nodes

OBJECTIVE:

- Create a Hosted Chef Account
- Upload our cookbooks to the Hosted Chef Server
- Add our old workstation as a managed node



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



v3.0.0



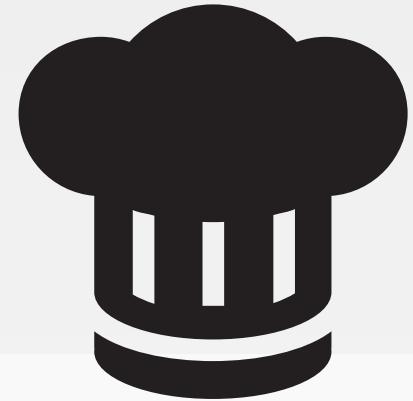
```
$ knife client list
```

```
ORGNAME-validator
```

v3.0.0

LAB

Hosted Chef



More easily manage multiple nodes

OBJECTIVE:

- ✓ Create a Hosted Chef Account
- Upload our cookbooks to the Hosted Chef Server
- Add our old workstation as a managed node



```
$ knife cookbook --help
```

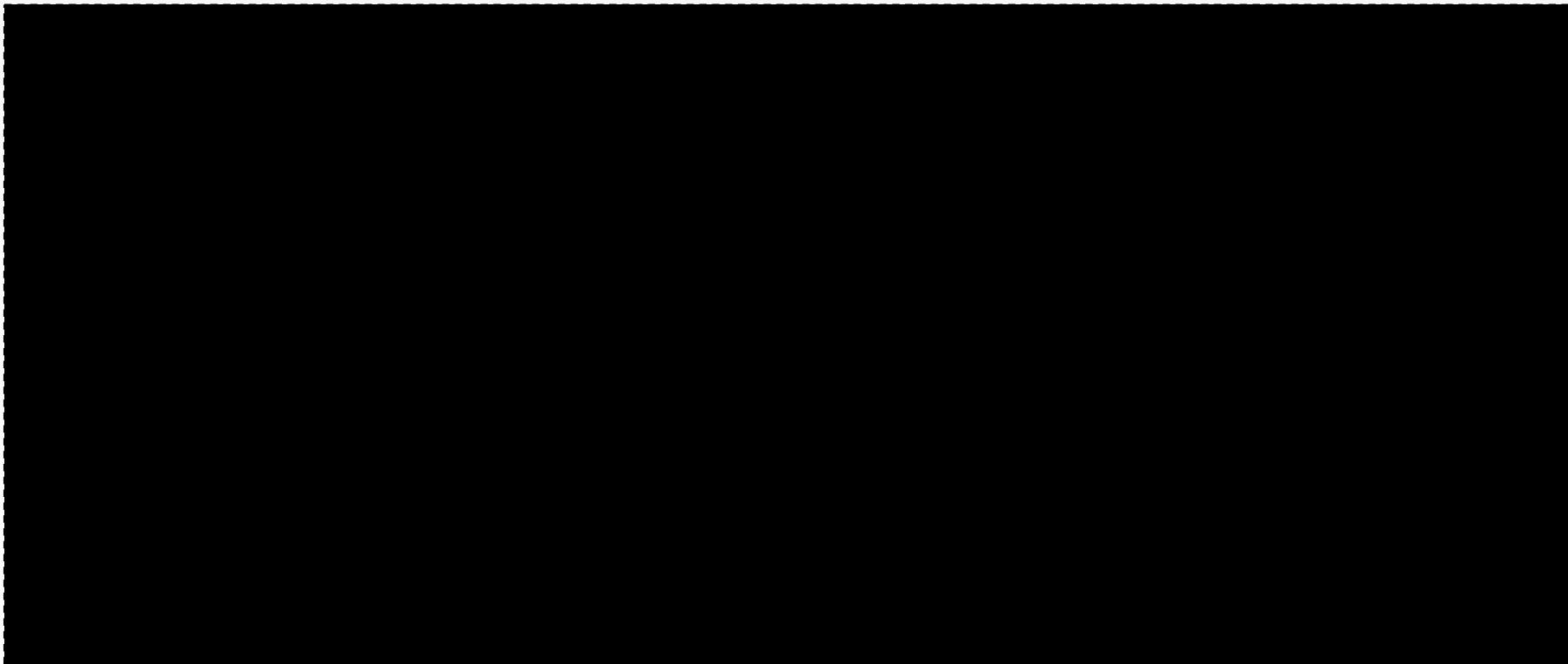
```
** COOKBOOK COMMANDS **

knife cookbook bulk delete REGEX (options)
knife cookbook create COOKBOOK (options)
knife cookbook delete COOKBOOK VERSION (options)
knife cookbook download COOKBOOK [VERSION] (options)
knife cookbook list (options)
knife cookbook metadata COOKBOOK (options)
knife cookbook metadata from FILE (options)
knife cookbook show COOKBOOK [VERSION] [PART] [FILENAME] (options)
knife cookbook test [COOKBOOKS...] (options)
knife cookbook upload [COOKBOOKS...] (options)
```

List cookbooks on the Chef Server



```
$ knife cookbook list
```



v3.0.0

Upload the apache cookbook



```
$ knife cookbook upload apache
```

```
Uploading apache [0.2.1]
```

```
Uploaded 1 cookbook.
```

List cookbooks on the Chef Server



```
$ knife cookbook list
```

```
apache      0.2.1
```

EXERCISE

Upload Cookbook



- Upload your remaining cookbooks

- Verify that are all uploaded

Upload the apache cookbook



```
$ knife cookbook upload workstation
```

```
Uploading workstation [0.2.1]
```

```
Uploaded 1 cookbook.
```

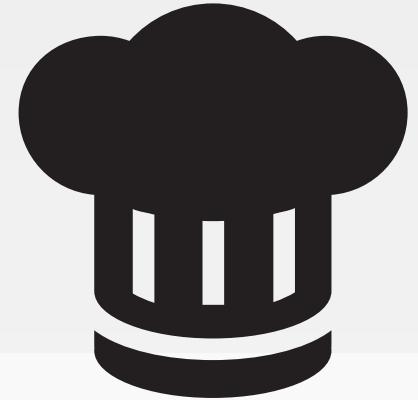


```
$ knife cookbook list
```

```
apache      0.2.1
workstation 0.2.1
```

LAB

Hosted Chef



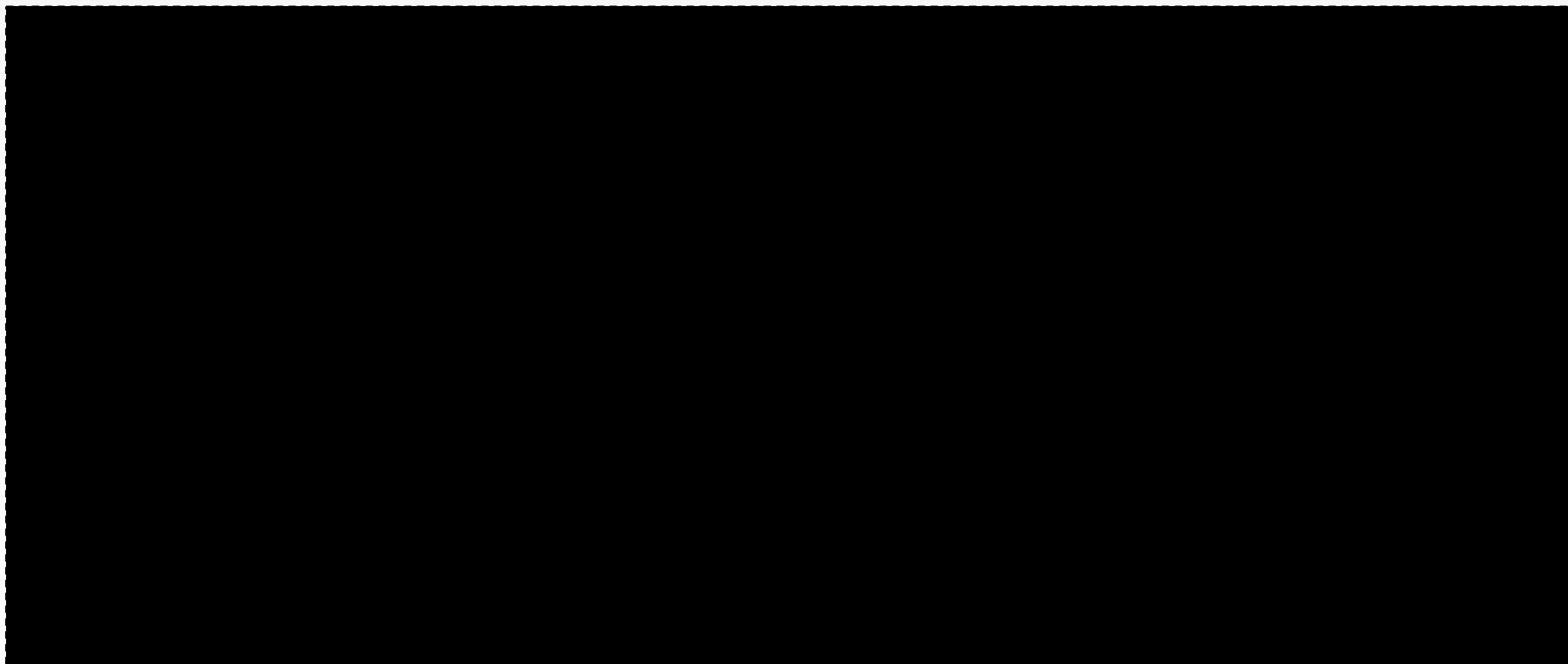
More easily manage multiple nodes

OBJECTIVE:

- ✓ Create a Hosted Chef Account
- ✓ Upload our cookbooks to the Hosted Chef Server
- Add our old workstation as a managed node



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



v3.0.0



```
$ knife bootstrap FQDN -x USER -P PWD --sudo -N node1
```

```
Connecting to ec2-204-236-155-223.us-west-1.compute.amazonaws.com
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Starting first Chef Client run...
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Starting Chef Client, version 12.3.0
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Creating a new client identity for node1
using the validator key.

ec2-204-236-155-223.us-west-1.compute.amazonaws.com resolving cookbooks for run list: []
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Synchronizing Cookbooks:
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Compiling Cookbooks...
ec2-204-236-155-223.us-west-1.compute.amazonaws.com [2015-05-15T21:59:18+00:00] WARN: Node
node1 has an empty run list.

ec2-204-236-155-223.us-west-1.compute.amazonaws.com Converging 0 resources
ec2-204-236-155-223.us-west-1.compute.amazonaws.com
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Running handlers:
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Running handlers complete
```

v3.0.0

 CHEF™
CODE CAN



```
$ knife node --help
```

```
FATAL: Cannot find sub command for: 'node --help'  
Available node subcommands: (for details, knife SUB-COMMAND --help)  
  
** NODE COMMANDS **  
knife node bulk delete REGEX (options)  
knife node create NODE (options)  
knife node delete NODE (options)  
knife node edit NODE (options)  
knife node environment set NODE ENVIRONMENT  
knife node from file FILE (options)  
knife node list (options)  
knife node run_list add [NODE] [ENTRY[,ENTRY]] (options)  
knife node run_list remove [NODE] [ENTRY[,ENTRY]] (options)
```



```
$ knife node list
```

```
node1
```

v3.0.0

 CHEF™
CODE CAN



```
$ knife node show node1
```

```
Node Name:    node1
Environment:  _default
FQDN:        ip-10-198-51-26.us-west-1.compute.internal
IP:          204.236.155.223
Run List:
Roles:
Recipes:
Platform:    ubuntu 14.04
Tags:
```

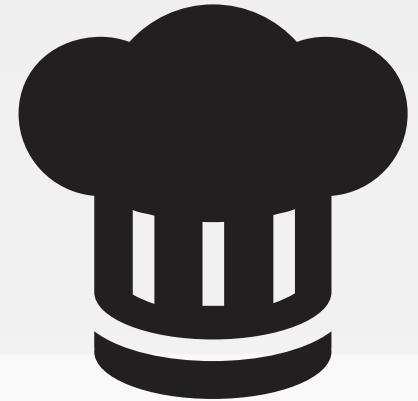


```
$ knife node run_list add node1 "recipe[apache]"
```

```
node1:  
  run_list: recipe[apache]
```

LAB

Hosted Chef



More easily manage multiple nodes

OBJECTIVE:

- ✓ Create a Hosted Chef Account
- ✓ Upload our cookbooks to the Hosted Chef Server
- ✓ Add our old workstation as a managed node

DISCUSSION

Discussion



What questions can we help you answer?



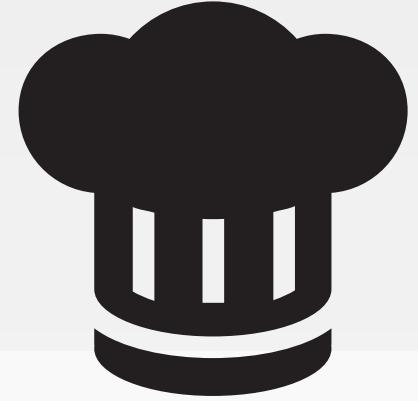
Community Cookbooks

v3.0.0



LAB

Proxy Server



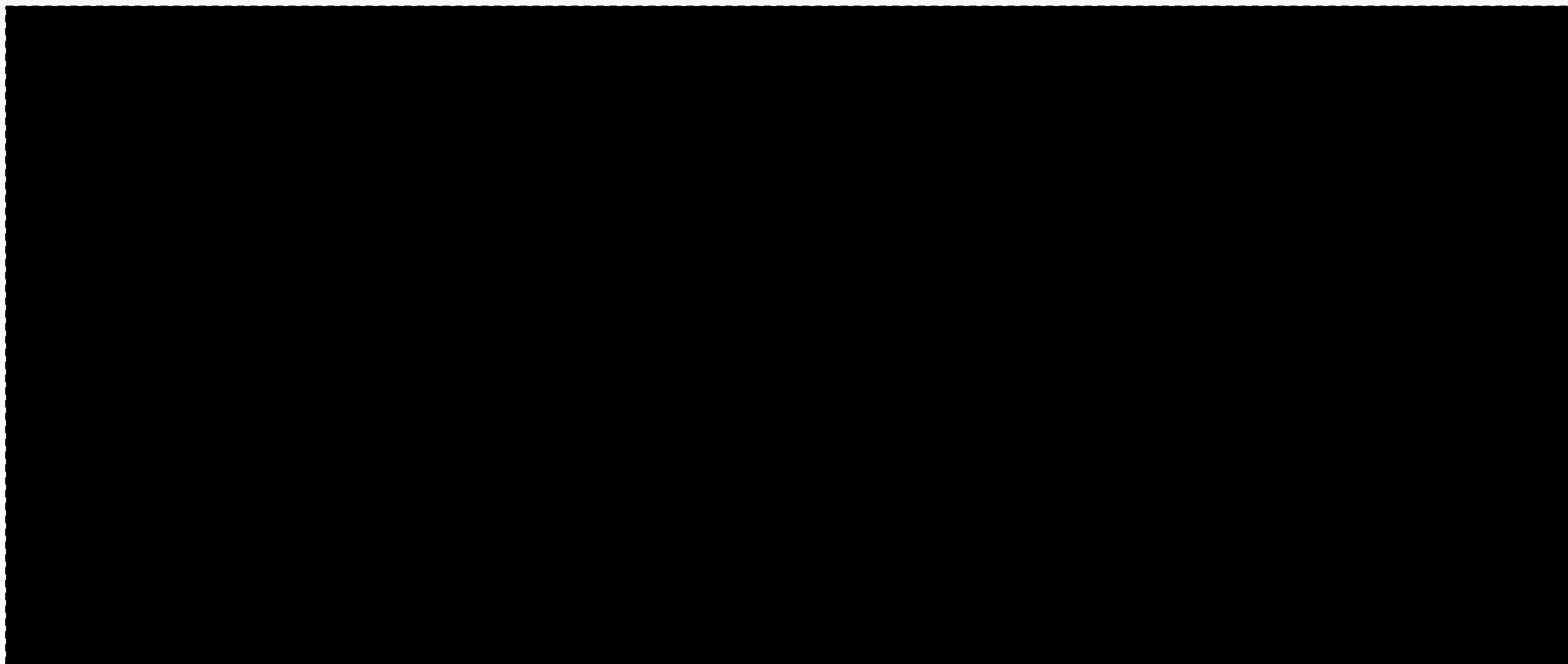
Adding a proxy will allow us to better grow our infrastructure.

