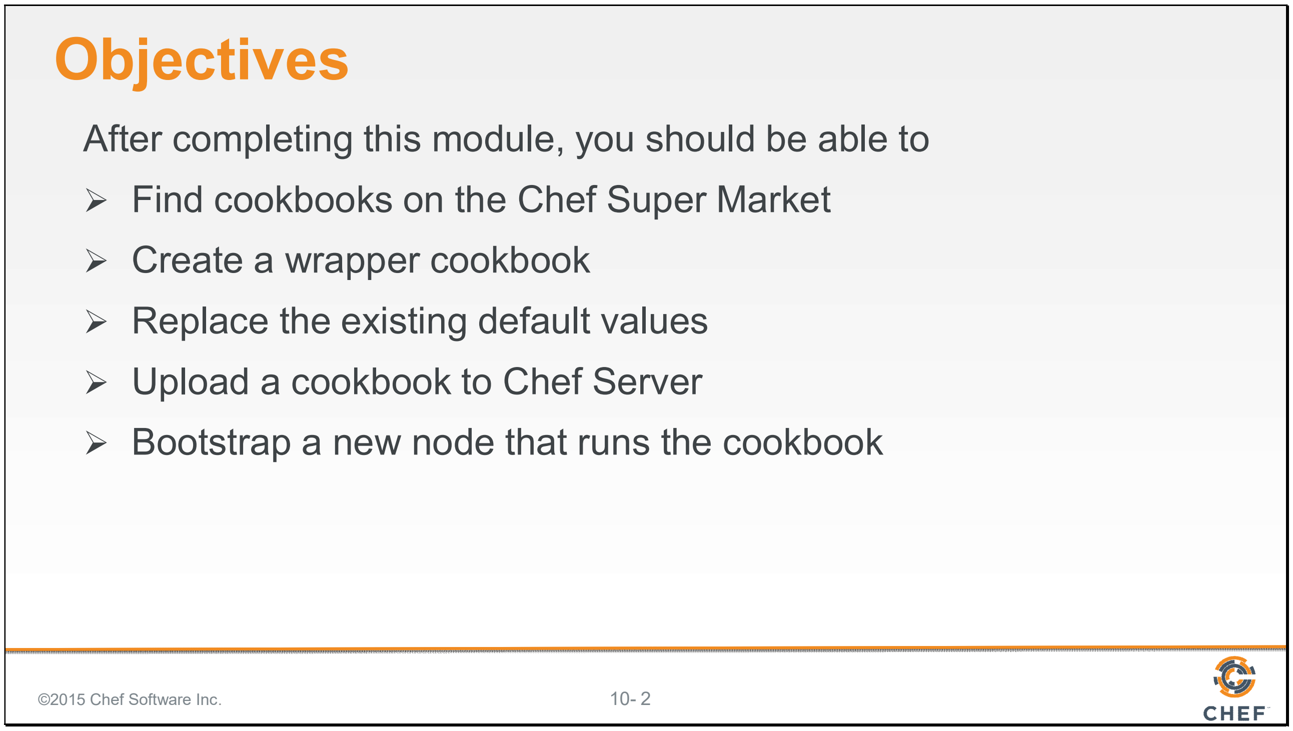
10: Community Cookbooks

Community Cookbooks

Find, Explore and View Chef Cookbooks

Slide 2



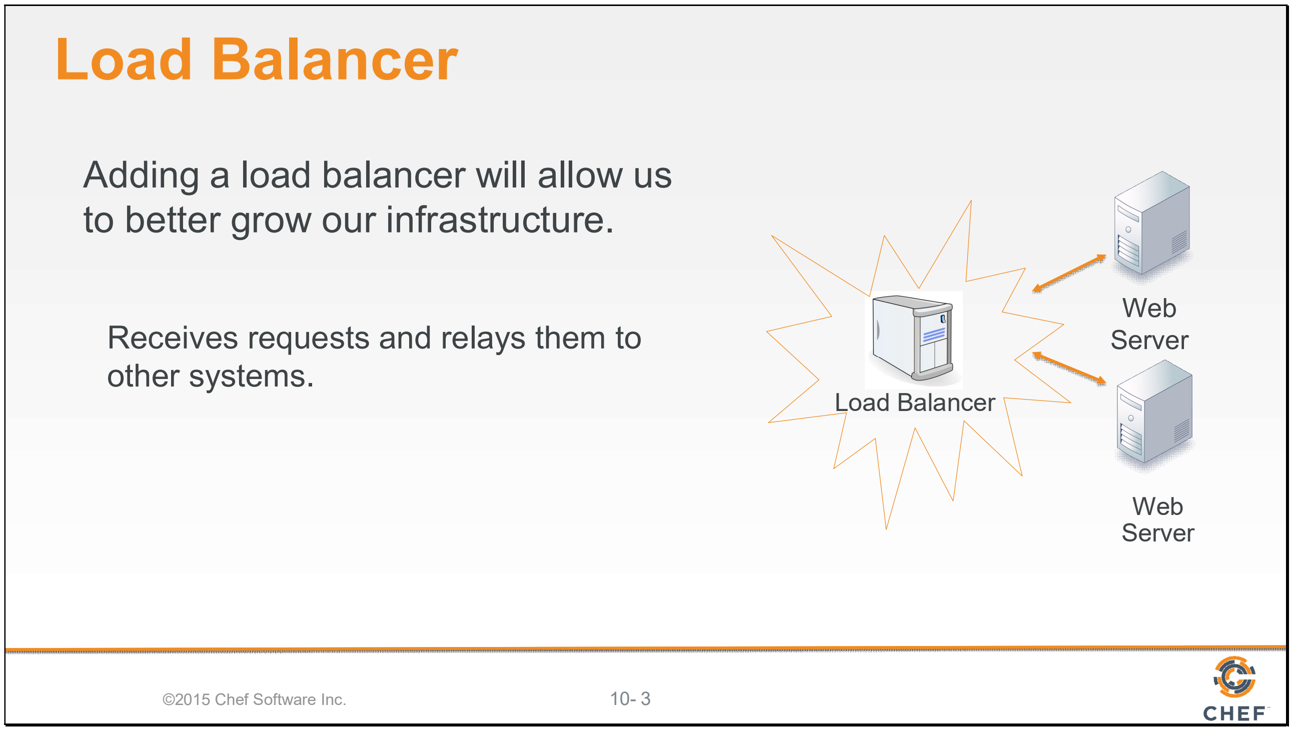
Objectives

After completing this module, you should be able to

- ➢ Find cookbooks on the Chef Super Market
- ➢ Create a wrapper cookbook
- ➢ Replace the existing default values
- ➢ Upload a cookbook to Chef Server
- ➢ Bootstrap a new node that runs the cookbook

In this module you will learn how to find cookbooks on the Chef Super Market, create a wrapper cookbook, replace the existing default values, upload a cookbook to Chef Server, and bootstrap a new node that runs the cookbook

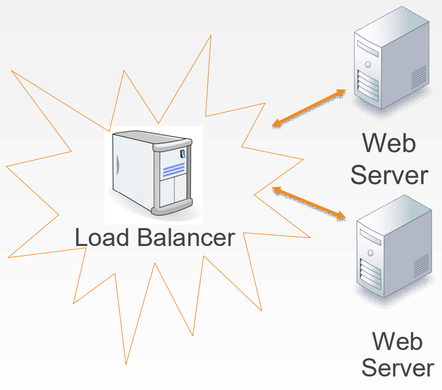
Slide 3



Load Balancer

Adding a load balancer will allow us to better grow our infrastructure.

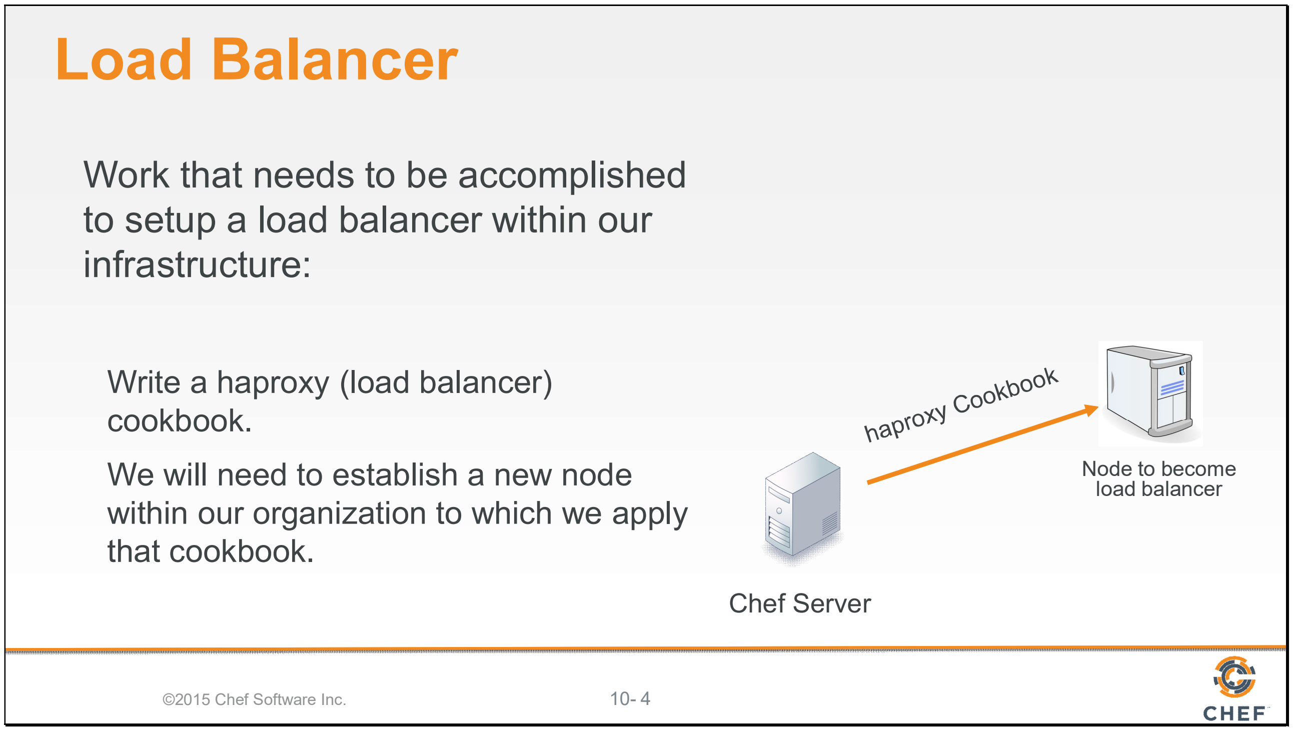
Receives requests and relays them to other systems.



With a single web server running with our organization, it's now time to talk about the next goal to tackle. We need to setup a load balancer.

A load balancer is able to receive requests and relay them to other systems. In our case, we specifically want to use the load balancer to balance the entire traffic load between one or more systems.

This means we will need to establish a new node within our organization, install the necessary software to make the node a load balancer, and configure it so that it will relay requests to our existing node running apache and to future nodes.

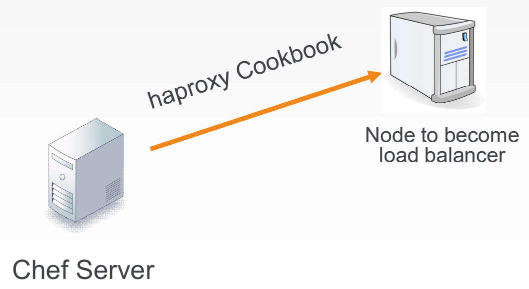


Load Balancer

Work that needs to be accomplished to setup a load balancer within our infrastructure:

Write a haproxy (load balancer) cookbook.

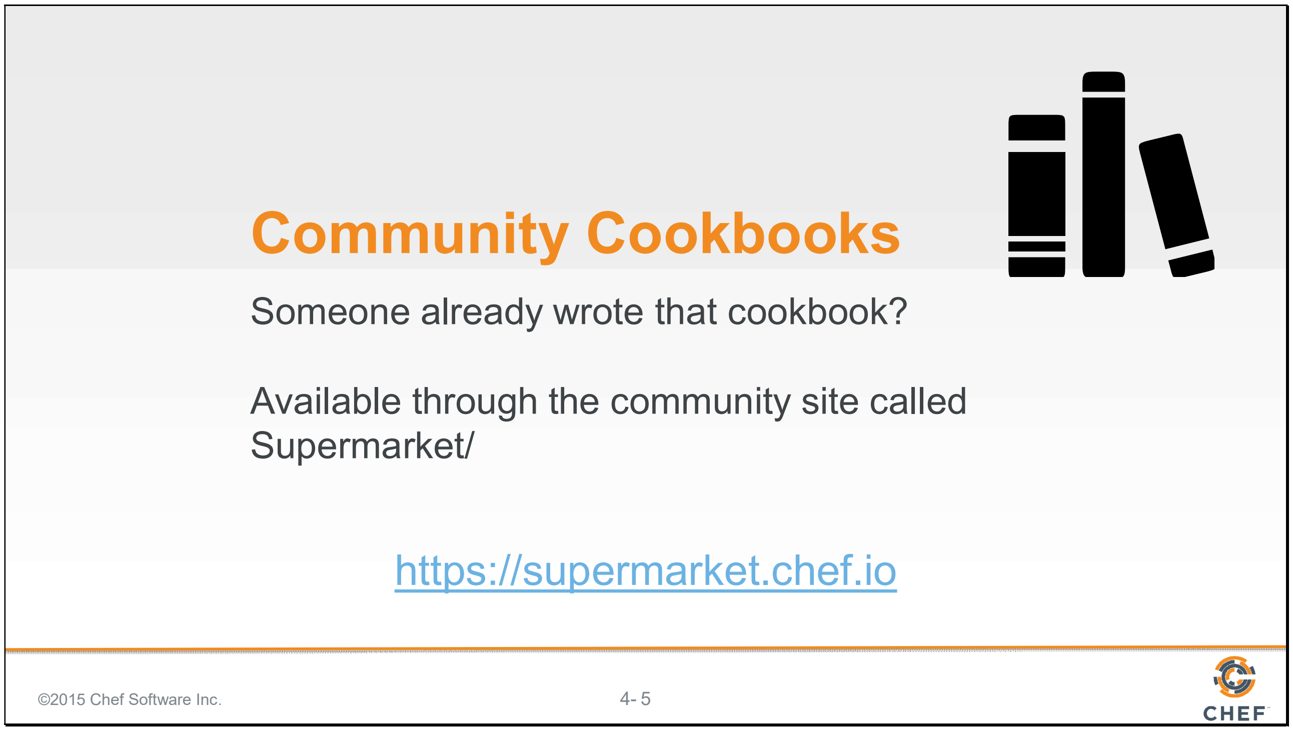
We will need to establish a new node within our organization to which we apply that cookbook.



Similar to how we installed and configured apache on our first node, we could do the same thing here with a load balancer. We could learn the package name for the application 'haproxy', learn which file manages the configuration, learn how to compose the configuration with custom values, and then manage the service.

Package, Template and Service are the core of configuration management. Nearly all the recipes you write for an application will center on using these three resources. We could spend some time focused on composing the cookbook recipe and testing it on our platform with our custom configuration.

Slide 5



Community Cookbooks

Someone already wrote that cookbook?

Available through the community site called Supermarket/

https://supermarket.chef.io

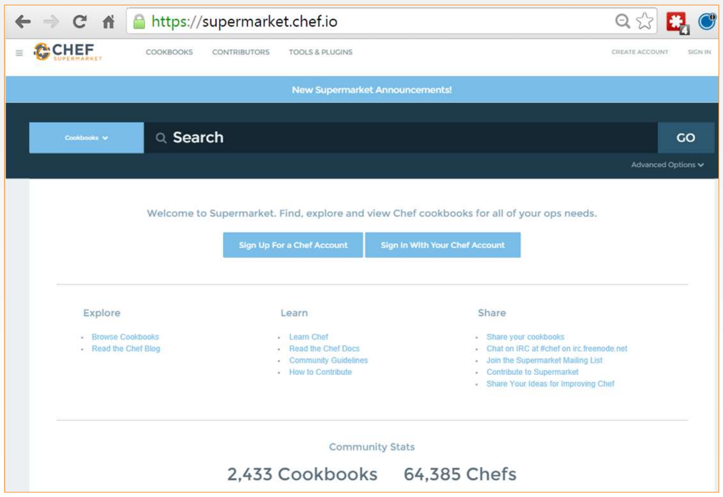
But what if we told you someone already wrote that cookbook?

Someone already has and that cookbook is available through the community site called Supermarket. Supermarket is a public repository of all the cookbooks shared by other developers, teams, and companies who want to share their knowledge and hard work with you to save you time.

Slide 6

Community Cookbooks

- Community cookbooks are managed by individuals.
- Chef does not verify or approve cookbooks in the Supermarket.
- Cookbooks may not work for various reasons.
- Still, there are real benefits to community cookbooks.


https://supermarket.chef.io
CHEF SUPERMARKET
COOKBOOKS
CONTRIBUTORS
TOOLS & PLUGINS
CREATE ACCOUNT
SIGN IN
New Supermarket Announcements!
Cookbooks
Search
GO
Advanced Options
Sign Up For a Chef Account
Sign In With Your Chef Account

Welcome to Supermarket. Find, explore and view Chef cookbooks for all of your ops needs.

Explore
- Browse Cookbooks
- Read the Chef Blog

Learn
- Learn Chef
- Read the Chef Docs
- Community Guidelines
- How to Contribute

Share
- Share your cookbooks
- Chat on IRC at #chef on irc.freenode.net
- Join the Supermarket Mailing List
- Contribute to Supermarket
- Share Your Ideas for Improving Chef

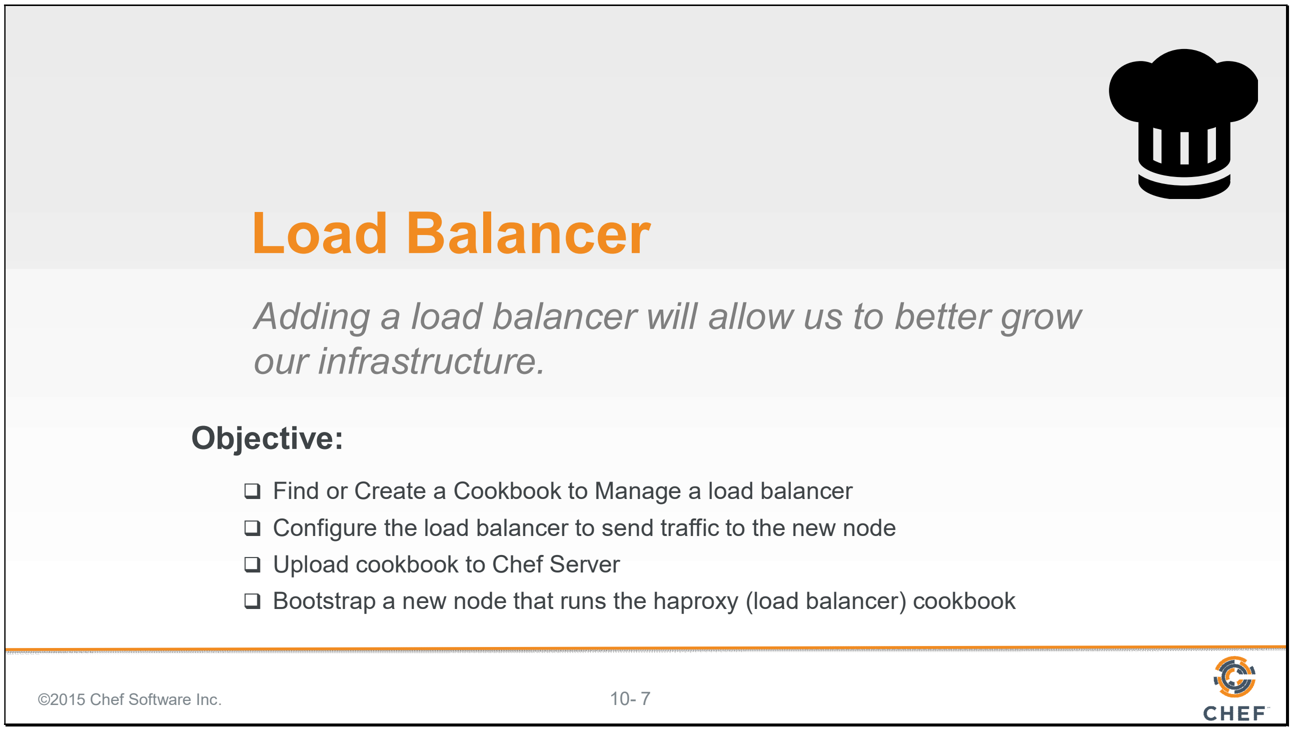
Community Stats

2,433 Cookbooks 64,385 Chefs

An important thing to remember is that on the community site are cookbooks managed by individuals. Chef does not verify or approve the cookbooks found in the Supermarket. These cookbooks solved problems for the original authors and then they decided to share them. This means that the cookbooks you find in the Supermarket may not be built or designed for your platform. It may not take into special consideration your needs and requirements. It may no longer be actively maintained.