OBJECTIVE:

- Find or Create a Cookbook to Manage a Proxy Server
- Configure the Proxy Server to send traffic to the new node
- Upload cookbook to Chef Server
- Bootstrap a new node that runs the Proxy Server cookbook



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



v3.0.0



```
$ chef generate cookbook cookbooks/myhaproxy
```

```
Compiling Cookbooks...
```

```
Recipe: code_generator::cookbook
```

```
* directory[/Users/webber/chef-repo/cookbooks/myhaproxy] action create
  - create new directory /Users/webber/chef-repo/cookbooks/myhaproxy
* template[/Users/webber/chef-repo/cookbooks/myhaproxy/metadata.rb] action
create_if_missing
  - create new file /Users/webber/chef-repo/cookbooks/myhaproxy/metadata.rb
  - update content in file /Users/webber/chef-repo/cookbooks/myhaproxy/metadata.rb from
none to 6d9e05
  (diff output suppressed by config)
* template[/Users/webber/chef-repo/cookbooks/myhaproxy/README.md] action create_if_missing
  - create new file /Users/webber/chef-repo/cookbooks/myhaproxy/README.md
  - update content in file /Users/webber/chef-repo/cookbooks/myhaproxy/README.md from none
to aa15b1
```



```
~/chef-repo/cookbooks/myhaproxy/metadata.rb
```

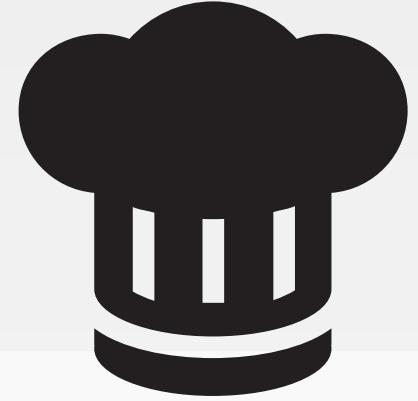
```
name          'myhaproxy'  
maintainer    'The Authors'  
maintainer_email 'you@example.com'  
license        'all_rights'  
description    'Installs/Configures myhaproxy'  
long_description 'Installs/Configures myhaproxy'  
version        '0.1.0'  
  
depends 'haproxy', '= 1.6.6'
```

v3.0.0



LAB

Proxy Server



Adding a proxy will allow us to better grow our infrastructure.

OBJECTIVE:

- ✓ Find or Create a Cookbook to Manage a Proxy Server
- ❑ Configure the Proxy Server to send traffic to the new node
- ❑ Upload cookbook to Chef Server
- ❑ Bootstrap a new node that runs the Proxy Server cookbook



```
$ knife node show --help
```

```
knife node show NODE (options)
  -a ATTR1 [--attribute ATTR2] ,    Show one or more attributes
        --attribute
  -s, --server-url URL          Chef Server URL
        --chef-zero-host HOST      Host to start chef-zero on
        --chef-zero-port PORT     Port (or port range) to start chef-zero on. Port ranges
  -k, --key KEY                 API Client Key
        --[no-]color              Use colored output, defaults to false on Windows, true
  -c, --config CONFIG           The configuration file to use
        --defaults                Accept default values for all questions
  -d, --disable-editing         Do not open EDITOR, just accept the data as is
  -e, --editor EDITOR          Set the editor to use for interactive commands
  -E, --environment            Show only the Chef environment
```



```
$ knife node show node1 -a ipaddress
```

```
node2:  
ipaddress: 172.31.15.124
```

Amazon EC2 Instances



The ipaddress and hostname are unfortunately not how we can address these nodes within our recipes.



```
$ knife node show node1 -a cloud
```

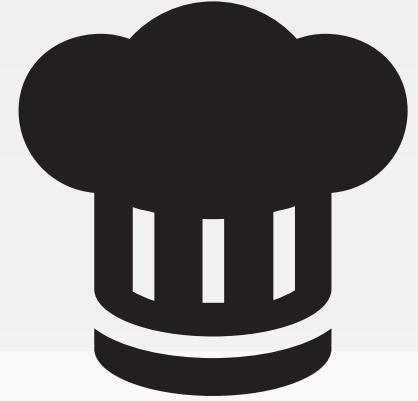
```
node2:  
  cloud:  
    local_hostname: ip-172-31-15-124.us-west-1.compute.internal  
    local_ipv4: 172.31.15.124  
    private_ips: 172.31.15.124  
    provider: ec2  
    public_hostname: ec2-52-8-71-11.us-west-1.compute.amazonaws.com  
    public_ips: 52.8.71.11  
    public_ipv4: 52.8.71.11
```

```
[] ~./chef-repo/cookbooks/myhaproxy/recipes/default.rb
```

```
#  
# Cookbook Name:: myhaproxy  
# Recipe:: default  
#  
# Copyright (c) 2015 The Authors, All Rights Reserved.  
  
node.default['haproxy']['members'] = [{  
    "hostname" => "ec2-52-8-71-11.us-west-1.compute.amazonaws.com",  
    "ipaddress" => "52.8.71.11",  
    "port" => 80,  
    "ssl_port" => 80  
}]  
  
include_recipe "haproxy::default"
```

LAB

Proxy Server



Adding a proxy will allow us to better grow our infrastructure.

OBJECTIVE:

- ✓ Find or Create a Cookbook to Manage a Proxy Server
- ✓ Configure the Proxy Server to send traffic to the new node
- ❑ Upload cookbook to Chef Server
- ❑ Bootstrap a new node that runs the Proxy Server cookbook

EXERCISE

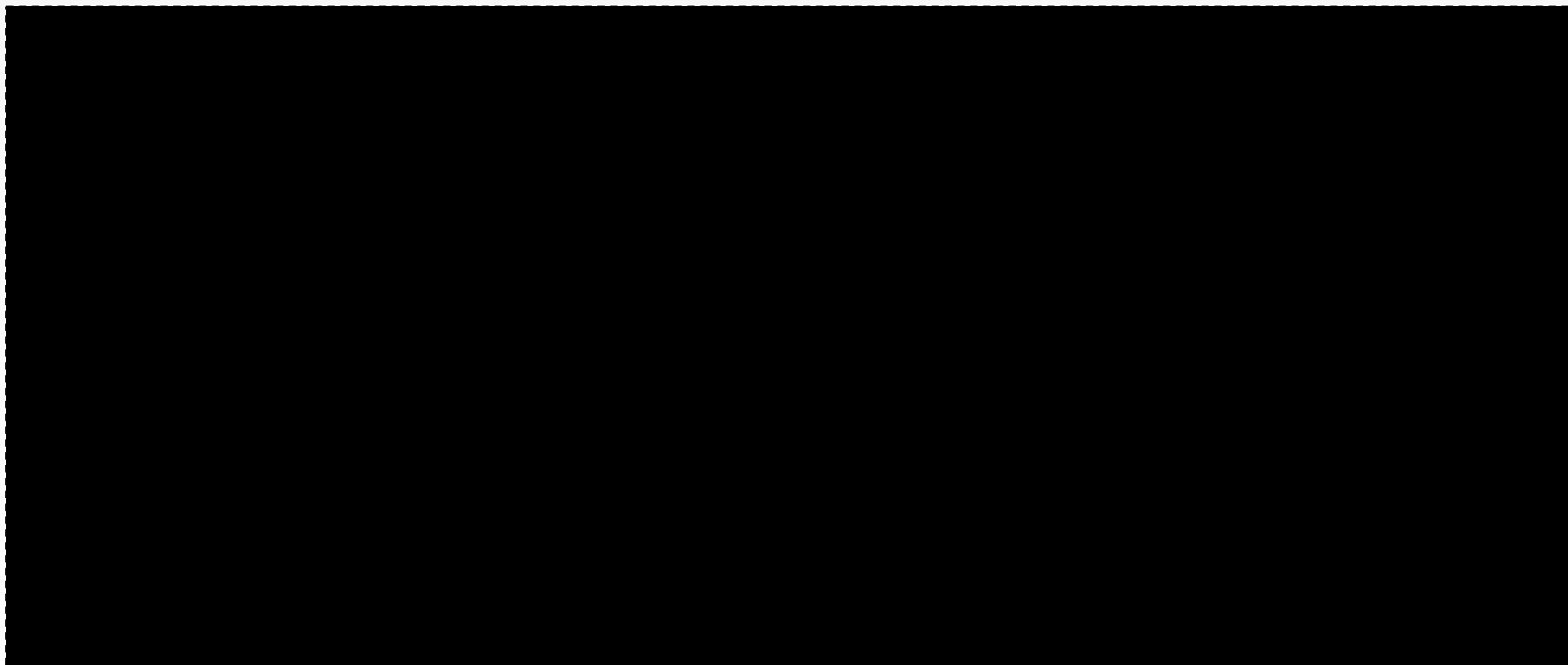
Upload a Cookbook



- Upload the proxy server cookbook to the Chef Server



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



v3.0.0



```
$ berks install
```

```
Resolving cookbook dependencies...
Fetching 'myhaproxy' from source at .
Fetching cookbook index from https://supermarket.chef.io...
Using build-essential (2.2.3)
Using haproxy (1.6.6)
Using myhaproxy (0.1.0) from source at .
Using cpu (0.2.0)
```



```
$ berks upload
```

```
Uploaded build-essential (2.2.3) to: 'https://api.opscode.com:443/organizations/vogue'  
Uploaded cpu (0.2.0) to: 'https://api.opscode.com:443/organizations/vogue'  
Uploaded haproxy (1.6.6) to: 'https://api.opscode.com:443/organizations/vogue'  
Uploaded myhaproxy (0.1.0) to: 'https://api.opscode.com:443/organizations/vogue'
```

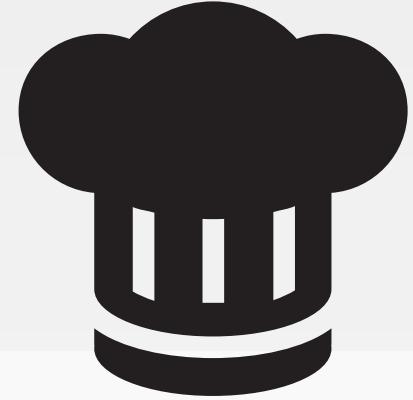


```
$ knife cookbook list
```

apache	0.2.1
build-essential	2.2.3
cpu	0.2.0
haproxy	1.6.6
myhaproxy	0.1.0
workstation	0.2.1

LAB

Proxy Server



Adding a proxy will allow us to better grow our infrastructure.

OBJECTIVE:

- ✓ Find or Create a Cookbook to Manage a Proxy Server
- ✓ Configure the Proxy Server to send traffic to the new node
- ✓ Upload cookbook to Chef Server
- Bootstrap a new node that runs the Proxy Server cookbook

EXERCISE

Bootstrap a Proxy Server



- Bootstrap a new node
- Update the run list of the new node to include the proxy server cookbook
- Login to that system and run chef-client
- Verify that traffic to the proxy is related to the web server.

Your Second Node



<http://bit.ly/1W3AgQn>

v3.0.0

 CHEF™
CODE CAN



```
$ knife bootstrap FQDN2 -x USER -P PWD --sudo -N node2
```

```
Connecting to ec2-50-18-19-208.us-west-1.compute.amazonaws.com
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Starting first Chef Client run...
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Starting Chef Client, version 12.3.0
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Creating a new client identity for node2
using the validator key.

ec2-50-18-19-208.us-west-1.compute.amazonaws.com resolving cookbooks for run list: []
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Synchronizing Cookbooks:
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Compiling Cookbooks...
ec2-50-18-19-208.us-west-1.compute.amazonaws.com [2015-05-15T22:28:58+00:00] WARN: Node node2
has an empty run list.

ec2-50-18-19-208.us-west-1.compute.amazonaws.com Converging 0 resources
ec2-50-18-19-208.us-west-1.compute.amazonaws.com
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Running handlers:
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Running handlers: [RunListHandler]
```

v3.0.0

 CHEF™
CODE CAN



```
$ knife node show node2
```

```
Node Name:    node2
Environment:  _default
FQDN:        ip-10-198-31-238.us-west-1.compute.internal
IP:          50.18.19.208
Run List:
Roles:
Recipes:      myhaproxy, myhaproxy::default, haproxy::default, haproxy::install_package
Platform:     ubuntu 14.04
Tags:
```



```
$ knife node run_list add node2 "recipe[myhaproxy]"
```

```
node2:  
  run_list: recipe[myhaproxy]
```



```
$ knife node show node2
```

```
Node Name:    node2
Environment:  _default
FQDN:        ip-10-198-31-238.us-west-1.compute.internal
IP:          50.18.19.208
Run List:    recipe[myhaproxy]
Roles:
Recipes:     myhaproxy, myhaproxy::default, haproxy::default, haproxy::install_package
Platform:    ubuntu 14.04
Tags:
```

SSH Woes



Logging into both systems is a pain. We can use another knife tool to allow us to send commands to all of our nodes.



```
$ knife ssh --help
```

```
knife ssh QUERY COMMAND (options)
  -a, --attribute ATTR          The attribute to use for opening the connection -
  default depends on the context
  -s, --server-url URL         Chef Server URL
  --chef-zero-host HOST         Host to start chef-zero on
  --chef-zero-port PORT         Port (or port range) to start chef-zero on. Port ranges
like 1000,1010 or 8889-9999 will try all given ports until one works.
  -k, --key KEY                API Client Key
  --[no-]color                  Use colored output, defaults to false on Windows, true
otherwise
  -C, --concurrency NUM        The number of concurrent connections
  -c, --config CONFIG          The configuration file to use
  --defaults                   Accept default values for all questions
  -d, --disable-editing         Do not open EDITOR just accept the data as is
```

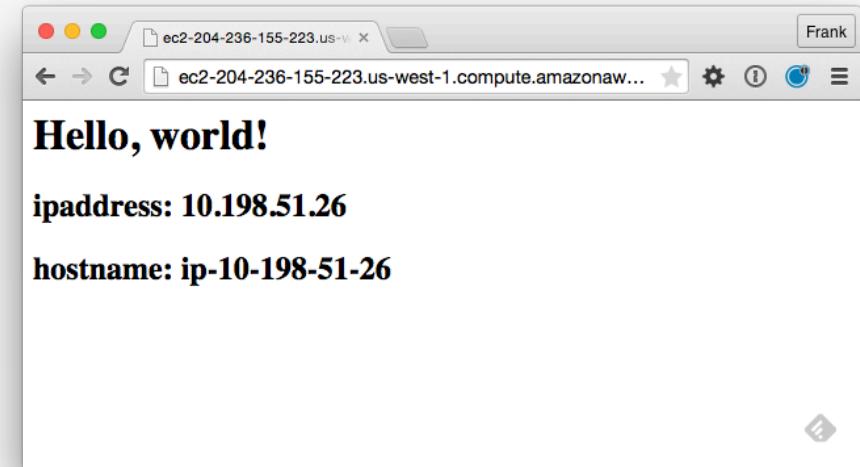
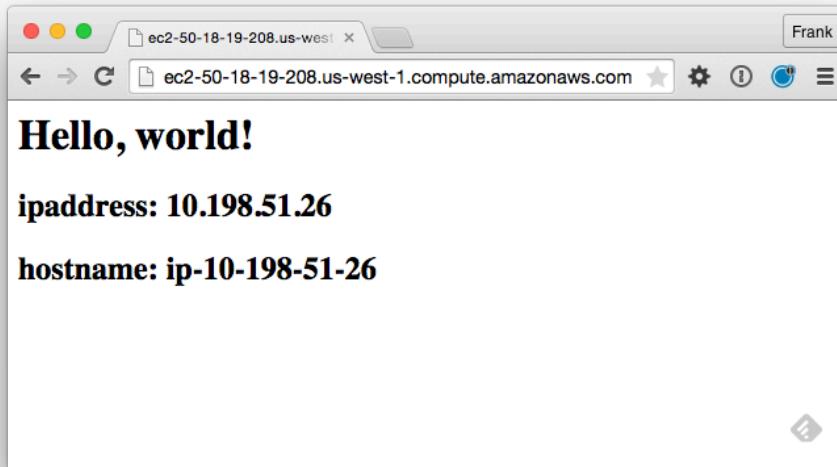


```
$ knife ssh "*:*" -x USERNAME -P PASSWORD "sudo chef-client"
```

```
ec2-50-18-19-208.us-west-1.compute.amazonaws.com      Starting Chef Client, version 12.3.0
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Starting Chef Client, version 12.3.0
ec2-50-18-19-208.us-west-1.compute.amazonaws.com      resolving cookbooks for run list:
["myhaproxy"]

ec2-204-236-155-223.us-west-1.compute.amazonaws.com resolving cookbooks for run list:
["apache"]

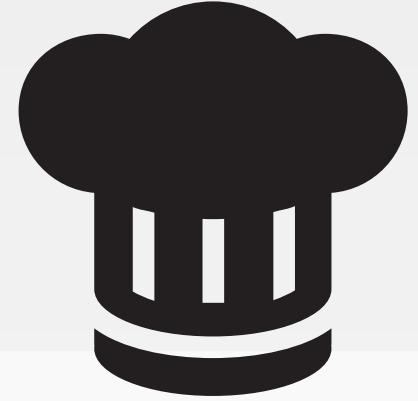
ec2-50-18-19-208.us-west-1.compute.amazonaws.com      Synchronizing Cookbooks:
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Synchronizing Cookbooks:
ec2-204-236-155-223.us-west-1.compute.amazonaws.com      - apache
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Compiling Cookbooks...
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Converging 4 resources
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Recipe: apache::server
ec2-50-18-19-208.us-west-1.compute.amazonaws.com      - build-essential
```



v3.0.0

LAB

Proxy Server



Adding a proxy will allow us to better grow our infrastructure.

OBJECTIVE:

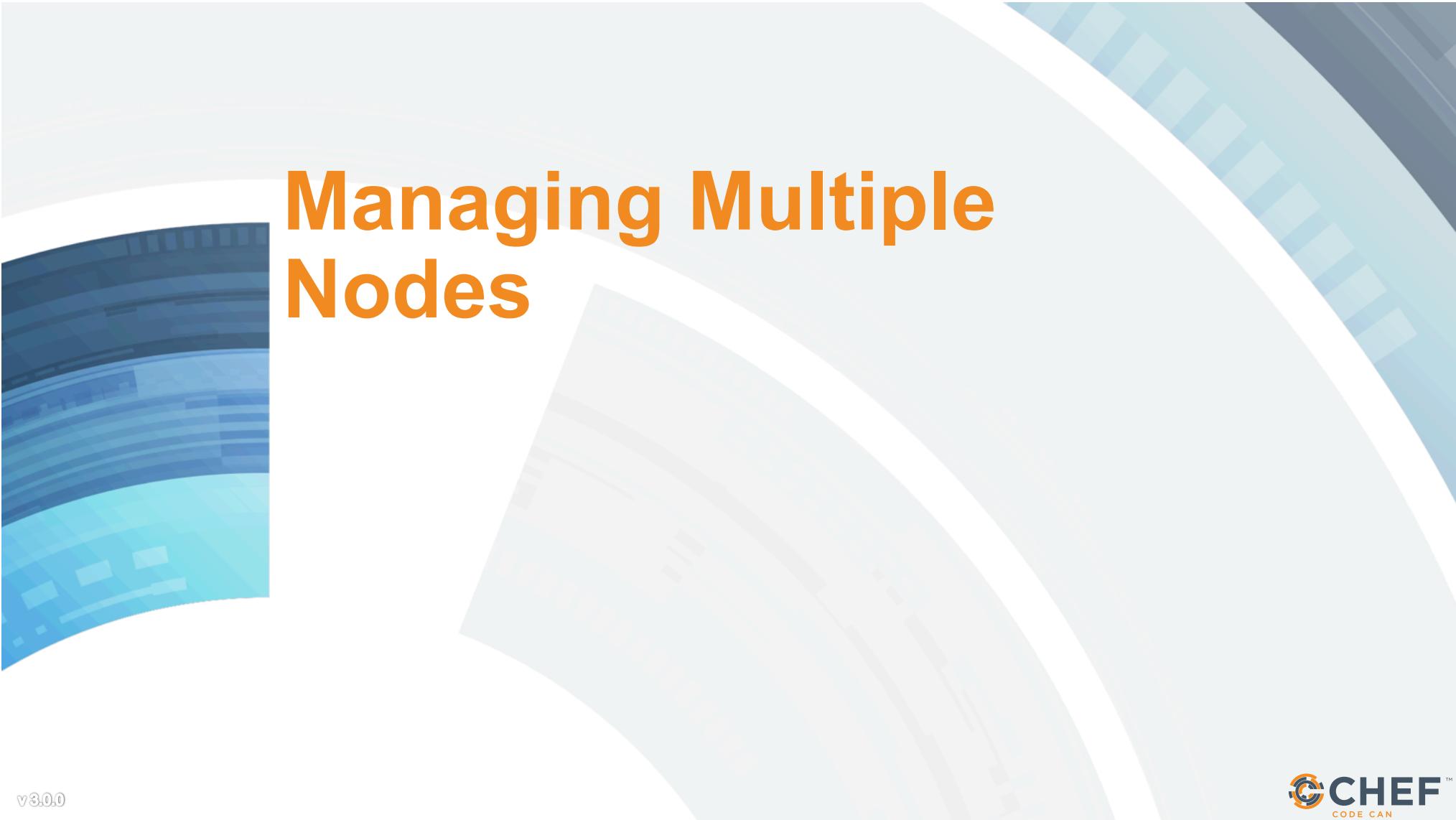
- ✓ Find or Create a Cookbook to Manage a Proxy Server
- ✓ Configure the Proxy Server to send traffic to the new node
- ✓ Upload cookbook to Chef Server
- ✓ Bootstrap a new node that runs the Proxy Server cookbook

DISCUSSION

Discussion



What questions can we help you answer?



Managing Multiple Nodes

v3.0.0



EXERCISE

Another Web Node



- Bootstrap a new node
- Update the run list of the new node to include the web server cookbook
- Login to that system and run chef-client
- Verify that the node's web server is functional

Your Third Node

<http://bit.ly/1W3AgQn>





```
$ knife bootstrap FQDN -x USER -P PWD --sudo -N node3
```

```
Connecting to ec2-54-176-64-173.us-west-1.compute.amazonaws.com
ec2-54-176-64-173.us-west-1.compute.amazonaws.com Starting first Chef Client run...
ec2-54-176-64-173.us-west-1.compute.amazonaws.com Starting Chef Client, version 12.3.0
ec2-54-176-64-173.us-west-1.compute.amazonaws.com Creating a new client identity for node3
using the validator key.

ec2-54-176-64-173.us-west-1.compute.amazonaws.com resolving cookbooks for run list: []
ec2-54-176-64-173.us-west-1.compute.amazonaws.com Synchronizing Cookbooks:
ec2-54-176-64-173.us-west-1.compute.amazonaws.com Compiling Cookbooks...
ec2-54-176-64-173.us-west-1.compute.amazonaws.com [2015-05-15T22:45:40+00:00] WARN: Node
node3 has an empty run list.

ec2-54-176-64-173.us-west-1.compute.amazonaws.com Converging 0 resources
ec2-54-176-64-173.us-west-1.compute.amazonaws.com
ec2-54-176-64-173.us-west-1.compute.amazonaws.com Running handlers:
ec2-54-176-64-173.us-west-1.compute.amazonaws.com Running handlers complete
```

v3.0.0

 CHEF™
CODE CAN



```
$ knife node show node3
```

```
Node Name:    node3
Environment:  _default
FQDN:        ip-10-197-105-148.us-west-1.compute.internal
IP:          54.176.64.173
Run List:
Roles:
Recipes:
Platform:    ubuntu 14.04
Tags:
```



```
$ knife node run_list add node3 "recipe[apache]"
```

```
node3:  
  run_list: recipe[apache]
```



```
$ knife ssh "*:*" -x USERNAME -P PWD "sudo chef-client"
```

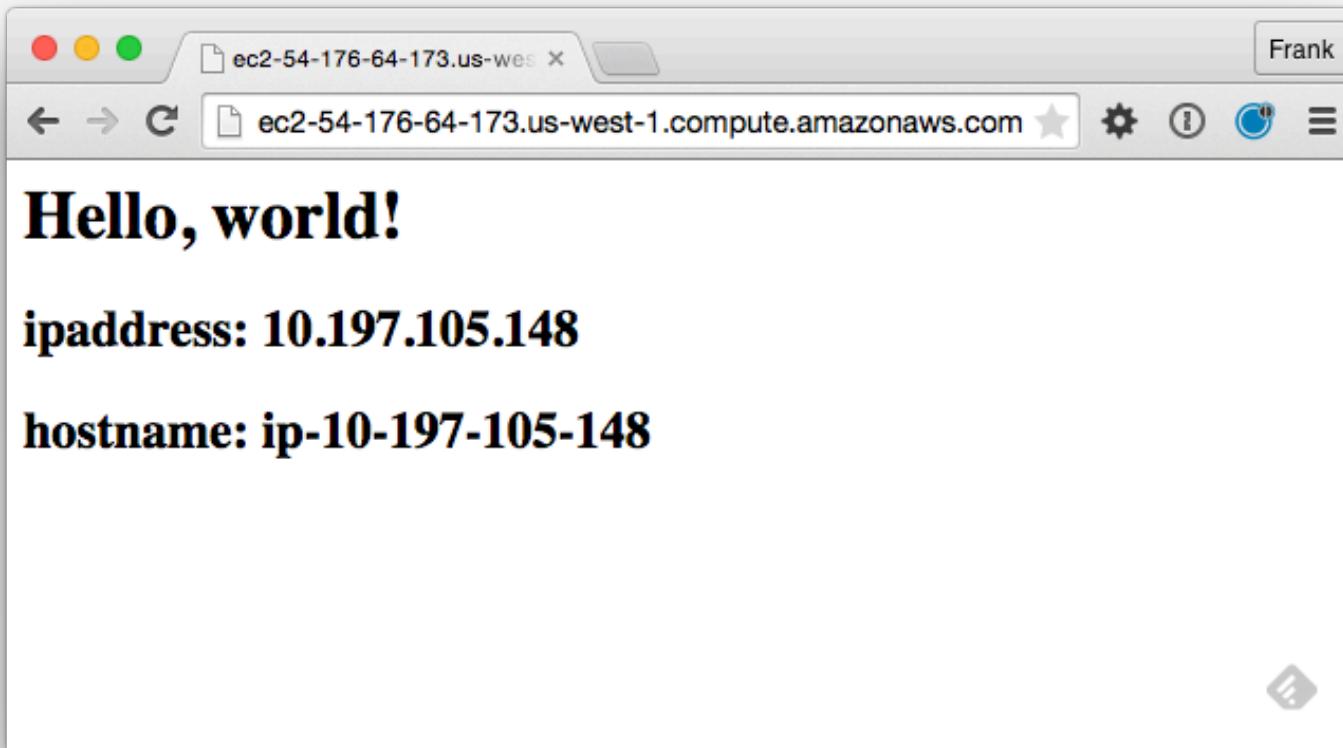
```
ec2-50-18-19-208.us-west-1.compute.amazonaws.com      Starting Chef Client, version 12.3.0
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Starting Chef Client, version 12.3.0
ec2-50-18-19-208.us-west-1.compute.amazonaws.com      resolving cookbooks for run list:
["myhaproxy"]

ec2-204-236-155-223.us-west-1.compute.amazonaws.com resolving cookbooks for run list:
["apache"]

ec2-50-18-19-208.us-west-1.compute.amazonaws.com      Synchronizing Cookbooks:
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Synchronizing Cookbooks:
ec2-204-236-155-223.us-west-1.compute.amazonaws.com      - apache
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Compiling Cookbooks...
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Converging 4 resources
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Recipe: apache::server
ec2-50-18-19-208.us-west-1.compute.amazonaws.com      - build-essential
```

v3.0.0

 CHEF™
CODE CAN



v3.0.0

EXERCISE

Update the Proxy



- Update the wrapped proxy server cookbook to include the new web node as a member.
- Upload that cookbook to the Chef Server
- Login to that system and run chef-client
- Verify that the proxy server delivers traffic to both web server nodes.

```
~/.chef-repo/cookbooks/myhaproxy/recipes/default.rb
```

```
node.default['haproxy']['members'] = [{

    "hostname" => "ec2-204-236-155-223.us-west-1.compute.amazonaws.com",
    "ipaddress" => "ec2-204-236-155-223.us-west-1.compute.amazonaws.com",
    "port" => 80,
    "ssl_port" => 80

}, {

    "hostname" => "ec2-54-176-64-173.us-west-1.compute.amazonaws.com",
    "ipaddress" => "ec2-54-176-64-173.us-west-1.compute.amazonaws.com",
    "port" => 80,
    "ssl_port" => 80

}]

include_recipe "haproxy::default"
```



```
~/chef-repo/cookbooks/myhaproxy/metadata.rb
```

```
name          'myhaproxy'  
maintainer    'The Authors'  
maintainer_email 'you@example.com'  
license        'all_rights'  
description    'Installs/Configures myhaproxy'  
long_description 'Installs/Configures myhaproxy'  
version        '0.2.0'
```

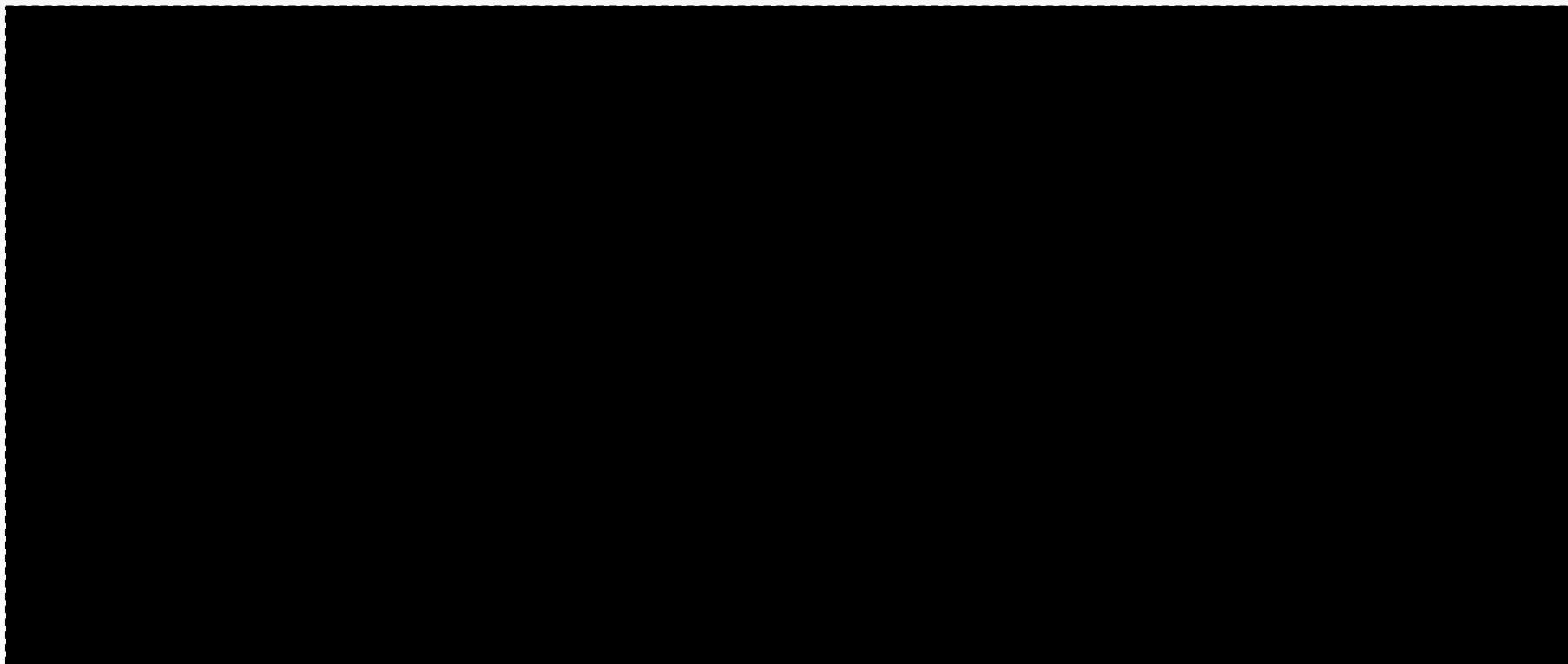
```
depends 'haproxy', '~> 1.6.6'
```

v3.0.0





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



v3.0.0



```
$ berks install
```

```
Resolving cookbook dependencies...
Fetching 'myhaproxy' from source at .
Fetching cookbook index from https://supermarket.chef.io...
Using build-essential (2.2.3)
Using cpu (0.2.0)
Using haproxy (1.6.6)
Using myhaproxy (0.2.0) from source at .
```

v3.0.0

 CHEF™
CODE CAN



```
$ berks upload
```

```
Skipping build-essential (2.2.3) (frozen)
```

```
Skipping cpu (0.2.0) (frozen)
```

```
Skipping haproxy (1.6.6) (frozen)
```

```
Uploaded myhaproxy (0.2.0) to: 'https://api.opscode.com:443/organizations/vogue'
```



```
$ knife ssh "*:*" -x USERNAME -P PWD "sudo chef-client"
```