Even if the cookbook does not work as a whole, there is still value in reading and understand the source code and extracting the pieces you need when creating your own. With all that said, there is a real benefit to the community site. When you find a cookbook that helps you deliver value quickly, it can be a tremendous boon to your productivity. This is what we are going to take advantage of with the haproxy cookbook.

Slide 7



Load Balancer

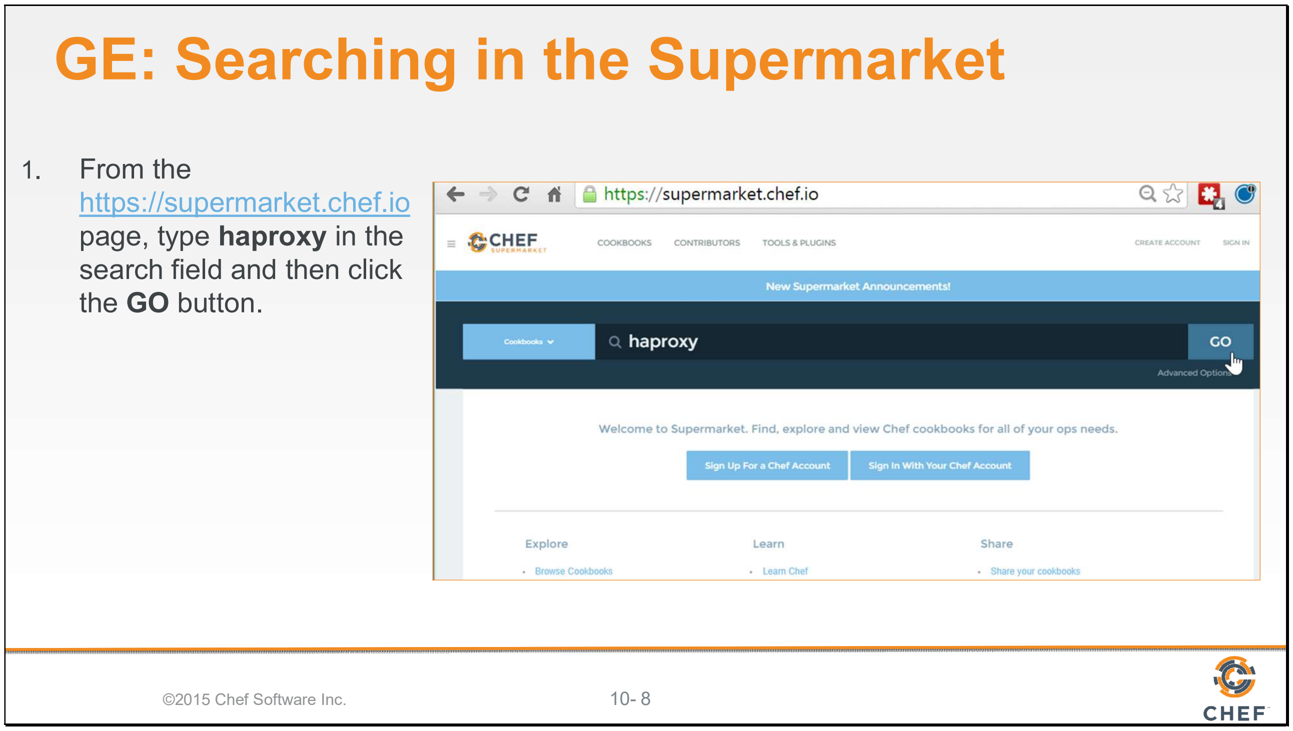
Adding a load balancer will allow us to better grow our infrastructure.

Objective:

- ❑ Find or Create a Cookbook to Manage a load balancer
- ❑ Configure the load balancer to send traffic to the new node
- ❑ Upload cookbook to Chef Server
- ❑ Bootstrap a new node that runs the haproxy (load balancer) cookbook

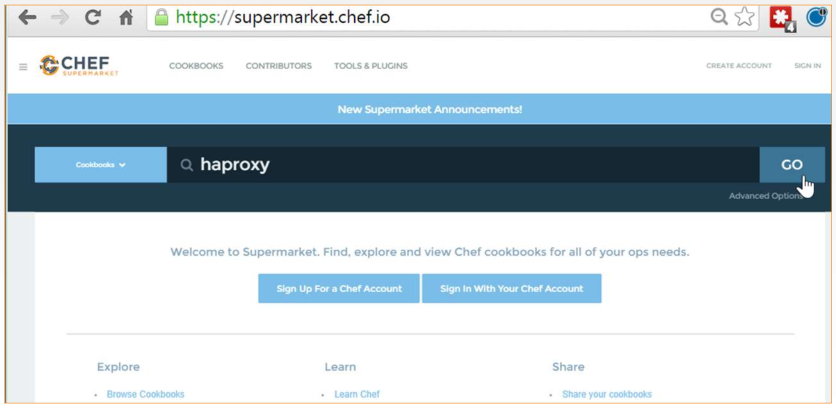
Let’s find the haproxy (load balancer) cookbook within the community site to learn more about it.

Slide 8



GE: Searching in the Supermarket

1. From the https://supermarket.chef.io page, type **haproxy** in the search field and then click the **GO** button.

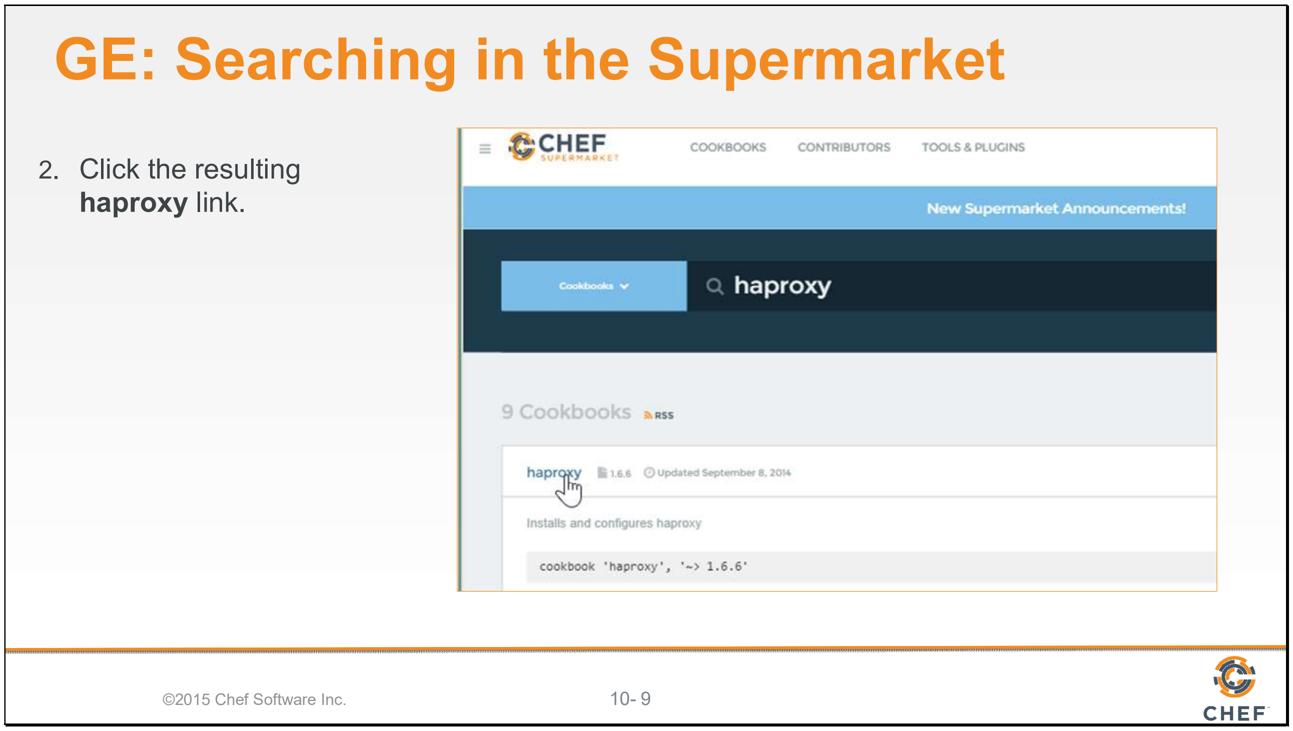

https://supermarket.chef.io
CHEF SUPERMARKET
COOKBOOKS
CONTRIBUTORS
TOOLS & PLUGINS
CREATE ACCOUNT
SIGN IN
New Supermarket Announcements!
Cookbooks
haproxy
GO
Advanced Options
Welcome to Supermarket. Find, explore and view Chef cookbooks for all of your ops needs.
Sign Up For a Chef Account
Sign In With Your Chef Account
Explore
Browse Cookbooks
Learn
Learn Chef
Share
Share your cookbooks

©2015 Chef Software Inc. 10- 8

From the Supermarket main page type the search term "haproxy" and the click the **GO** button.

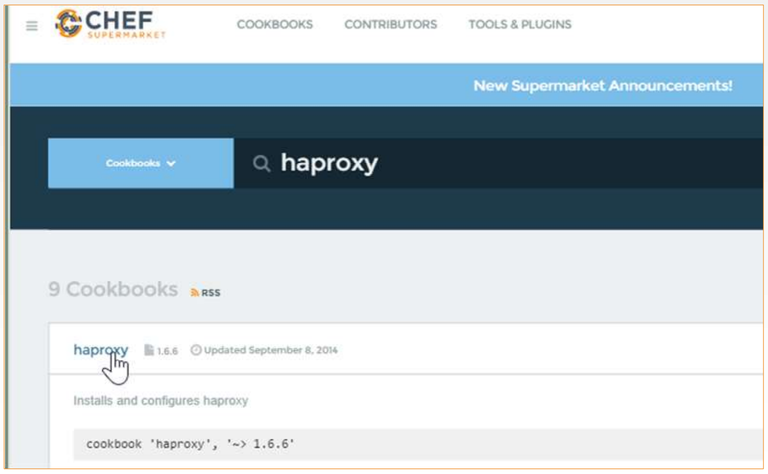
Below the search term will show us all the matching cookbooks. The haproxy cookbook is in that result set.

Slide 9



GE: Searching in the Supermarket

2. Click the resulting **haproxy** link.



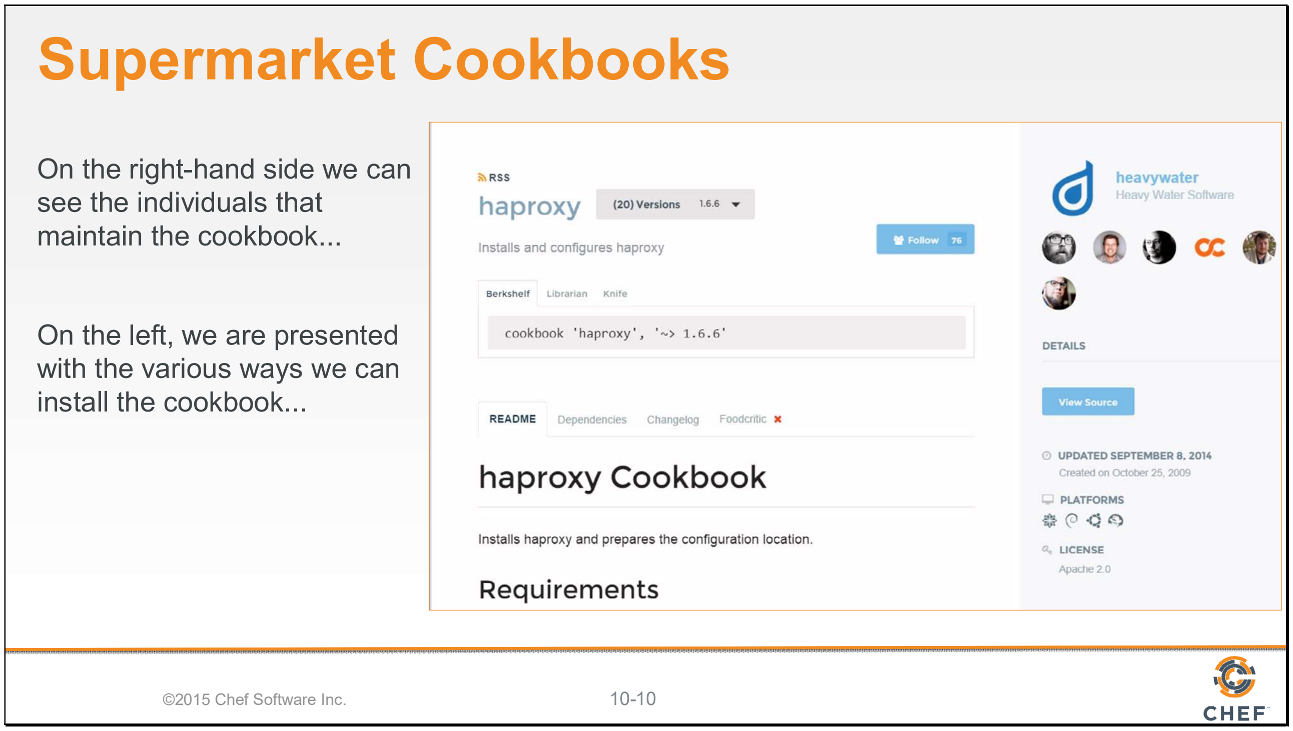
haproxy 1.6.6 Updated September 8, 2014

Installs and configures haproxy

```
cookbook 'haproxy', '~> 1.6.6'
```

Cookbooks usually map one-to-one to a piece of software and usually are named after the piece of software that they manage. Select the cookbook named haproxy from the search results.

Slide 10



Supermarket Cookbooks

On the right-hand side we can see the individuals that maintain the cookbook...

On the left, we are presented with the various ways we can install the cookbook...

RSS

haproxy (20) Versions 1.6.6

Installs and configures haproxy

Follow 76

Berkshelf Librarian Knife

```
cookbook 'haproxy', '~> 1.6.6'
```

README Dependencies Changelog Foodcritic

haproxy Cookbook

Installs haproxy and prepares the configuration location.

Requirements

heavywater
Heavy Water Software

DETAILS

View Source

UPDATED SEPTEMBER 8, 2014
Created on October 25, 2009

PLATFORMS

LICENSE
Apache 2.0

At this point you are presented with information that describes the cookbook. Starting on the right-hand side we see the individuals that maintain the cookbook, a link to view the source details, last updated date, supported platforms, licensing, and a link to download the cookbook.

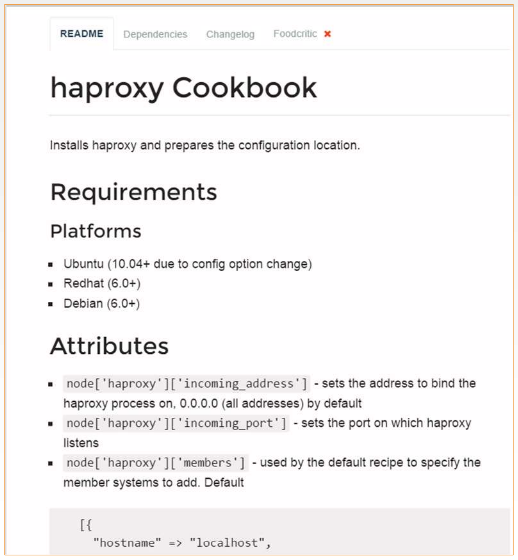
On the left, we are presented with the various ways we can install the cookbook, the README that describes information about the cookbook, any cookbooks that this cookbook may depend on, a history of the changes, and its food critic rating--which is a code evaluator for best practices.

Slide 11

Supermarket Cookbooks

The area to focus most of your attention from the beginning is the README.

Reading and understanding the README at a glance is difficult. It is a skill that comes with time.



haproxy Cookbook

Installs haproxy and prepares the configuration location.

Requirements

Platforms

- Ubuntu (10.04+ due to config option change)
- Redhat (6.0+)
- Debian (6.0+)

Attributes

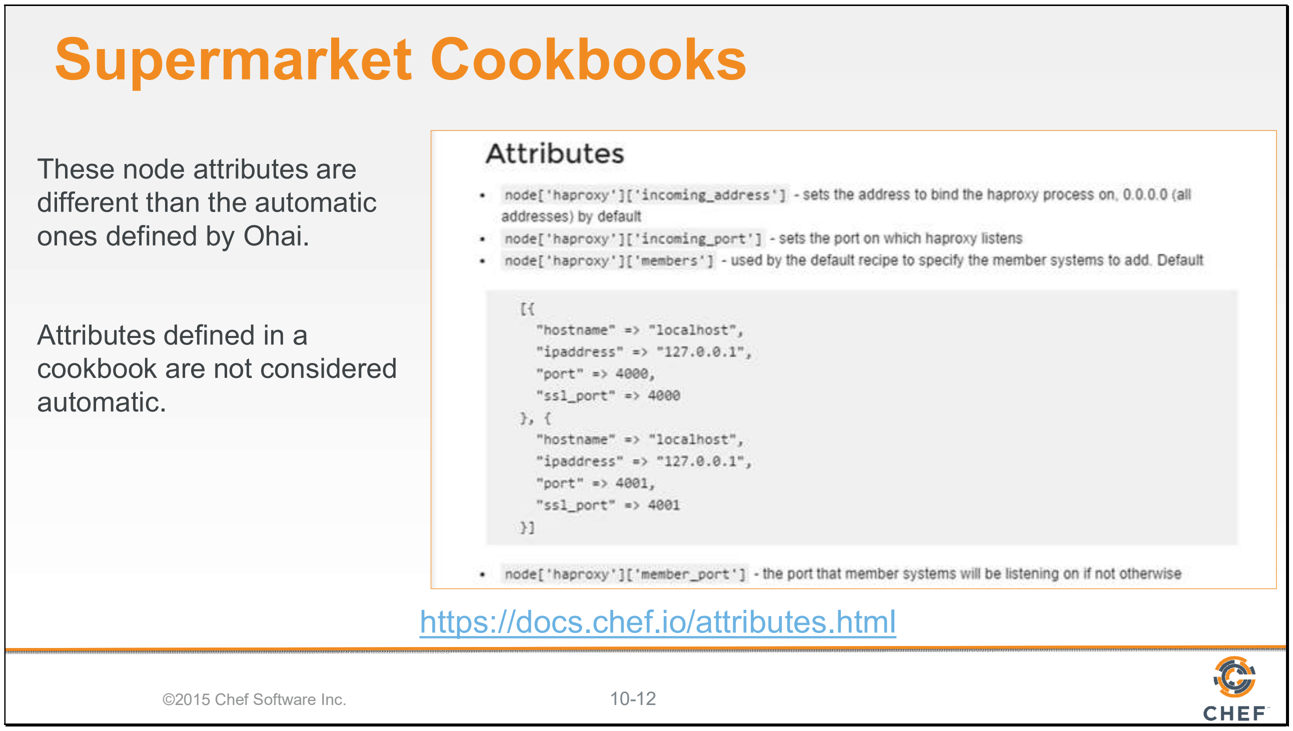
- `node['haproxy']['incoming_address']` - sets the address to bind the haproxy process on, 0.0.0.0 (all addresses) by default
- `node['haproxy']['incoming_port']` - sets the port on which haproxy listens
- `node['haproxy']['members']` - used by the default recipe to specify the member systems to add. Default

```
[{
  "hostname" => "localhost",
```

The area to focus most of your attention from the beginning is the README. The README describes the various attributes that are defined within the cookbook and the purpose of the recipe. This is the same README file found in the cookbooks we currently have within our organization. This one, however, has had far more details added to give new users like us the ability to understand more quickly what the cookbook does and how it does it.