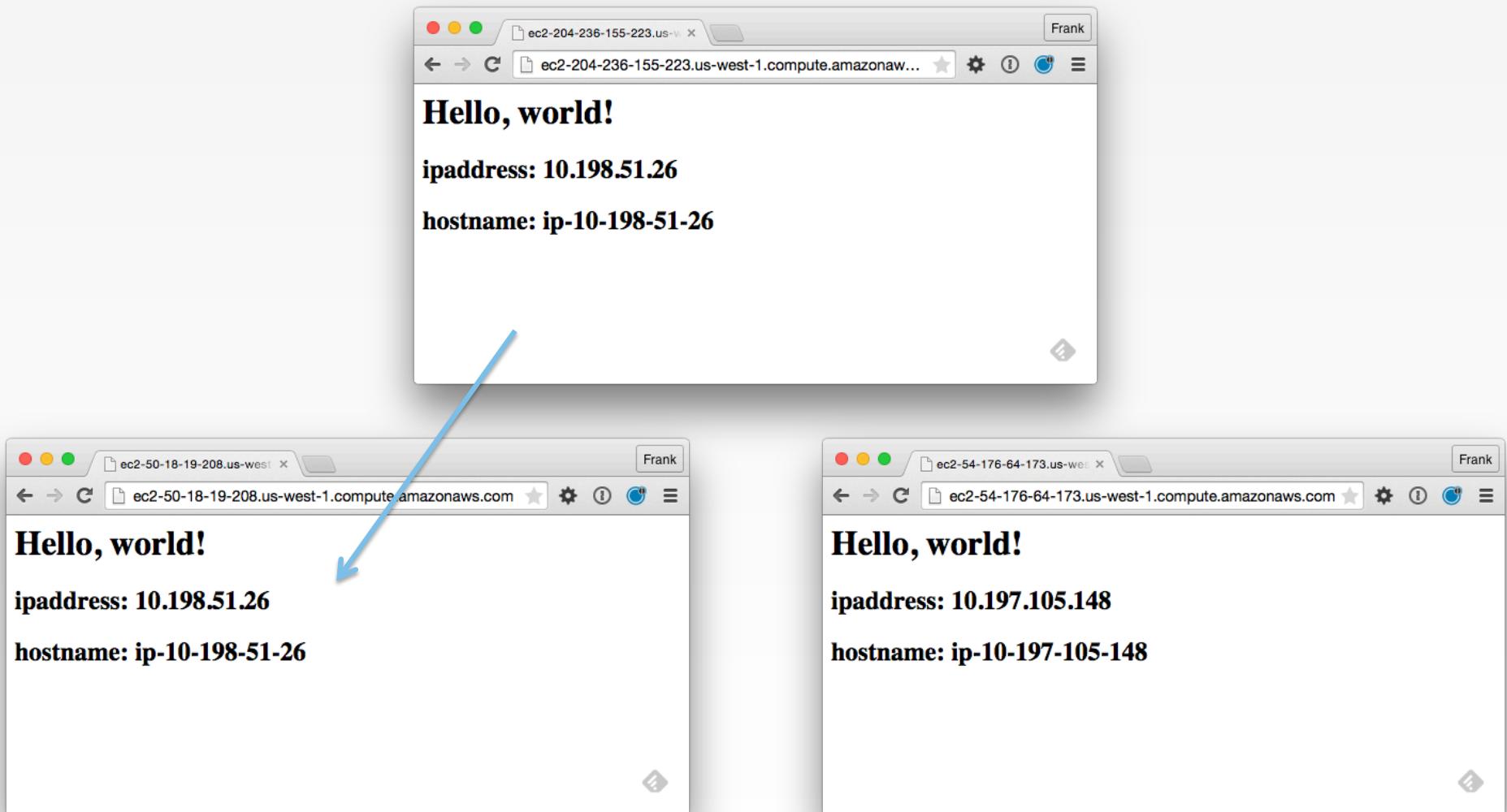
```
ec2-50-18-19-208.us-west-1.compute.amazonaws.com      Starting Chef Client, version 12.3.0
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Starting Chef Client, version 12.3.0
ec2-50-18-19-208.us-west-1.compute.amazonaws.com      resolving cookbooks for run list:
["myhaproxy"]

ec2-204-236-155-223.us-west-1.compute.amazonaws.com resolving cookbooks for run list:
["apache"]

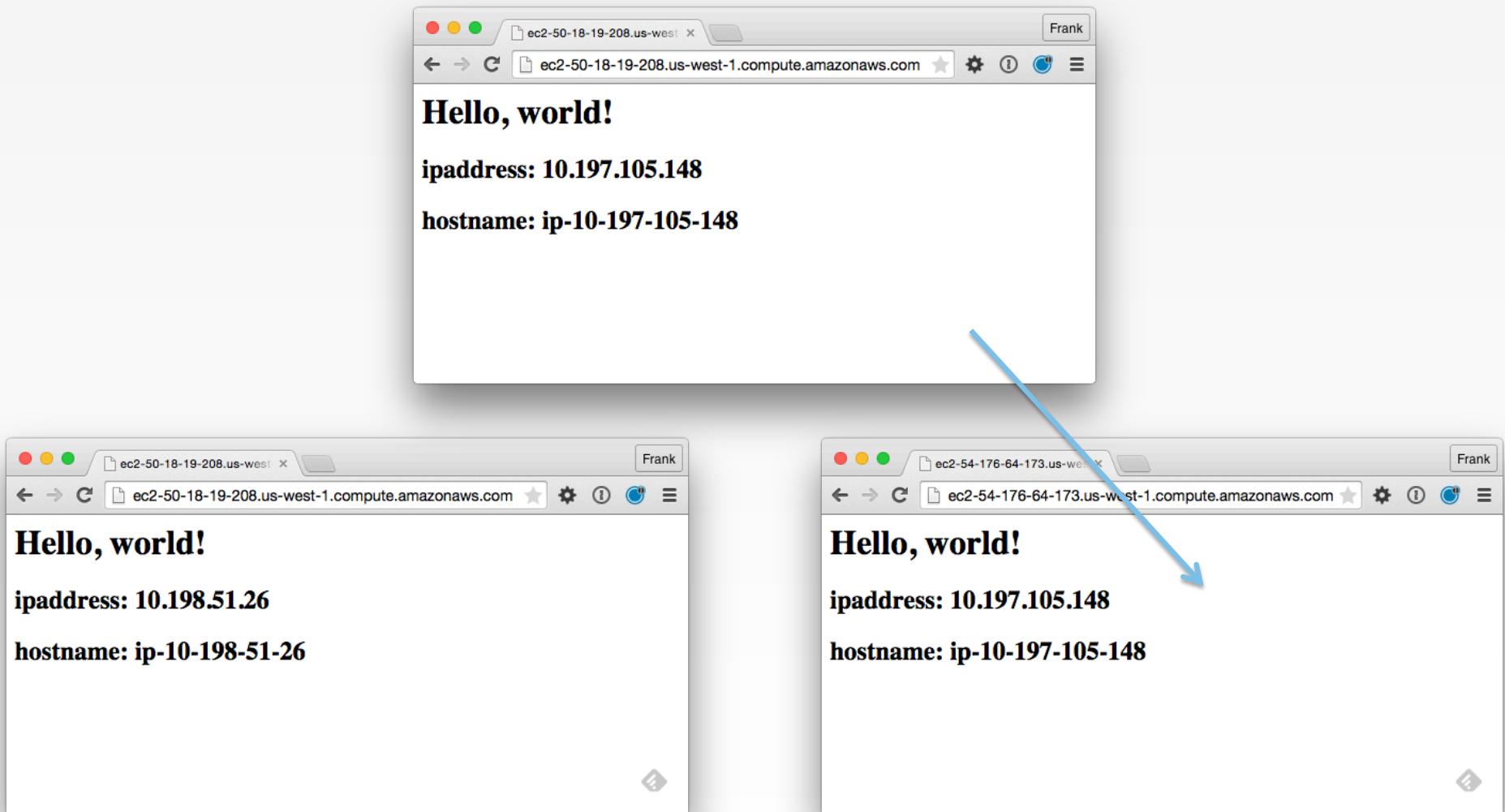
ec2-50-18-19-208.us-west-1.compute.amazonaws.com      Synchronizing Cookbooks:
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Synchronizing Cookbooks:
ec2-204-236-155-223.us-west-1.compute.amazonaws.com      - apache
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Compiling Cookbooks...
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Converging 4 resources
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Recipe: apache::server
ec2-50-18-19-208.us-west-1.compute.amazonaws.com      - build-essential
```

v3.0.0

 CHEF™
CODE CAN



v3.0.0



v3.0.0

DISCUSSION

Discussion



What questions can we help you answer?

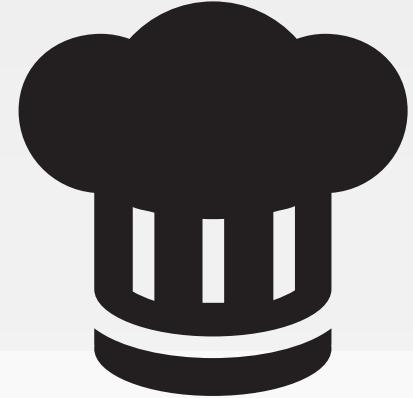


Roles

v3.0.0

LAB

Roles for Everyone



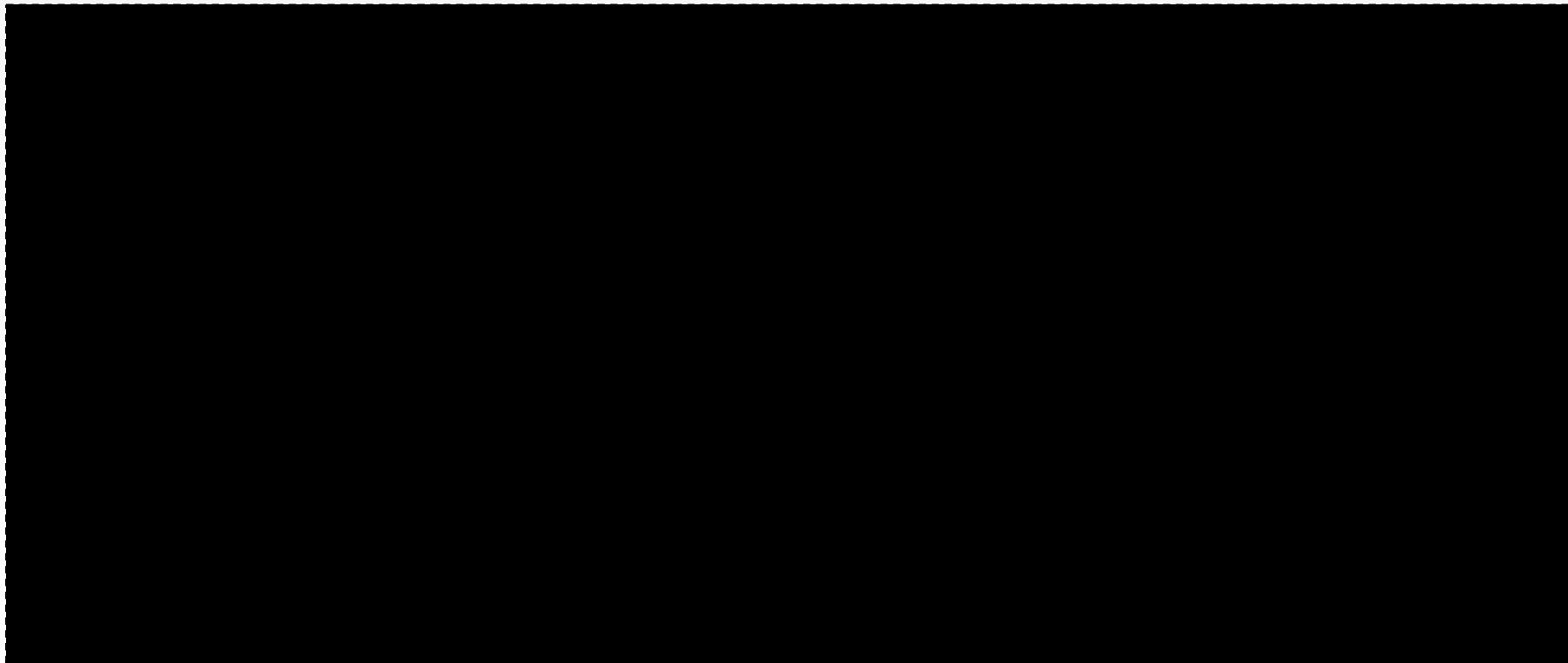
We will give our nodes a role to better describe them and so we can configure them in a similar manner.

OBJECTIVE:

- Give our proxy node a "proxy" Role
- Give our web nodes a "web" Role



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



v3.0.0

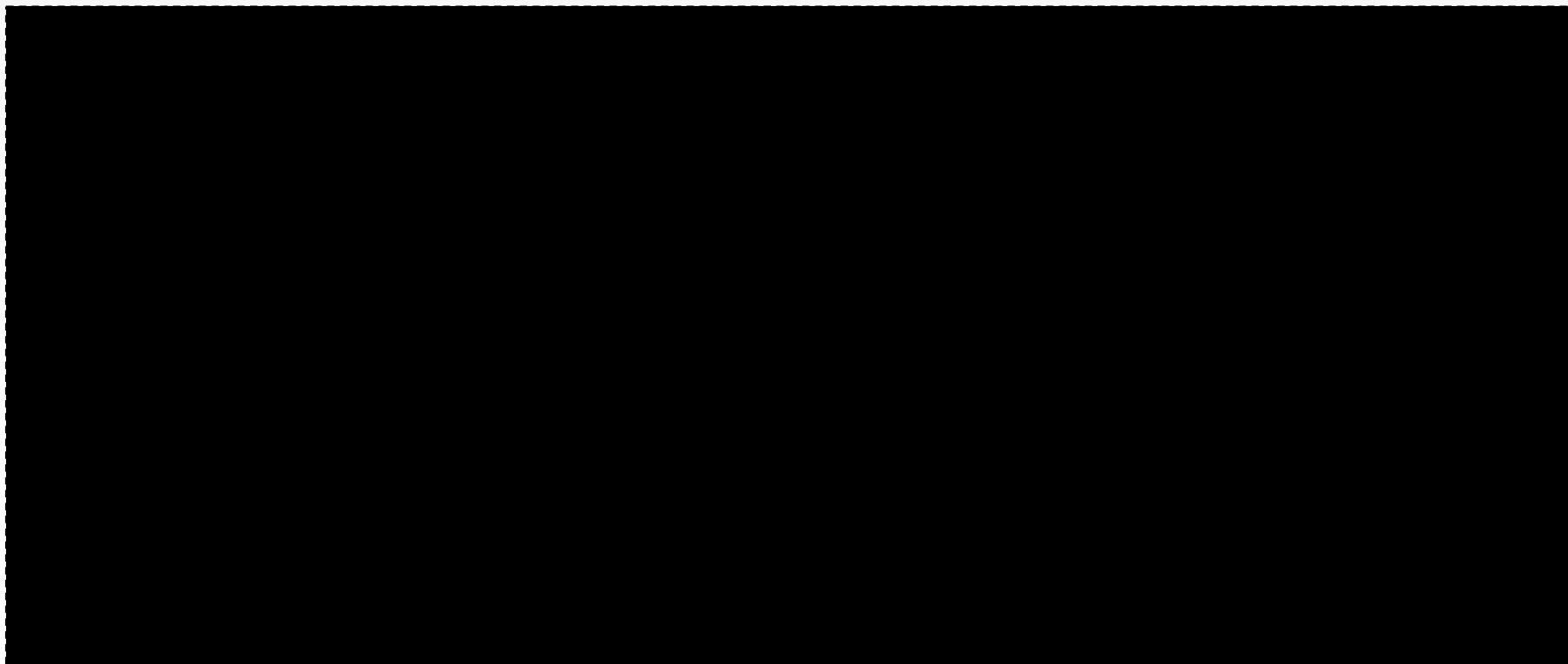


```
$ knife role --help
```

```
** ROLE COMMANDS **  
knife role bulk delete REGEX (options)  
knife role create ROLE (options)  
knife role delete ROLE (options)  
knife role edit ROLE (options)  
knife role env_run_list add [ROLE] [ENVIRONMENT] [ENTRY[,ENTRY]] (options)  
knife role env_run_list clear [ROLE] [ENVIRONMENT]  
knife role env_run_list remove [ROLE] [ENVIRONMENT] [ENTRIES]  
knife role env_run_list replace [ROLE] [ENVIRONMENT] [OLD_ENTRY] [NEW_ENTRY]  
knife role env_run_list set [ROLE] [ENVIRONMENT] [ENTRIES]  
knife role from file FILE [FILE..] (options)  
knife role list (options)  
knife role run_list add [ROLE] [ENTRY[,ENTRY]] (options)
```



```
$ knife role list
```