Reading and understanding the README at a glance is difficult. It is a skill that comes with time. For the haproxy cookbook, there is an defined attribute that establishes the members that receive the proxy requests from the load balancer. This is available in a node attribute available through `node['haproxy']['members']`.

Slide 12



Supermarket Cookbooks

These node attributes are different than the automatic ones defined by Ohai.

Attributes defined in a cookbook are not considered automatic.

Attributes

- `node['haproxy']['incoming_address']` - sets the address to bind the haproxy process on, 0.0.0.0 (all addresses) by default
- `node['haproxy']['incoming_port']` - sets the port on which haproxy listens
- `node['haproxy']['members']` - used by the default recipe to specify the member systems to add. Default

```
[{
  "hostname" => "localhost",
  "ipaddress" => "127.0.0.1",
  "port" => 4000,
  "ssl_port" => 4000
}, {
  "hostname" => "localhost",
  "ipaddress" => "127.0.0.1",
  "port" => 4001,
  "ssl_port" => 4001
}]
```

- `node['haproxy']['member_port']` - the port that member systems will be listening on if not otherwise

https://docs.chef.io/attributes.html

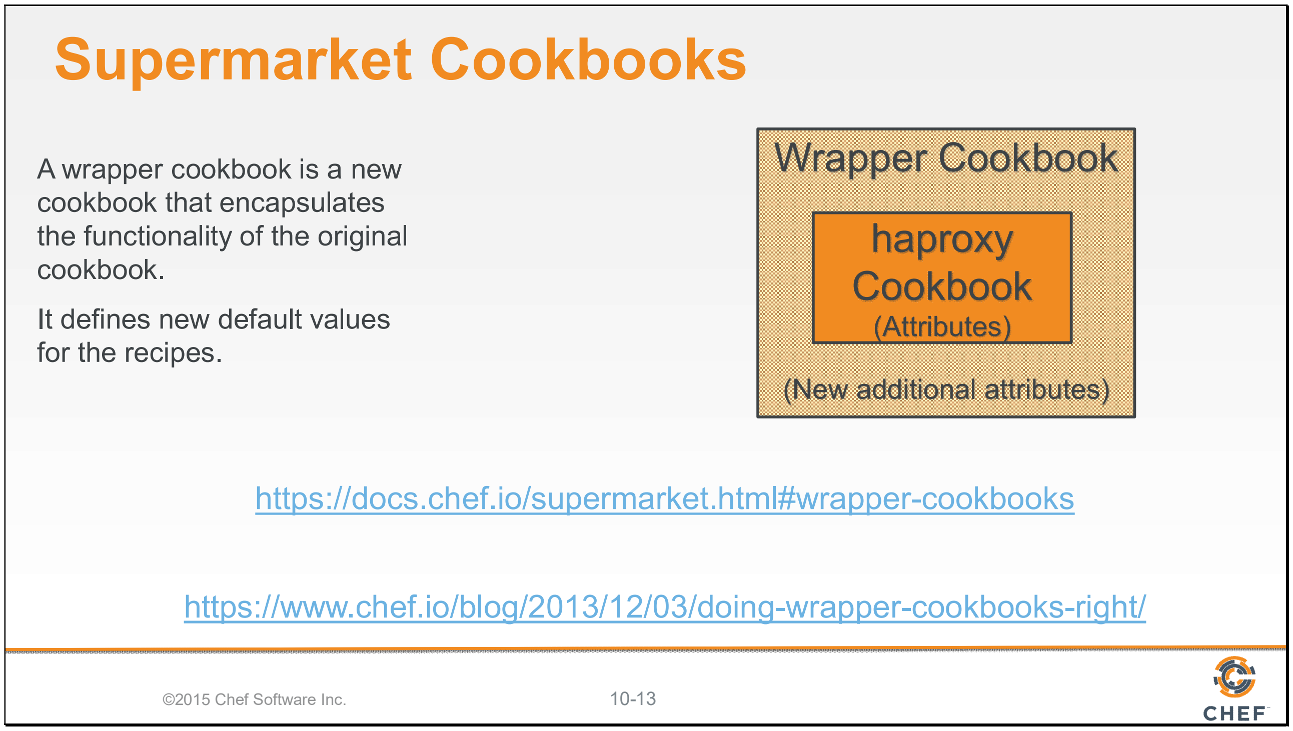
©2015 Chef Software Inc. 10-12

CHEF

Prior to this point we have seen how node attributes are defined by Ohai but cookbooks also have this ability to define node attributes. These node attributes are different than the ones defined by Ohai as well. Ohai attributes are considered automatic attributes and generally inalienable characteristics about the node.

Attributes defined in a cookbook are not considered automatic. They are simply default values that we may change. There are many ways that we provide new default values for these. One way that we will learn is defining a wrapper cookbook.

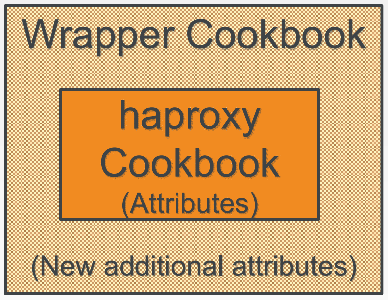
Slide 13



Supermarket Cookbooks

A wrapper cookbook is a new cookbook that encapsulates the functionality of the original cookbook.

It defines new default values for the recipes.



https://docs.chef.io/supermarket.html#wrapper-cookbooks

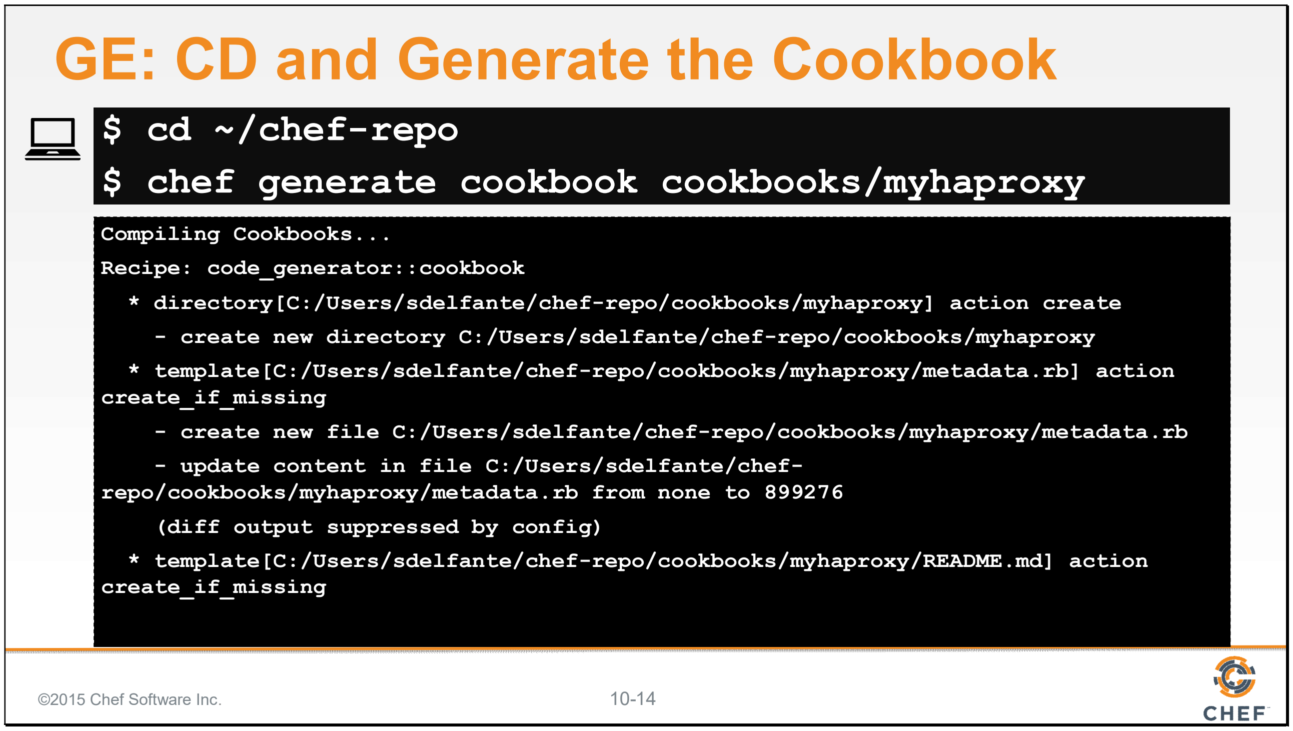
https://www.chef.io/blog/2013/12/03/doing-wrapper-cookbooks-right/

A wrapper cookbook is a new cookbook that encapsulates the functionality of the original cookbook but allows us to define new default values for the recipes.

This is a common method for overriding cookbooks because it allows us to leave the original cookbook untouched. We simply provide new default values that we want and then include the recipes that we want to run.

Let's generate our wrapper cookbook named myhaproxy. Traditionally we would name the cookbook with a prefix of the name of our company and then follow it by the cookbook name 'company-cookbook'.

Slide 14



GE: CD and Generate the Cookbook

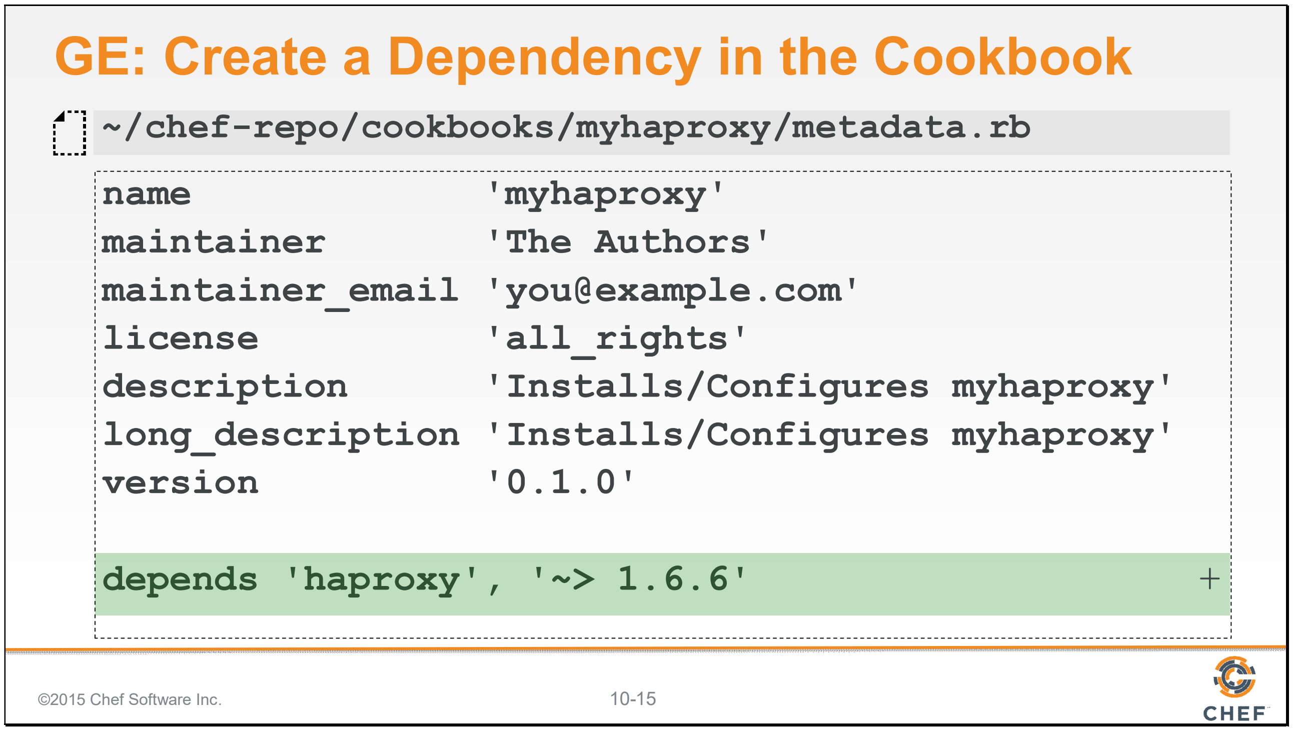
```
$ cd ~/chef-repo
$ chef generate cookbook cookbooks/myhaproxy
```

```
Compiling Cookbooks...
Recipe: code_generator::cookbook
  * directory[C:/Users/sdelfante/chef-repo/cookbooks/myhaproxy] action create
    - create new directory C:/Users/sdelfante/chef-repo/cookbooks/myhaproxy
  * template[C:/Users/sdelfante/chef-repo/cookbooks/myhaproxy/metadata.rb] action
create_if_missing
    - create new file C:/Users/sdelfante/chef-repo/cookbooks/myhaproxy/metadata.rb
    - update content in file C:/Users/sdelfante/chef-
repo/cookbooks/myhaproxy/metadata.rb from none to 899276
    (diff output suppressed by config)
  * template[C:/Users/sdelfante/chef-repo/cookbooks/myhaproxy/README.md] action
create_if_missing
```

CHEF

Change to your chef-repo directory and then generate your new cookbook.

Slide 15



GE: Create a Dependency in the Cookbook

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

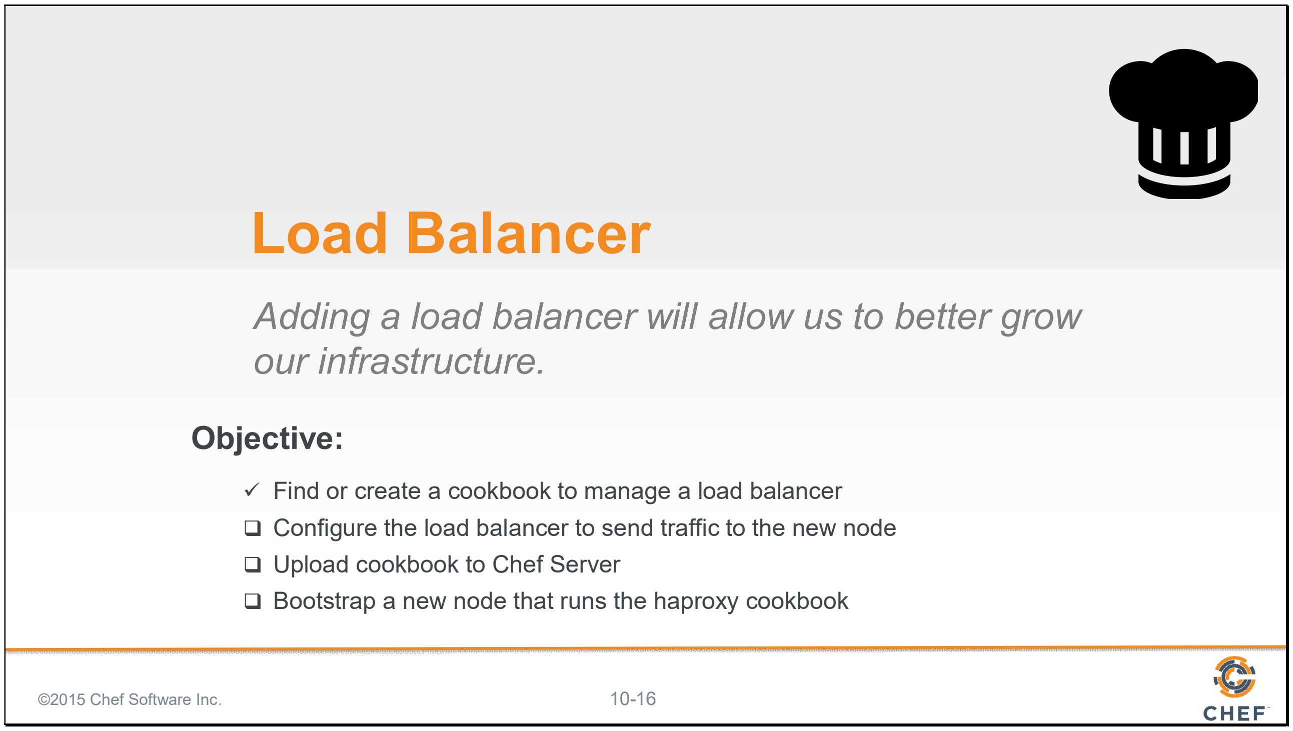
```
name             'myhaproxy'
maintainer       'The Authors'
maintainer_email 'you@example.com'
license          'all_rights'
description      'Installs/Configures myhaproxy'
long_description 'Installs/Configures myhaproxy'
version          '0.1.0'

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

©2015 Chef Software Inc. 10-15

Set up a dependency within your haproxy cookbook. Establishing this dependency informs the Chef Server that whenever you deliver this cookbook to a node, you should also deliver with it the mentioned dependent cookbooks.

This is important because your cookbook is simply going to set up new default values and then execute the recipes defined in the original cookbook.

Slide 16



Load Balancer

Adding a load balancer will allow us to better grow our infrastructure.

Objective:

- ✓ Find or create a cookbook to manage a load balancer
- ❑ Configure the load balancer to send traffic to the new node
- ❑ Upload cookbook to Chef Server
- ❑ Bootstrap a new node that runs the haproxy cookbook

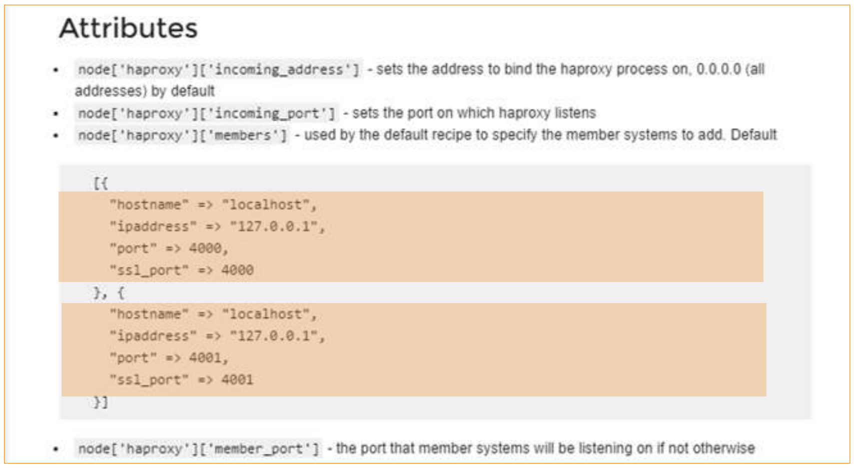
Now that you have the dependency on the haproxy cookbook in your wrapper cookbook, you need to learn what new default values you need to add to the recipe.

Slide 17

Supermarket Cookbooks

Currently, the haproxy cookbook assumes that there are two different services running on the localhost at port 4000 and port 4001.

In a moment, you'll need to change that.



Attributes

- `node['haproxy']['incoming_address']` - sets the address to bind the haproxy process on, 0.0.0.0 (all addresses) by default
- `node['haproxy']['incoming_port']` - sets the port on which haproxy listens
- `node['haproxy']['members']` - used by the default recipe to specify the member systems to add. Default