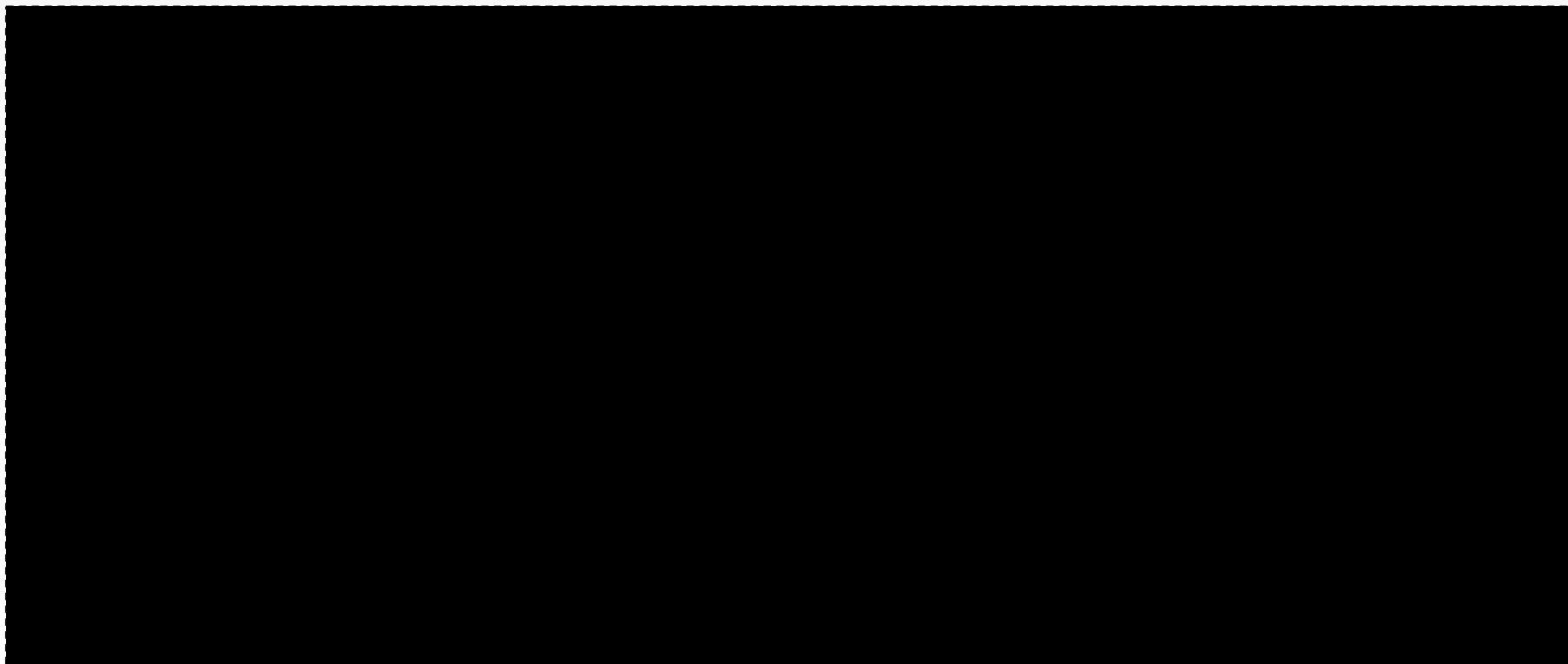


v3.0.0

 CHEF™
CODE CAN



```
$ mkdir roles
```



v3.0.0

 CHEF™
CODE CAN

```
[] ~chef-repo/roles/proxy.json
```

```
{  
  "name": "proxy",  
  "description": "Proxy server",  
  "run_list": [  
    "recipe[myhaproxy]"  
  ]  
}
```

v3.0.0





```
$ knife role from file proxy.json
```

```
Updated Role proxy!
```

v3.0.0



```
$ knife role list
```

```
proxy
```

v3.0.0

 CHEF™
CODE CAN



```
$ knife role show proxy
```

```
chef_type:          role
default_attributes:
description:       Proxy Server
env_run_lists:
json_class:        Chef::Role
name:              proxy
override_attributes:
run_list:          recipe[myhaproxy]
```



```
$ knife node --help
```

```
** NODE COMMANDS **  
knife node bulk delete REGEX (options)  
knife node create NODE (options)  
knife node delete NODE (options)  
knife node edit NODE (options)  
knife node environment set NODE ENVIRONMENT  
knife node from file FILE (options)  
knife node list (options)  
knife node run_list add [NODE] [ENTRY[,ENTRY]] (options)  
knife node run_list remove [NODE] [ENTRY[,ENTRY]] (options)  
knife node run_list set NODE ENTRIES (options)  
knife node show NODE (options)
```



```
$ knife node run_list set node2 "role[proxy]"
```

```
node2:  
  run_list: role[proxy]
```



```
$ knife node show node2
```

```
Node Name:    node2
Environment:  _default
FQDN:        ip-10-198-31-238.us-west-1.compute.internal
IP:          50.18.19.208
Run List:    role[proxy]
Roles:
Recipes:     myhaproxy, myhaproxy::default, haproxy::default, haproxy::install_package
Platform:   ubuntu 14.04
Tags:
```

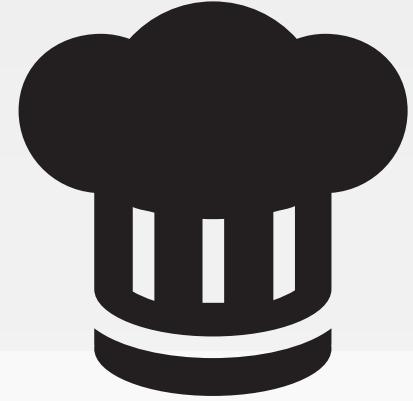


```
$ knife ssh "name:node2" -x USER -P PWD "sudo chef-client"
```

```
c2-50-18-19-208.us-west-1.compute.amazonaws.com Starting Chef Client, version 12.3.0
ec2-50-18-19-208.us-west-1.compute.amazonaws.com resolving cookbooks for run list:
["myhaproxy"]
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Synchronizing Cookbooks:
ec2-50-18-19-208.us-west-1.compute.amazonaws.com   - build-essential
ec2-50-18-19-208.us-west-1.compute.amazonaws.com   - cpu
ec2-50-18-19-208.us-west-1.compute.amazonaws.com   - haproxy
ec2-50-18-19-208.us-west-1.compute.amazonaws.com   - myhaproxy
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Compiling Cookbooks...
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Converging 9 resources
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Recipe: haproxy::install_package
ec2-50-18-19-208.us-west-1.compute.amazonaws.com   * apt_package[haproxy] action install (up
to date)
```

LAB

Roles for Everyone



We will give our nodes a role to better describe them and so we can configure them in a similar manner.

OBJECTIVE:

- Give our proxy node a "proxy" Role
- Give our web nodes a "web" Role

EXERCISE

Define a Web Role



- Create a role named 'web' that has the run list 'recipe[apache]'
- Set node1's run list to be "role[web]"
- Set node3's run list to be "role[web]"

```
[] ~chef-repo/roles/web.json
```

```
{  
  "name": "web",  
  "description": "Web server",  
  "run_list": [  
    "recipe[apache]"  
  ]  
}
```

v3.0.0





```
$ knife role from file web.json
```

```
Updated Role web!
```

v3.0.0



```
$ knife role list
```

```
proxy
```

```
web
```

v3.0.0



```
$ knife role show web
```

```
chef_type:          role
default_attributes:
description:        Web Server
env_run_lists:
json_class:         Chef::Role
name:               web
override_attributes:
run_list:           recipe[apache]
```



```
$ knife node run_list set node1 "role[web]"
```

```
node1:  
  run_list: role[web]
```



```
$ knife node run_list set node3 "role[web]"
```

```
node3:  
  run_list: role[web]
```

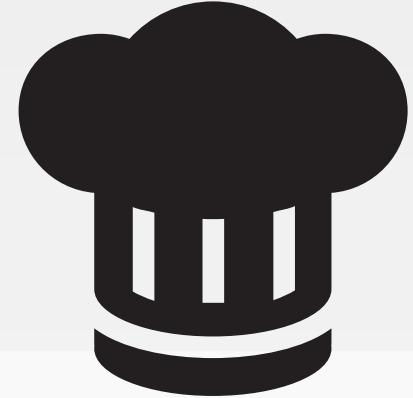


```
$ knife ssh "name:node1 OR name:node3" -x USER -P PWD \
"sudo chef-client"
```

```
ec2-54-176-64-173.us-west-1.compute.amazonaws.com    Starting Chef Client, version 12.3.0
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Starting Chef Client, version 12.3.0
ec2-54-176-64-173.us-west-1.compute.amazonaws.com    resolving cookbooks for run list:
["apache"]
ec2-54-176-64-173.us-west-1.compute.amazonaws.com    Synchronizing Cookbooks:
ec2-54-176-64-173.us-west-1.compute.amazonaws.com        - apache
ec2-54-176-64-173.us-west-1.compute.amazonaws.com    Compiling Cookbooks...
ec2-54-176-64-173.us-west-1.compute.amazonaws.com    Converging 4 resources
ec2-54-176-64-173.us-west-1.compute.amazonaws.com    Recipe: apache::server
ec2-204-236-155-223.us-west-1.compute.amazonaws.com resolving cookbooks for run list:
["apache"]
ec2-204-236-155-223.us-west-1.compute.amazonaws.com Synchronizing Cookbooks:
```

LAB

Roles for Everyone



We will give our nodes a role to better describe them and so we can configure them in a similar manner.

OBJECTIVE:

- ✓ Give our proxy node a "proxy" Role
- ✓ Give our web nodes a "web" Role

DISCUSSION

Discussion



What questions can we help you answer?



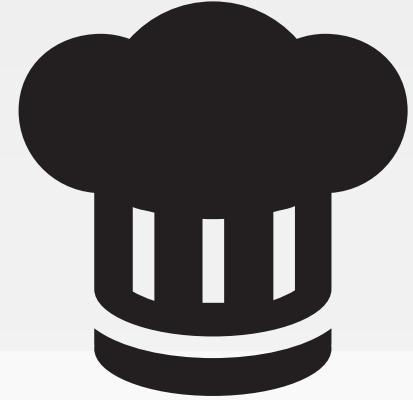
Search

v3.0.0



LAB

Dynamic Web Proxy



Every time we create a web node we need to update our proxy cookbook. That doesn't feel right!

OBJECTIVE:

- Update the wrapped proxy cookbook to dynamically use nodes with the web role



```
$ knife node show node1 -a cloud
```

```
node1:  
  cloud:  
    local_hostname: ip-10-198-51-26.us-west-1.compute.internal  
    local_ipv4: 10.198.51.26  
    private_ips: 10.198.51.26  
    provider: ec2  
    public_hostname: ec2-204-236-155-223.us-west-1.compute.amazonaws.com  
    public_ips: 204.236.155.223  
    public_ipv4: 204.236.155.223
```



```
$ knife node show node3 -a cloud
```

```
node3:  
  cloud:  
    local_hostname: ip-10-197-105-148.us-west-1.compute.internal  
    local_ipv4:      10.197.105.148  
    private_ips:     10.197.105.148  
    provider:       ec2  
    public_hostname: ec2-54-176-64-173.us-west-1.compute.amazonaws.com  
    public_ips:      54.176.64.173  
    public_ipv4:     54.176.64.173
```

~/.chef-repo/cookbooks/myhaproxy/default.rb

```
node.default['haproxy']['members'] = [{

    "hostname" => "ec2-204-236-155-223.us-west-1.compute.amazonaws.com",
    "ipaddress" => "ec2-204-236-155-223.us-west-1.compute.amazonaws.com",
    "port" => 80,
    "ssl_port" => 80
},
{
    "hostname" => "ec2-54-176-64-173.us-west-1.compute.amazonaws.com",
    "ipaddress" => "ec2-54-176-64-173.us-west-1.compute.amazonaws.com",
    "port" => 80,
    "ssl_port" => 80
}
]

include_recipe "haproxy::default"
```

~/.chef-repo/cookbooks/myhaproxy/default.rb

```
all_web_nodes = search("node", "role:web")

members = []

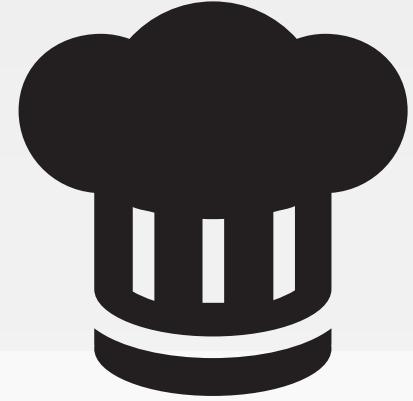
all_web_nodes.each do |web_node|
  member = {
    "hostname" => web_node["cloud"]["public_hostname"],
    "ipaddress" => web_node["cloud"]["public_ipv4"],
    "port" => 80,
    "ssl_port" => 80
  }
  members.push(member)
end

node.default['haproxy']['members'] = members

include_recipe "haproxy::default"
```

LAB

Dynamic Web Proxy



Every time we create a web node we need to update our proxy cookbook. That doesn't feel right!

OBJECTIVE:

- ✓ Update the wrapped proxy cookbook to dynamically use nodes with the web role

EXERCISE

Upload the Cookbook



- Update the minor versions of the proxy cookbook
- Upload the proxy cookbook
- Run chef-client on the proxy node
- Verify that the proxy node relays requests to both web nodes



```
~/chef-repo/cookbooks/myhaproxy/metadata.rb
```

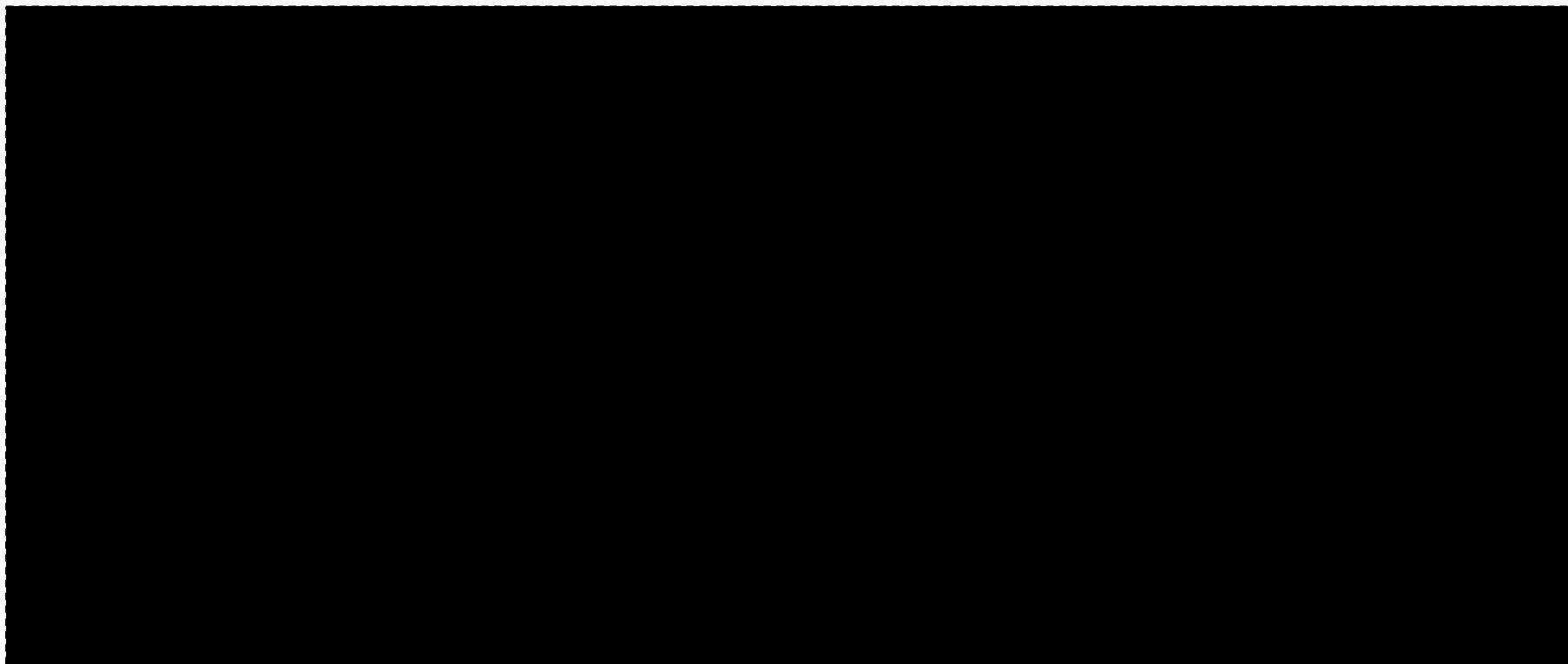
```
name          'myhaproxy'  
maintainer    'The Authors'  
maintainer_email 'you@example.com'  
license        'all_rights'  
description    'Installs/Configures myhaproxy'  
long_description 'Installs/Configures myhaproxy'  
version        '0.3.0'  
  
depends 'haproxy', '~> 1.6.6'
```

v3.0.0





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



v3.0.0



```
$ berks install
```

```
Resolving cookbook dependencies...
Fetching 'myhaproxy' from source at .
Fetching cookbook index from https://supermarket.chef.io...
Using build-essential (2.2.3)
Using haproxy (1.6.6)
Using myhaproxy (0.1.0) from source at .
Using cpu (0.2.0)
```



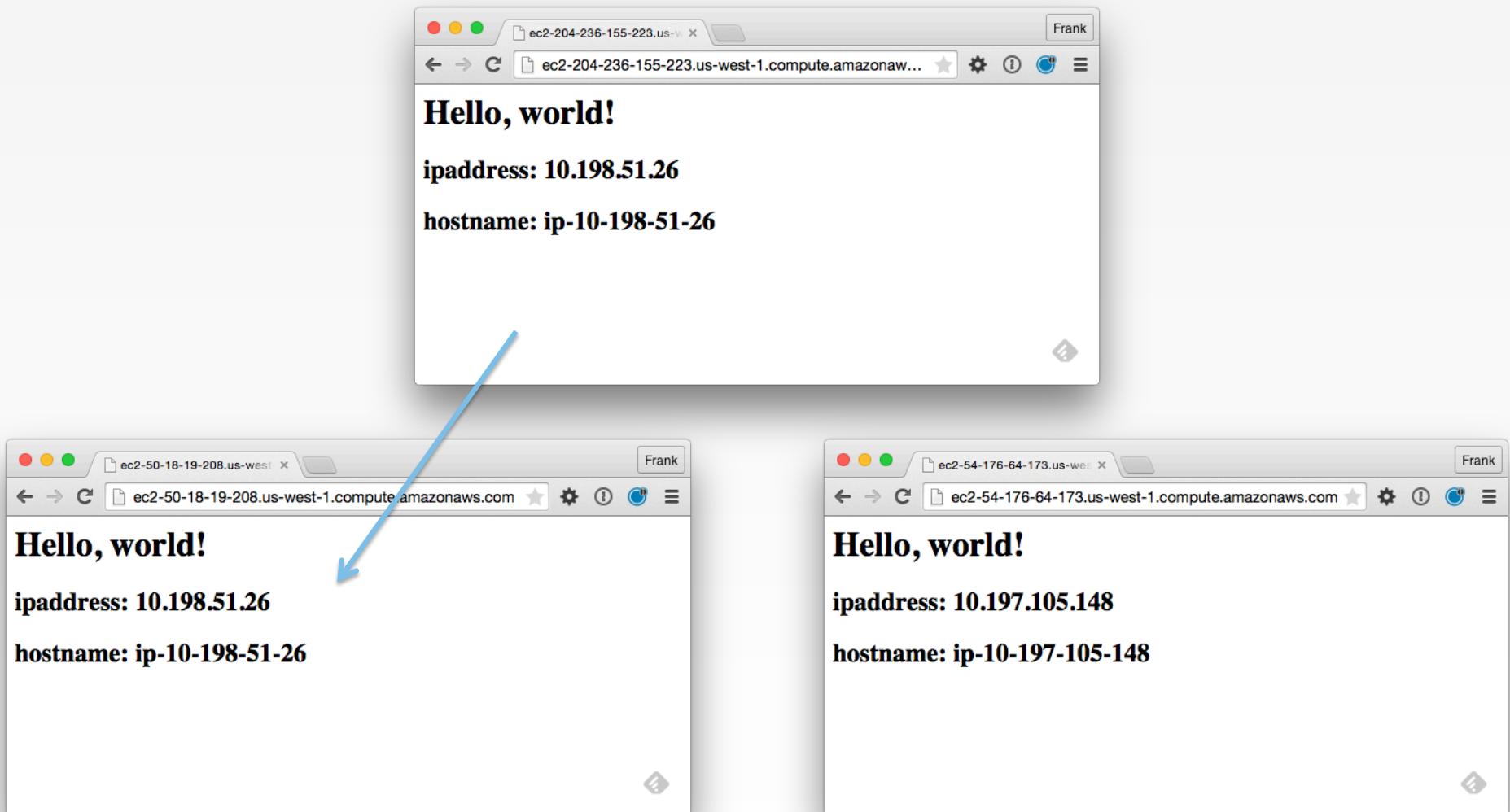
```
$ berks upload
```

```
Uploaded build-essential (2.2.3) to: 'https://api.opscode.com:443/organizations/vogue'  
Uploaded cpu (0.2.0) to: 'https://api.opscode.com:443/organizations/vogue'  
Uploaded haproxy (1.6.6) to: 'https://api.opscode.com:443/organizations/vogue'  
Uploaded myhaproxy (0.3.0) to: 'https://api.opscode.com:443/organizations/vogue'
```

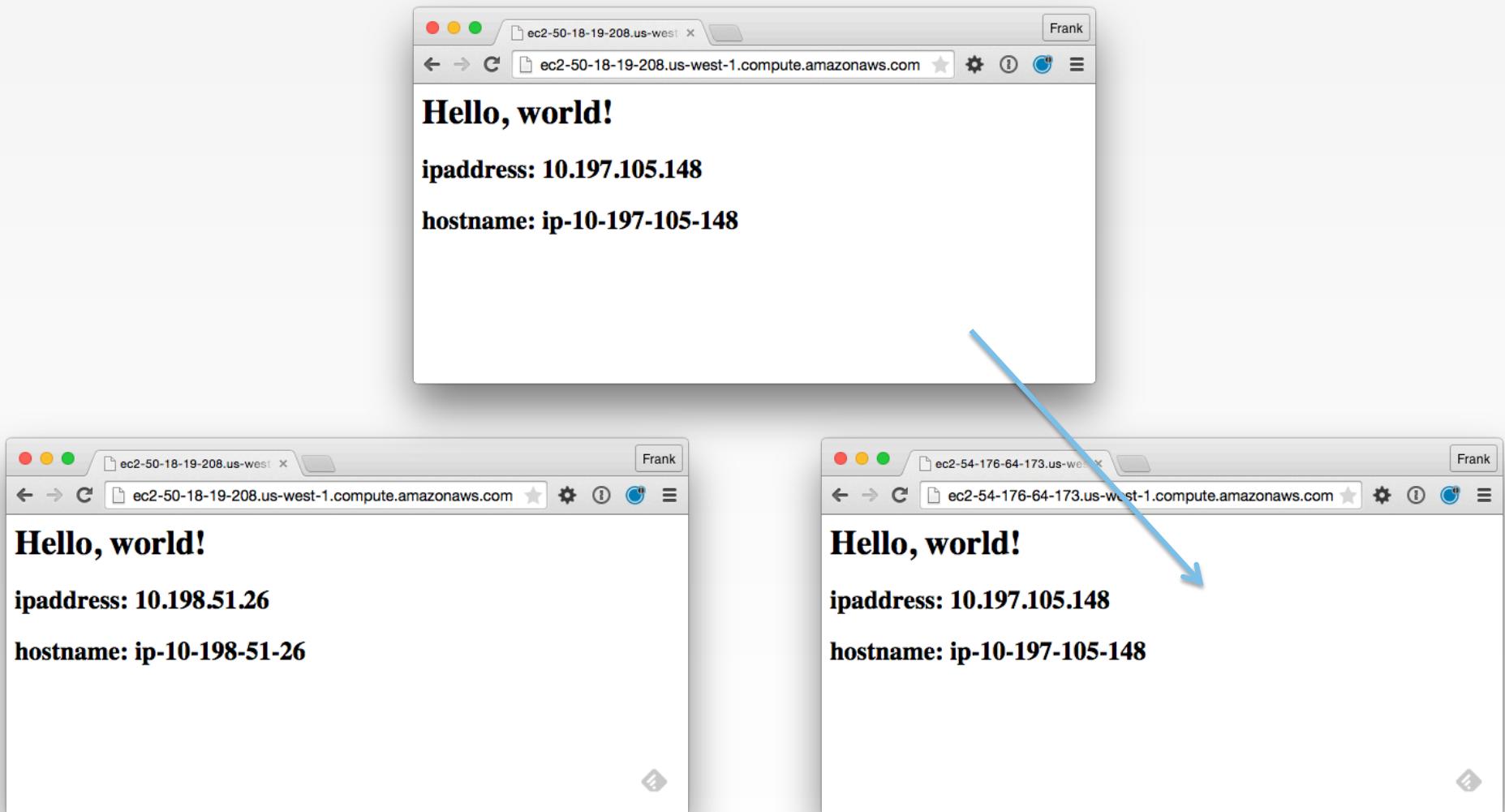


```
$ knife ssh "name:node2" -x USER -P PWD "sudo chef-client"
```

```
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Starting Chef Client, version 12.3.0
ec2-50-18-19-208.us-west-1.compute.amazonaws.com resolving cookbooks for run list:
["myhaproxy"]
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Synchronizing Cookbooks:
ec2-50-18-19-208.us-west-1.compute.amazonaws.com   - cpu
ec2-50-18-19-208.us-west-1.compute.amazonaws.com   - build-essential
ec2-50-18-19-208.us-west-1.compute.amazonaws.com   - haproxy
ec2-50-18-19-208.us-west-1.compute.amazonaws.com   - myhaproxy
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Compiling Cookbooks...
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Converging 9 resources
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Recipe: haproxy::install_package
ec2-50-18-19-208.us-west-1.compute.amazonaws.com   * apt_package[haproxy] action install (up
to date)
```



v3.0.0



v3.0.0

DISCUSSION

Discussion



What questions can we help you answer?



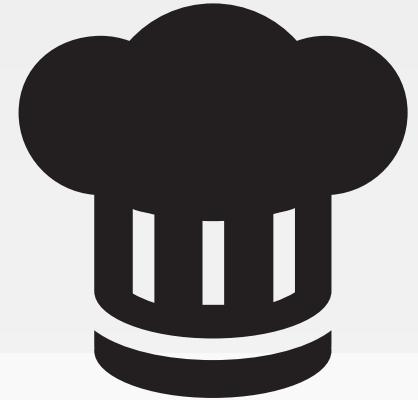
Environments

v3.0.0



LAB

Production



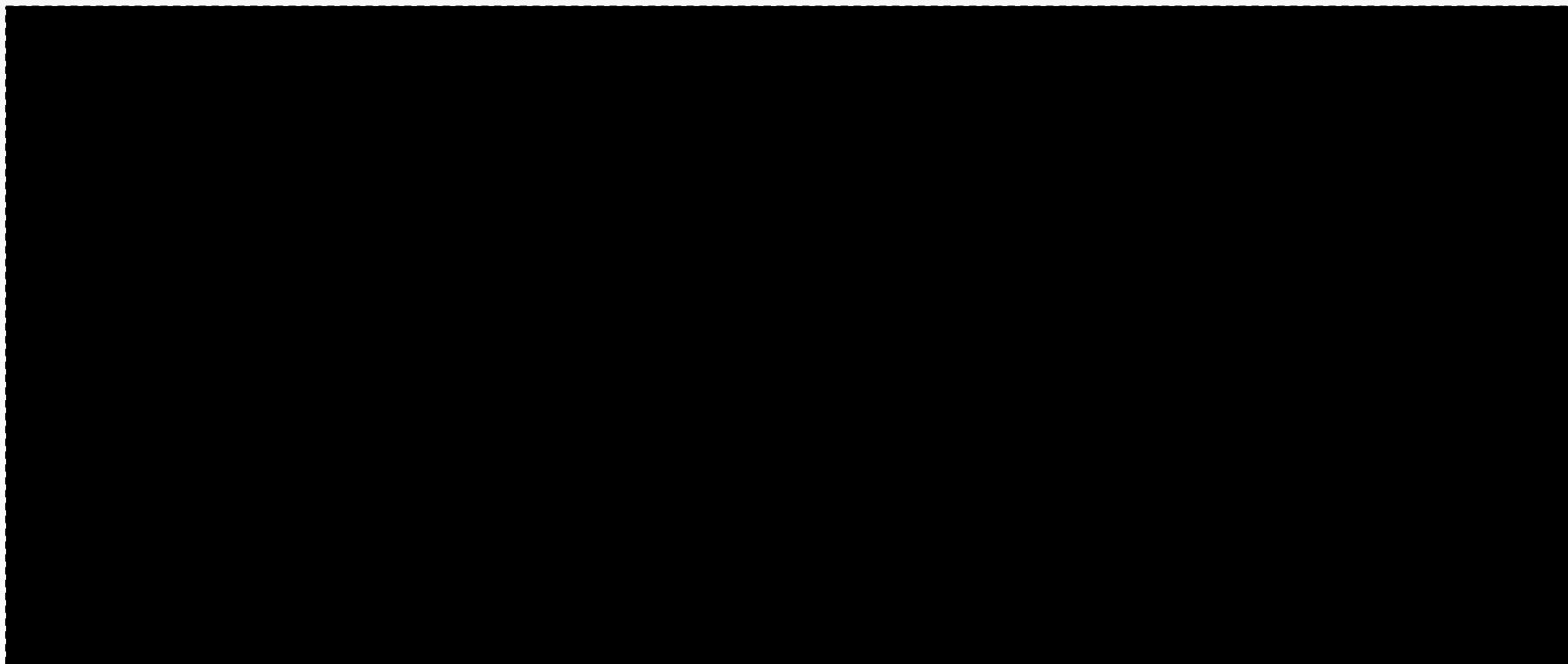
Let's create a reliable environment for our nodes.