```
[{
  "hostname" => "localhost",
  "ipaddress" => "127.0.0.1",
  "port" => 4000,
  "ssl_port" => 4000
}, {
  "hostname" => "localhost",
  "ipaddress" => "127.0.0.1",
  "port" => 4001,
  "ssl_port" => 4001
}]
```

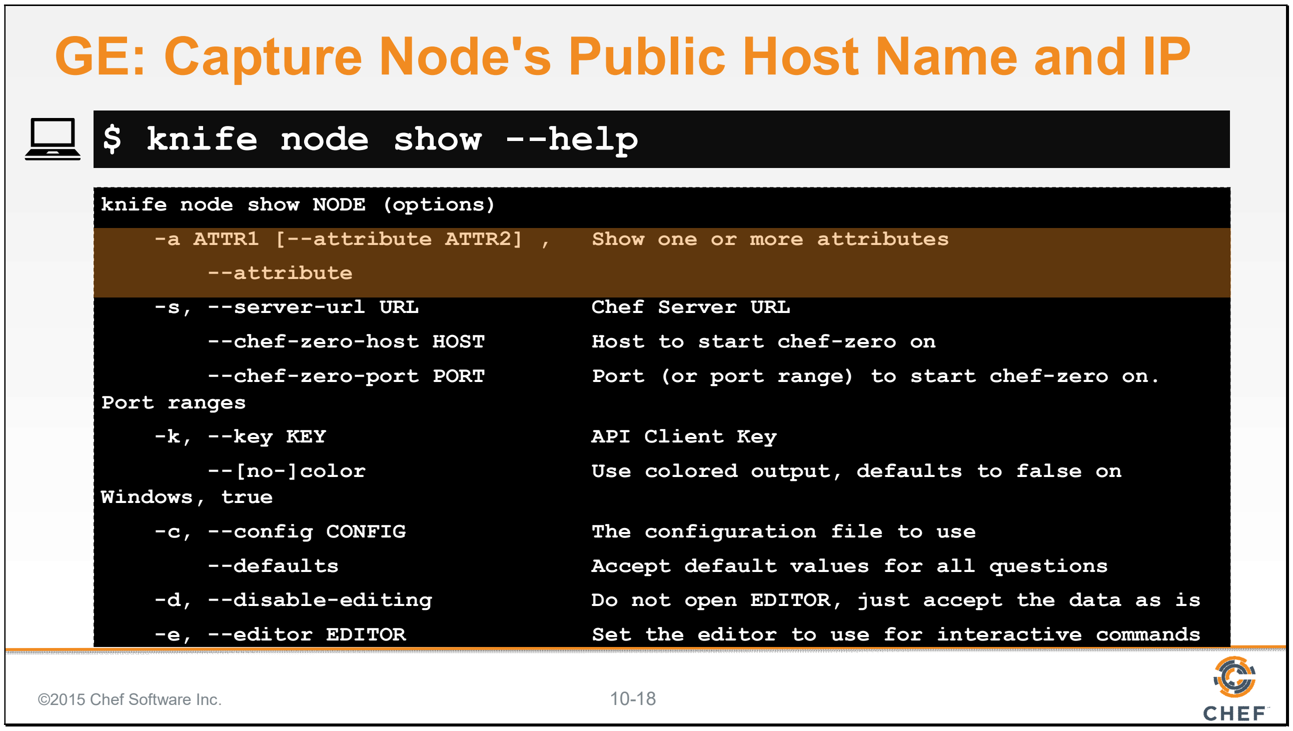
- `node['haproxy']['member_port']` - the port that member systems will be listening on if not otherwise

https://docs.chef.io/supermarket.html#wrapper-cookbooks

Currently the haproxy cookbook assumes that there are two different services running on the localhost at port 4000 and port 4001. The haproxy process will relay messages to itself to those two ports.

That is not our configuration. First, we currently only have one system that we want to route traffic. Second, we want to have the traffic routed not to localhost but instead to our webserver, node1, which will have a completely different hostname and IP address.

Slide 18



GE: Capture Node's Public Host Name and IP

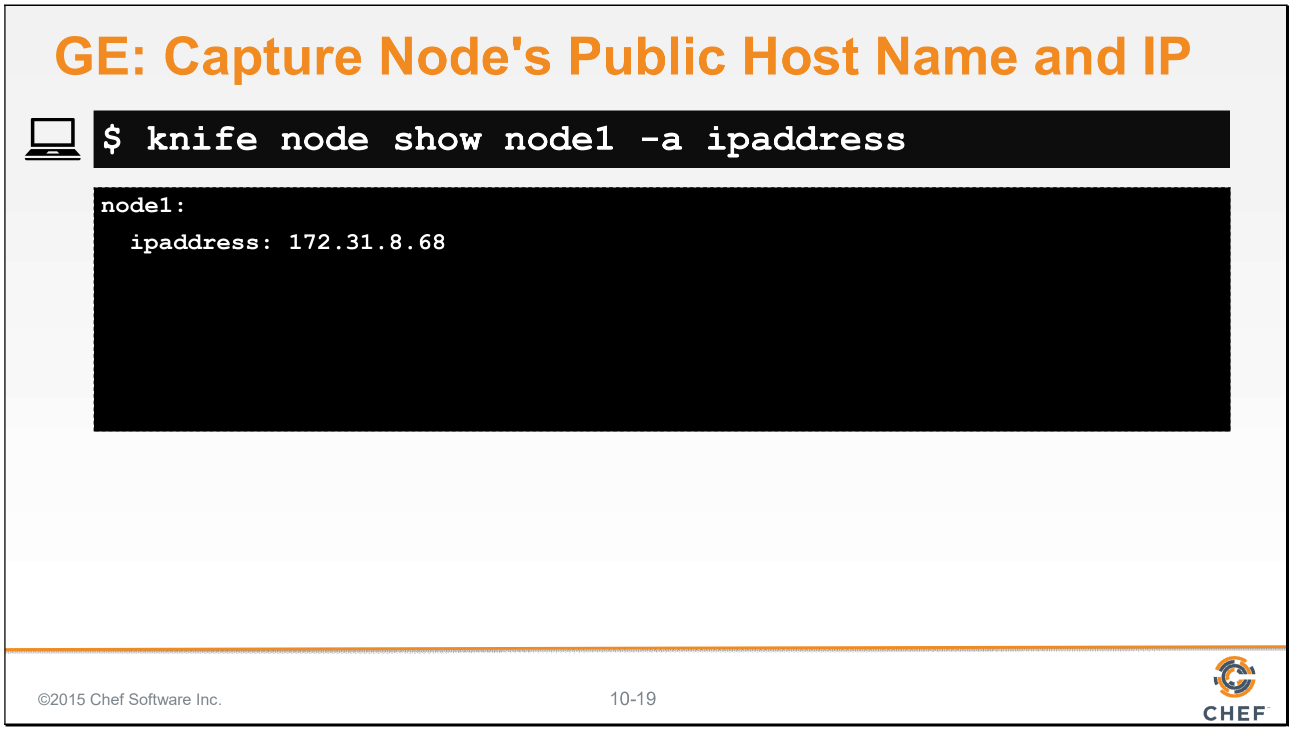
```
$ knife node show --help
```

```
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
```

This new default value for the haproxy members needs to define the information about the webserver node, node1. So you need to capture the node's public host name and public IP address.

The 'knife node show' command will display information about the node. You can ask to see a specific attribute on a node with the –a flag or the --attribute flag.

Slide 19



GE: Capture Node's Public Host Name and IP

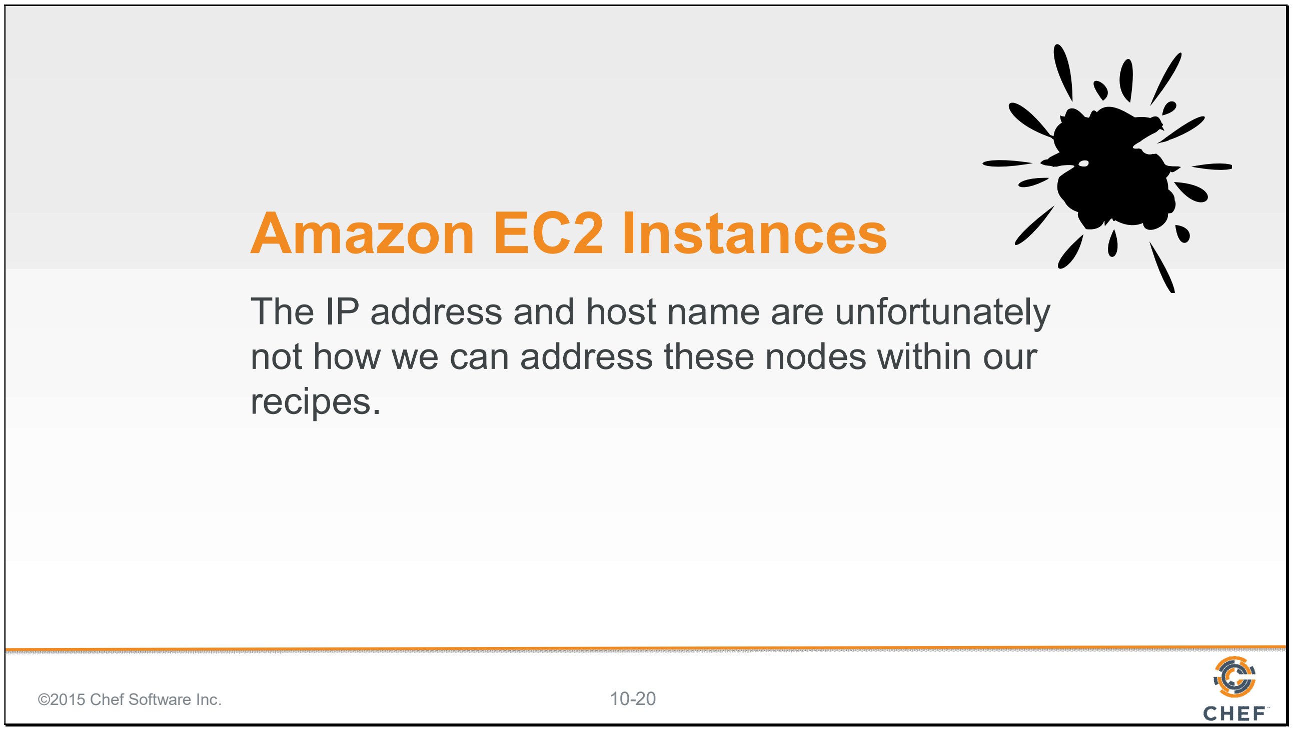
```
$ knife node show node1 -a ipaddress
```