OBJECTIVE:

- Deploy our site to Production



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



v3.0.0

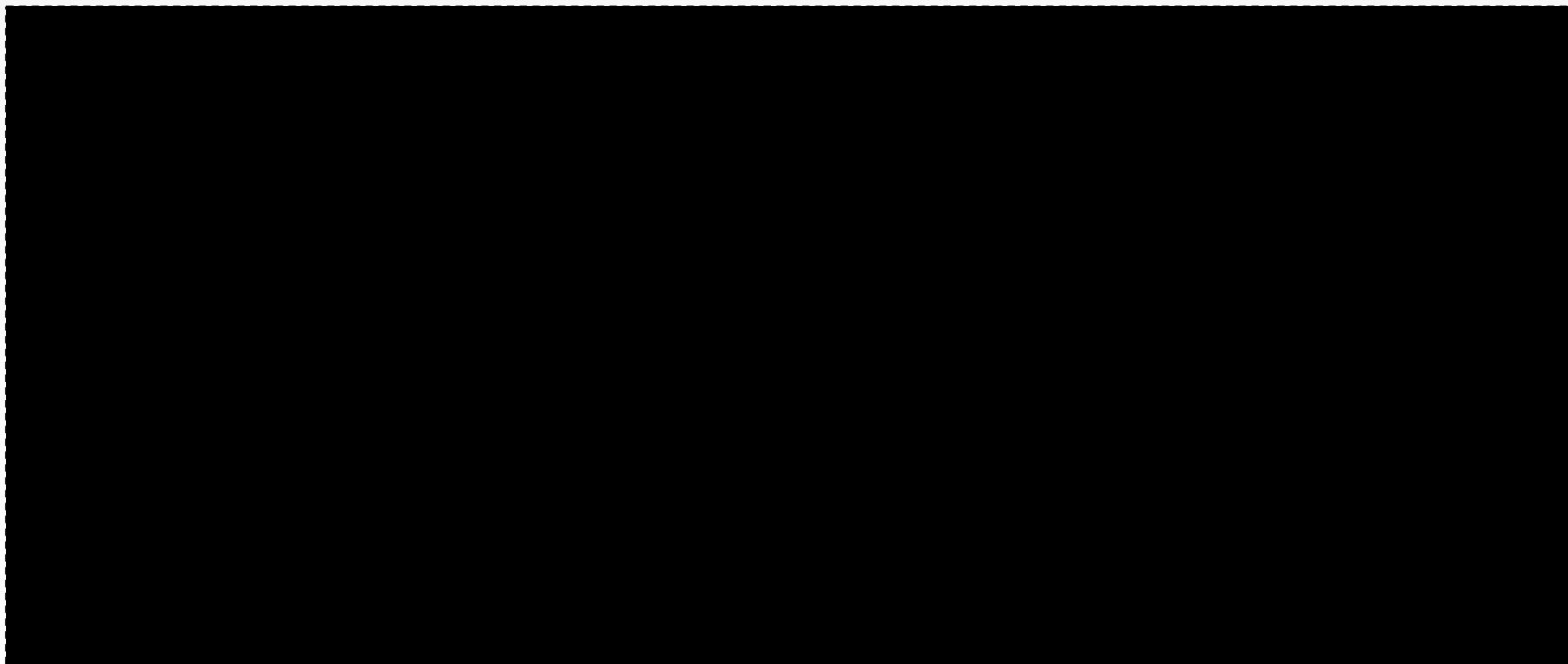


```
$ knife environment --help
```

```
** ENVIRONMENT COMMANDS **  
knife environment compare [ENVIRONMENT..] (options)  
knife environment create ENVIRONMENT (options)  
knife environment delete ENVIRONMENT (options)  
knife environment edit ENVIRONMENT (options)  
knife environment from file FILE [FILE..] (options)  
knife environment list (options)  
knife environment show ENVIRONMENT (options)
```



```
$ mkdir environments
```



v3.0.0

```
~/.chef-repo/environments/production.json
```

```
{  
  "name": "production",  
  "description": "Where we run production code",  
  "cookbook_versions": {  
    "apache" : "= 0.2.1",  
    "myhaproxy" : "= 0.3.0"  
  }  
}
```

v3.0.0





```
$ knife environment list
```

```
_default
```

v3.0.0



```
$ knife environment from file production.json
```

```
Updated Environment production
```

v3.0.0

 CHEF™
CODE CAN



```
$ knife environment list
```

```
_default  
production
```



```
$ knife environment show production
```

```
chef_type:           environment
cookbook_versions:
  apache:      = 0.2.1
  myhaproxy:   = 0.3.0
default_attributes:
description:        Where we run production code
json_class:         Chef::Environment
name:               production
override_attributes:
```



```
$ knife node --help
```

```
** NODE COMMANDS **  
knife node bulk delete REGEX (options)  
knife node create NODE (options)  
knife node delete NODE (options)  
knife node edit NODE (options)  
knife node environment set NODE ENVIRONMENT  
knife node from file FILE (options)  
knife node list (options)  
knife node run_list add [NODE] [ENTRY[,ENTRY]] (options)  
knife node run_list remove [NODE] [ENTRY[,ENTRY]] (options)  
knife node run_list set NODE ENTRIES (options)  
knife node show NODE (options)
```



```
$ knife node environment set --help
```

```
knife node environment set NODE_ENVIRONMENT
  -s, --server-url URL          Chef Server URL
  --chef-zero-host HOST         Host to start chef-zero on
  --chef-zero-port PORT        Port (or port range) to start chef-zero on.  Port ranges
like 1000,1010 or 8889-9999 will try all given ports until one works.
  -k, --key KEY                API Client Key
  --[no-]color                 Use colored output, defaults to false on Windows, true
otherwise
  -c, --config CONFIG          The configuration file to use
  --defaults                   Accept default values for all questions
  -d, --disable-editing        Do not open EDITOR, just accept the data as is
  -e, --editor EDITOR          Set the editor to use for interactive commands
  -E, --environment ENVIRONMENT
this will be flagrantly ignored)  Set the Chef environment (except for in searches, where
```



```
$ knife node environment set node1 production
```

```
node1:  
  chef_environment:
```



```
$ knife node show node1
```

```
Node Name:    node1
Environment:  production
FQDN:        ip-10-198-51-26.us-west-1.compute.internal
IP:          204.236.155.223
Run List:    role[web]
Roles:       web
Recipes:     apache, apache::default, apache::server
Platform:   ubuntu 14.04
Tags:
```

EXERCISE

More Nodes to Production



- Set node2's environment to production



```
$ knife node environment set node2 production
```

```
node2:  
  chef_environment:
```

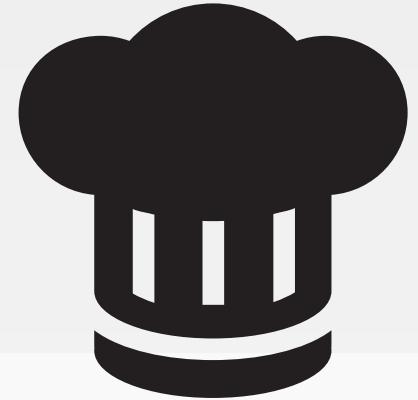


```
$ knife node show node2
```

```
Node Name:    node2
Environment:  production
FQDN:        ip-10-198-31-238.us-west-1.compute.internal
IP:          50.18.19.208
Run List:    role[proxy]
Roles:       proxy
Recipes:     myhaproxy, myhaproxy::default, haproxy::default, haproxy::install_package
Platform:    ubuntu 14.04
Tags:
```

LAB

Production



Let's create a reliable environment for our nodes.

OBJECTIVE:

- ✓ Deploy our site to Production

EXERCISE

Union Environment



- Create an environment named "union" that has no cookbook restrictions.
- Move node3 into the union environment
- Run chef-client on all the nodes
- Verify the proxy only sends requests to the production web node

```
[] ~chef-repo/environments/union.json
```

```
{  
  "name": "union",  
  "description": "Where code and applications are tested"  
  // no cookbook version constraints  
}
```

v3.0.0





```
$ knife environment from file union.json
```

```
Updated Environment union
```

v3.0.0



```
$ knife environment list
```

```
_default  
production  
union
```



```
$ knife environment show union
```

```
chef_type:           environment
cookbook_versions:
default_attributes:
description:        Where code and applications are tested
json_class:         Chef::Environment
name:               union
override_attributes:
```



```
$ knife node environment set node3 union
```

```
node3:  
  chef_environment:
```



```
$ knife node show node3
```

```
Node Name:    node3
Environment:  union
FQDN:        ip-10-197-105-148.us-west-1.compute.internal
IP:          54.176.64.173
Run List:    role[web]
Roles:
Recipes:     apache, apache::default, apache::server
Platform:   ubuntu 14.04
Tags:
```



```
$ knife ssh "*:*" -x USER -P PWD "sudo chef-client"
```

```
ec2-50-18-19-208.us-west-1.compute.amazonaws.com      Starting Chef Client, version 12.3.0
ec2-204-236-155-223.us-west-1.compute.amazonaws.com  Starting Chef Client, version 12.3.0
ec2-54-176-64-173.us-west-1.compute.amazonaws.com    Starting Chef Client, version 12.3.0
ec2-50-18-19-208.us-west-1.compute.amazonaws.com    resolving cookbooks for run list:
["myhaproxy"]

ec2-54-176-64-173.us-west-1.compute.amazonaws.com  resolving cookbooks for run list:
["apache"]

ec2-50-18-19-208.us-west-1.compute.amazonaws.com    Synchronizing Cookbooks:
ec2-204-236-155-223.us-west-1.compute.amazonaws.com  resolving cookbooks for run list:
["apache"]

ec2-50-18-19-208.us-west-1.compute.amazonaws.com    - myhaproxy
ec2-50-18-19-208.us-west-1.compute.amazonaws.com    - cpu
ec2-50-18-19-208.us-west-1.compute.amazonaws.com    - build-essential
ec2-50-18-19-208.us-west-1.compute.amazonaws.com    - haproxy
```

Web Node in Union



The node in the union environment is still receiving requests from the load balancer.

DISCUSSION

Expected Situation



What did we expect to happen when we moved a web node to a different environment?

DISCUSSION

Actual Situation



What is the actual situation?

DISCUSSION

Balancing Nodes



What cookbook handles balancing the requests between web nodes?

DISCUSSION

Balancing Nodes



What recipe within that cookbook sets up the request balancing between the two nodes?

DISCUSSION

Search Criteria



How are we currently searching for web nodes?

DISCUSSION

Search Criteria



How can we further refine our search results?

EXERCISE

Update the Search



- ❑ Update the search criteria to return:

node's role is webserver AND node's environment is the same as the proxy server's environment

- ❑ Update the Patch Version
- ❑ Upload the Cookbook
- ❑ Run chef-client on the proxy node

https://docs.chef.io/chef_search.html#and

https://docs.chef.io/chef_search.html#environments

<https://docs.chef.io/environments.html#find-environment-from-recipe>

```
~/.chef-repo/cookbooks/myhaproxy/recipes/default.rb
```

```
#  
# Cookbook Name:: myhaproxy  
# Recipe:: default  
#  
# Copyright (c) 2015 The Authors, All Rights Reserved.  
  
all_web_nodes = search("node","role:web AND chef_environment:#{node.chef_environment}")  
  
members = []  
  
#...
```

v3.0.0



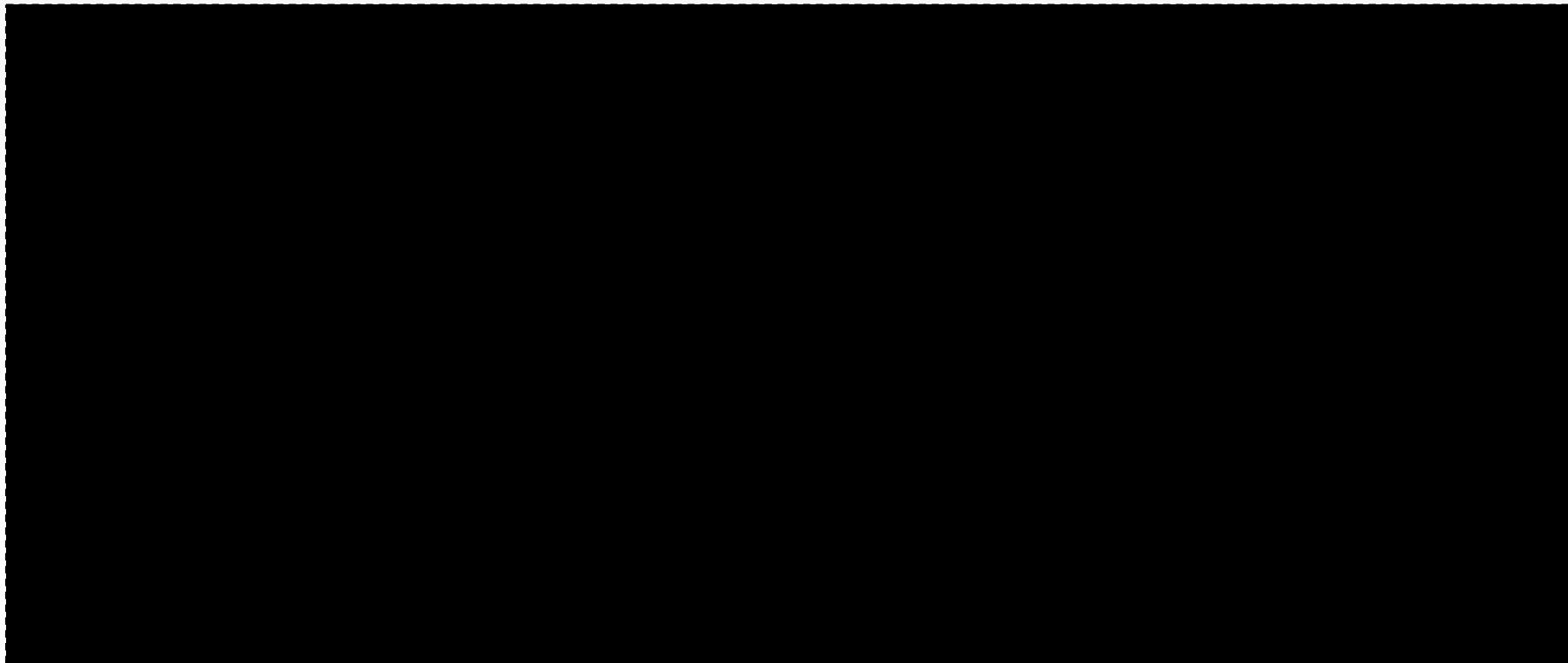


```
~/chef-repo/cookbooks/myhaproxy/metadata.rb
```

```
name          'myhaproxy'  
maintainer    'The Authors'  
maintainer_email 'you@example.com'  
license        'all_rights'  
description    'Installs/Configures myhaproxy'  
long_description 'Installs/Configures myhaproxy'  
version        '0.3.1'  
  
depends 'haproxy', '= 1.6.6'
```



```
$ cd cookbooks/myhaproxy
```



v3.0.0

 CHEF™
CODE CAN



```
$ berks install
```

```
Resolving cookbook dependencies...
Fetching 'myhaproxy' from source at .
Fetching cookbook index from https://supermarket.chef.io...
Using build-essential (2.2.3)
Installing haproxy (1.6.6)
Using cpu (0.2.0)
Using myhaproxy (0.3.1) from source at .
```



```
$ berks upload
```

```
Skipping build-essential (2.2.3) (frozen)
```

```
Skipping cpu (0.2.0) (frozen)
```

```
Skipping haproxy (1.6.6) (frozen)
```

```
Uploaded myhaproxy (0.3.1) to: 'https://api.opscode.com:443/organizations/vogue'
```



```
$ knife ssh "name:node2" -x USER -P PWD "sudo chef-client"
```

```
Skipping build-essential (2.2.3) (frozen)
```

```
Skipping cpu (0.2.0) (frozen)
```

```
Skipping haproxy (1.6.6) (frozen)
```

```
Uploaded myhaproxy (0.3.1) to: 'https://api.opscode.com:443/organizations/vogue'
```

Updated Cookbook



The proxy is still delivering requests to both nodes!

DISCUSSION

Updated Cookbook



What changes did we make to the cookbook?

v3.0.0

 CHEF™
CODE CAN

DISCUSSION

Updated Cookbook



How can we check to see if the cookbook has been successfully uploaded?

DISCUSSION

Source and Version



How can we check our search query to ensure that it returns the correct set of nodes?

DISCUSSION

Source and Version



How could the change to the version affect our latest cookbook?



```
~/chef-repo/cookbooks/myhaproxy/metadata.rb
```

```
name          'myhaproxy'  
maintainer    'The Authors'  
maintainer_email 'you@example.com'  
license        'all_rights'  
description    'Installs/Configures myhaproxy'  
long_description 'Installs/Configures myhaproxy'  
version        '0.3.1'  
  
depends 'haproxy', '= 1.6.6'
```

v3.0.0



```
[] ~chef-repo/environments/production.rb
```

```
name "production"
description "Where we run production code"

cookbook "apache", "= 0.2.1"
cookbook "myhaproxy", "= 0.3.0"
```

v3.0.0



EXERCISE

Update Production



- Update the environment named production:

'myhaproxy' cookbook version equal to '0.3.1'

```
[] ~chef-repo/environments/production.rb
```

```
name "production"
description "Where we run production code"

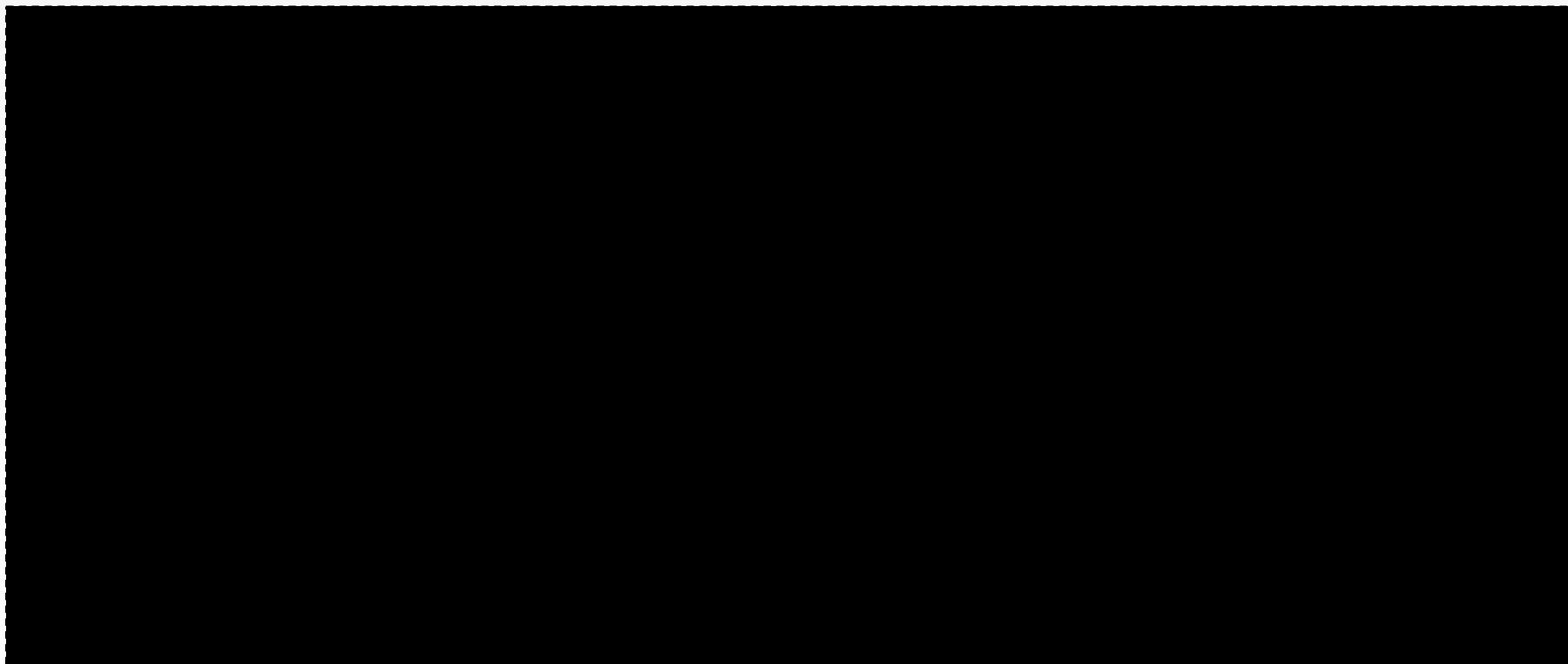
cookbook "apache", "= 0.2.1"
cookbook "myhaproxy", "= 0.3.1"
```

v3.0.0





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



v3.0.0



```
$ knife environment from file production.rb
```

```
Updated Environment production
```

v3.0.0



```
$ knife environment show production
```

```
chef_type:           environment
cookbook_versions:
  apache:      = 0.2.1
  myhaproxy:   = 0.3.1
default_attributes:
description:        Where we run production code
json_class:         Chef::Environment
name:               production
override_attributes:
```



```
$ knife ssh "name:node2" -x USER -P PWD "sudo chef-client"
```

```
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Starting Chef Client, version 12.3.0
ec2-50-18-19-208.us-west-1.compute.amazonaws.com resolving cookbooks for run list:
["myhaproxy"]
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Synchronizing Cookbooks:
ec2-50-18-19-208.us-west-1.compute.amazonaws.com   - cpu
ec2-50-18-19-208.us-west-1.compute.amazonaws.com   - build-essential
ec2-50-18-19-208.us-west-1.compute.amazonaws.com   - haproxy
ec2-50-18-19-208.us-west-1.compute.amazonaws.com   - myhaproxy
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Compiling Cookbooks...
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Converging 9 resources
ec2-50-18-19-208.us-west-1.compute.amazonaws.com Recipe: haproxy::install_package
ec2-50-18-19-208.us-west-1.compute.amazonaws.com   * apt_package[haproxy] action install (up
to date)
ec2-50-18-19-208.us-west-1.compute.amazonaws.com
ec2-50-18-19-208.us-west-1.compute.amazonaws.com
ec2-50-18-19-208.us-west-1.compute.amazonaws.com
```

One more thing...



v3.0.0

 **CHEF**™
CODE CAN

EXERCISE

New Home Page



- Update the apache cookbook's template to new page found at URL
- Bump version of cookbook
- Upload cookbook
- Update all nodes
- Verify only the web node in the union updated

EXERCISE

Ready for Production



- Update the production environment cookbook restriction to latest apache version
- Update all nodes
- Verify that all production nodes have updated

DISCUSSION

Discussion



What questions can we help you answer?



Further Resources

v3.0.0



LAB

What do I do now?



To learn more you are going to need more practice and more resources. It takes a village and a couple of web servers.

OBJECTIVE:

- Provide you with resources that you can read
- Provide you with resources that you can watch
- Provide you with resources that you can listen
- Provide you with resources that you can attend

DOCS

<https://goo.gl/cExAi1>



The slides from this workshop.

v3.0.0

 CHEF™
CODE CAN

DOCS
docs.chef.io

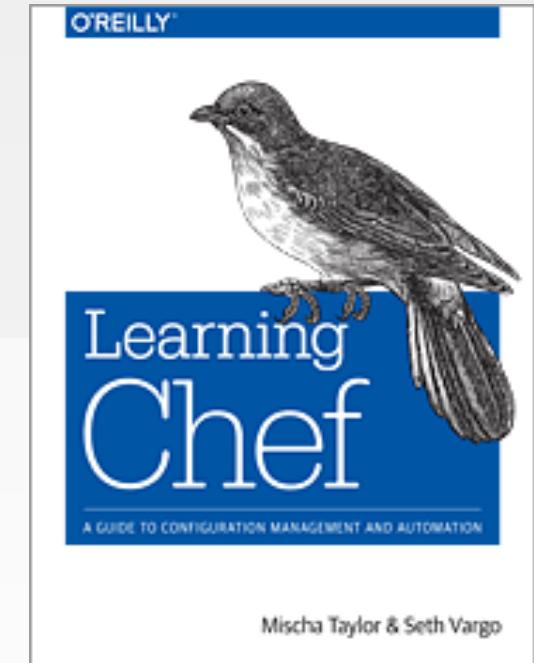


v3.0.0



Learning Chef

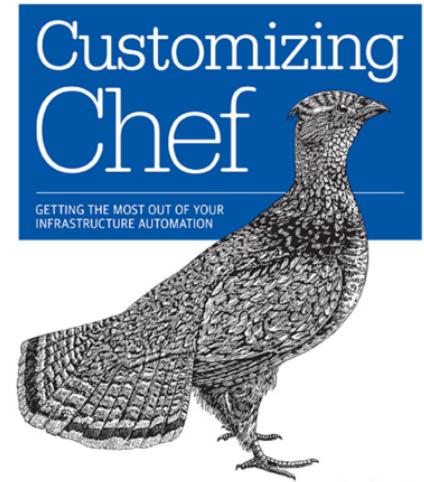
A Guide to Configuration Management
and Automation



Customizing Chef

Getting the Most Out of Your
Infrastructure Automation

O'REILLY®





Activity focused learning for those new to Chef.

v3.0.0



LAB

What do I do now?



To learn more you are going to need more practice and more resources. It takes a village and a couple of web servers.

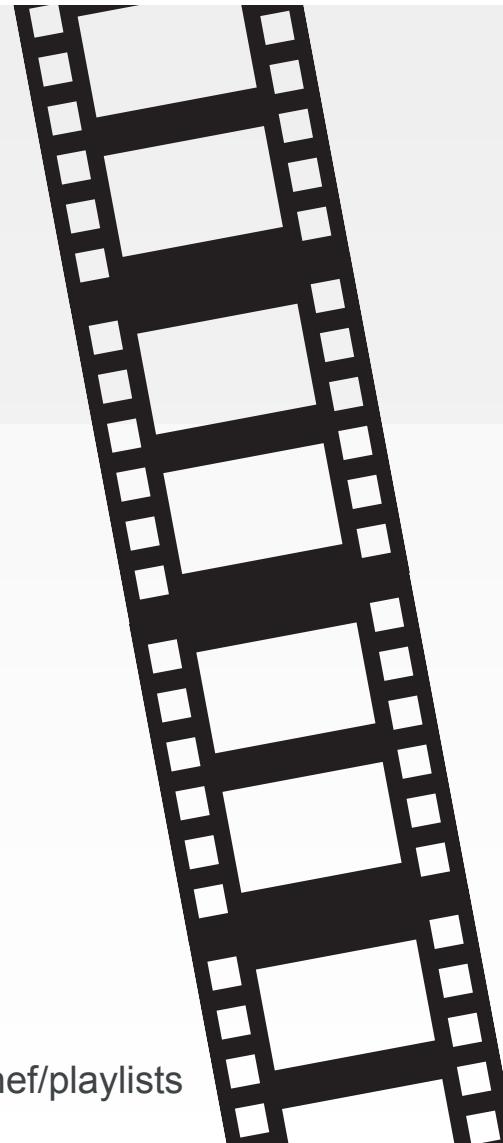
OBJECTIVE:

- Provide you with resources that you can read
- Provide you with resources that you can watch
- Provide you with resources that you can listen
- Provide you with resources that you can attend

Youtube Channel

- ChefConf Talks
- Training Videos

<https://www.youtube.com/user/getchef/playlists>



foodfightshow.org

Food Fight is a bi-weekly podcast for the Chef community. We bring together the smartest people in the Chef community and the broader DevOps world to discuss the thorniest issues in system administration.



LAB

What do I do now?



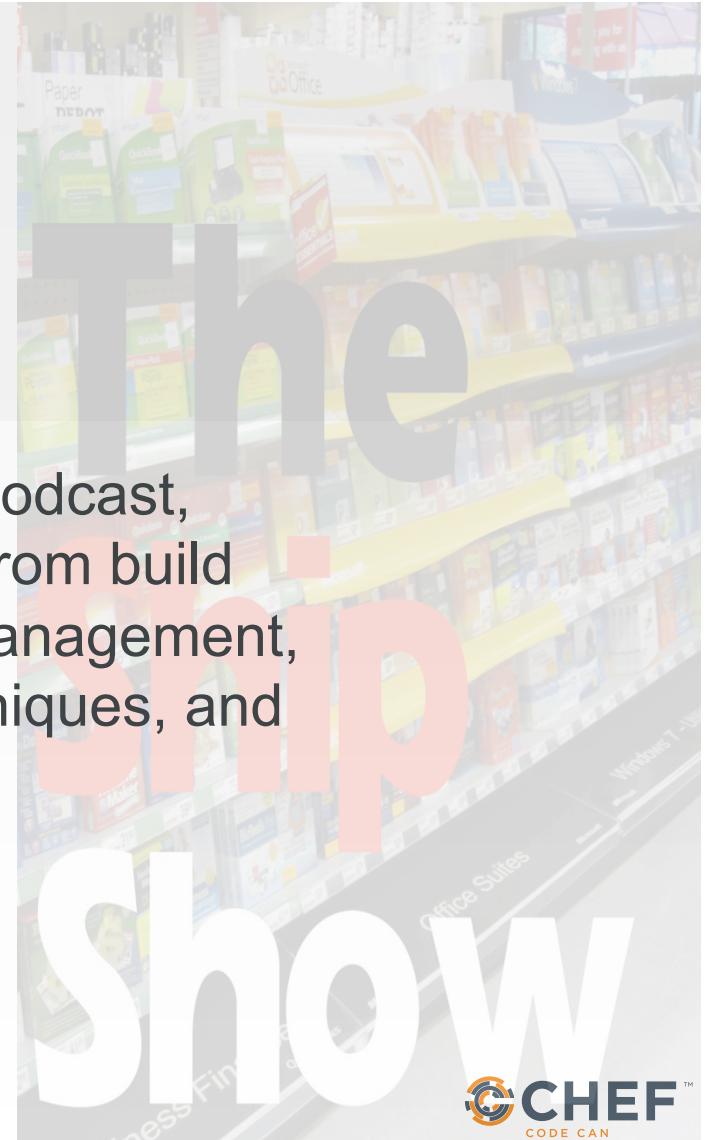
To learn more you are going to need more practice and more resources. It takes a village and a couple of web servers.

OBJECTIVE:

- Provide you with resources that you can read
- Provide you with resources that you can watch
- Provide you with resources that you can listen
- Provide you with resources that you can attend

theshipshow.com

The Ship Show is a twice-monthly podcast, featuring discussion on everything from build engineering to devops to release management, plus interviews, new tools and techniques, and reviews.



foodfightshow.org

Food Fight is a bi-weekly podcast for the Chef community. We bring together the smartest people in the Chef community and the broader DevOps world to discuss the thorniest issues in system administration.



LAB

What do I do now?



To learn more you are going to need more practice and more resources. It takes a village and a couple of web servers.

OBJECTIVE:

- ✓ Provide you with resources that you can read
- ✓ Provide you with resources that you can watch
- ✓ Provide you with resources that you can listen
- ❑ Provide you with resources that you can attend

Chef Developers' IRC Meeting



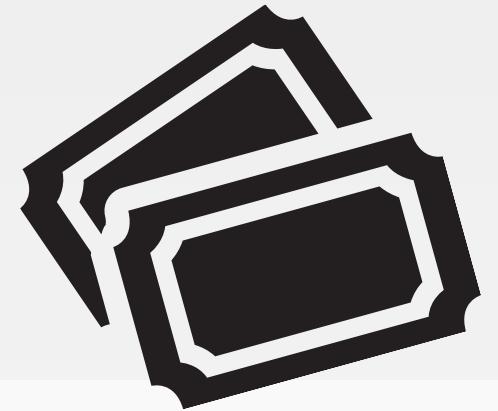
Join members of the Chef Community in a meeting for Chef Developers where we'll discuss the future of the Chef project and other things pertinent to the community.

<irc.freenode.net#chef-hacking>

v3.0.0

<https://github.com/chef/chef-community-irc-meetings>



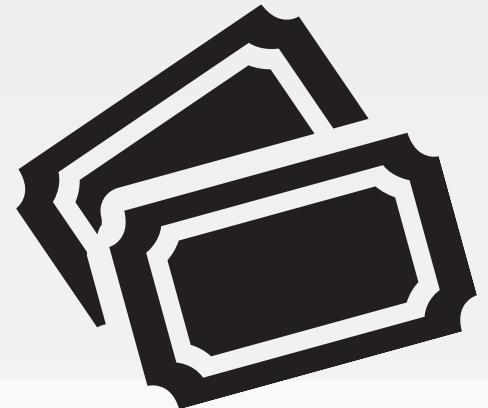


Chef Community Summit

The Chef Community will gather for two days of open space sessions and brainstorm on Chef best practices.

The Chef Community Summit is a facilitated Open Space event. The participants of the summit propose topics, organize an agenda, and discuss and work on the ideas that are most important to the community.

ChefConf



It's the gathering of hundreds of Chef community members. We get together to learn about the latest and greatest in the industry (both the hows and the whys), as well as exchange ideas, brainstorm solutions, and give hugs, which has become the calling card of the DevOps community, and the Chef community in particular.

DISCUSSION

Feedback!



feedback.chef.io

Chef Training Feedback

Survey in your inbox

v3.0.0



DISCUSSION

Discussion



What questions can we answer for you?