```
node1:
  ipaddress: 172.31.8.68
```

You can display the IP address of node1 with the '-a' flag and specifying the attribute 'ipaddress'.

With cloud providers that generate machines for you often assign internal IP addresses, those values may not work properly.

Slide 20



Amazon EC2 Instances

The IP address and host name are unfortunately not how we can address these nodes within our recipes.

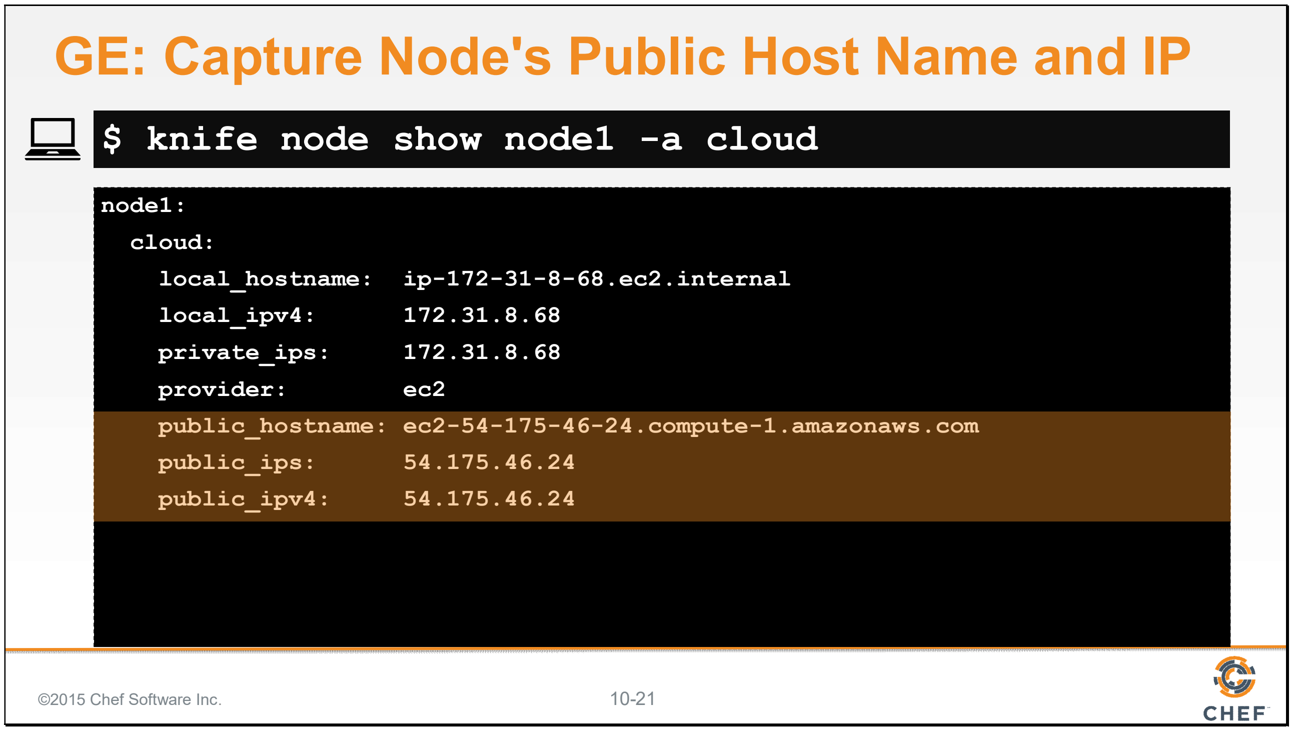
©2015 Chef Software Inc.

10-20

The reason you may need to ask the node for a different set of attributes is that we are using Amazon as a cloud provider for our instances. These instances are displaying the internal IP address when we ask for the ipaddress attribute.

Ohai collects attributes from the current cloud provider and makes them available in an attribute named 'cloud'. We can look at the cloud attribute on our first node and see that it returns for us information about the node.

Slide 21



GE: Capture Node's Public Host Name and IP

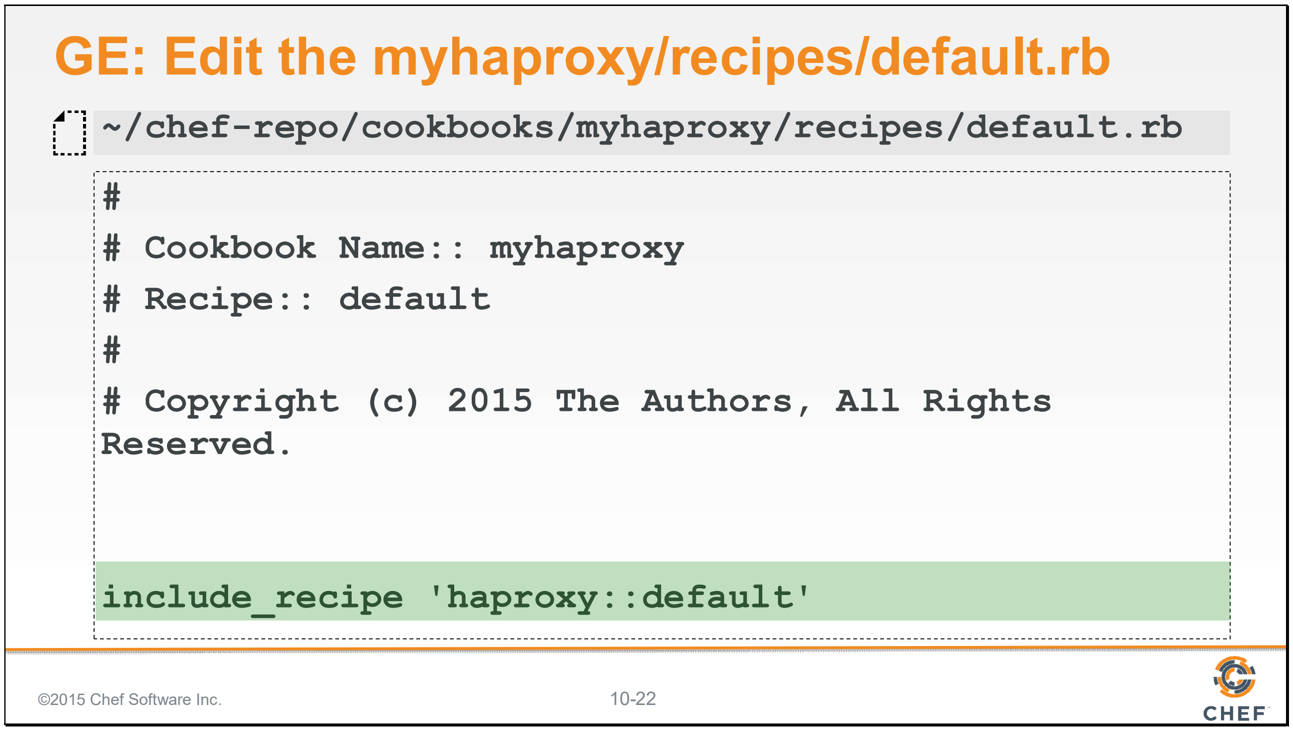
```
$ knife node show node1 -a cloud
```

```
node1:
  cloud:
    local_hostname:  ip-172-31-8-68.ec2.internal
    local_ipv4:      172.31.8.68
    private_ips:     172.31.8.68
    provider:        ec2
    public_hostname: ec2-54-175-46-24.compute-1.amazonaws.com
    public_ips:      54.175.46.24
    public_ipv4:     54.175.46.24
```

If you use 'knife node show' to display the 'cloud' attribute for node1, you will see the local, private, and public connection information.

Capture and write down the public hostname and the public ipv4 address of node1. You will need this in the recipe you are going to write.

Slide 22



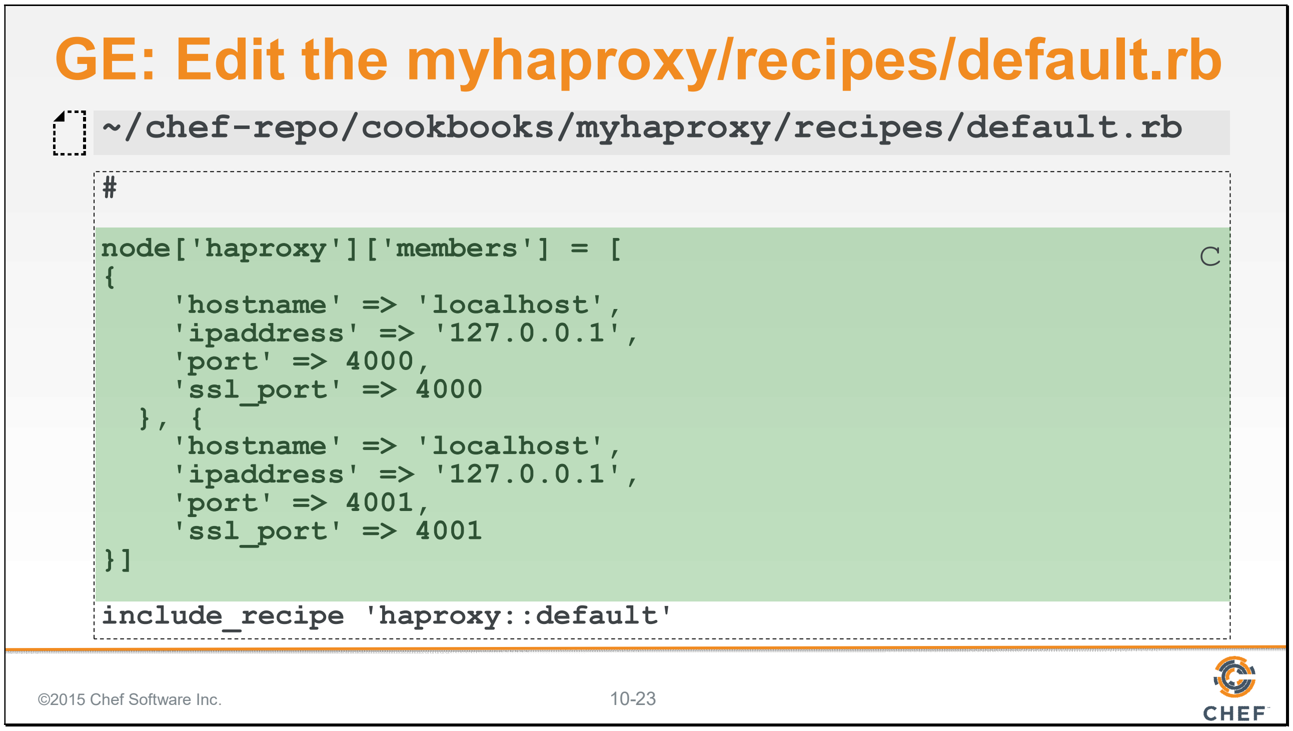
GE: Edit the myhaproxy/recipes/default.rb

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

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


include_recipe 'haproxy::default'
```

First, within the myhaproxy cookbook you will use the include_recipe method to specify the fully-qualified name of the cookbook and recipe that you want to execute. In this case, when you run your wrapped cookbooks recipe, you'll want it to run the original cookbook's default recipe.

Slide 23



GE: Edit the myhaproxy/recipes/default.rb

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

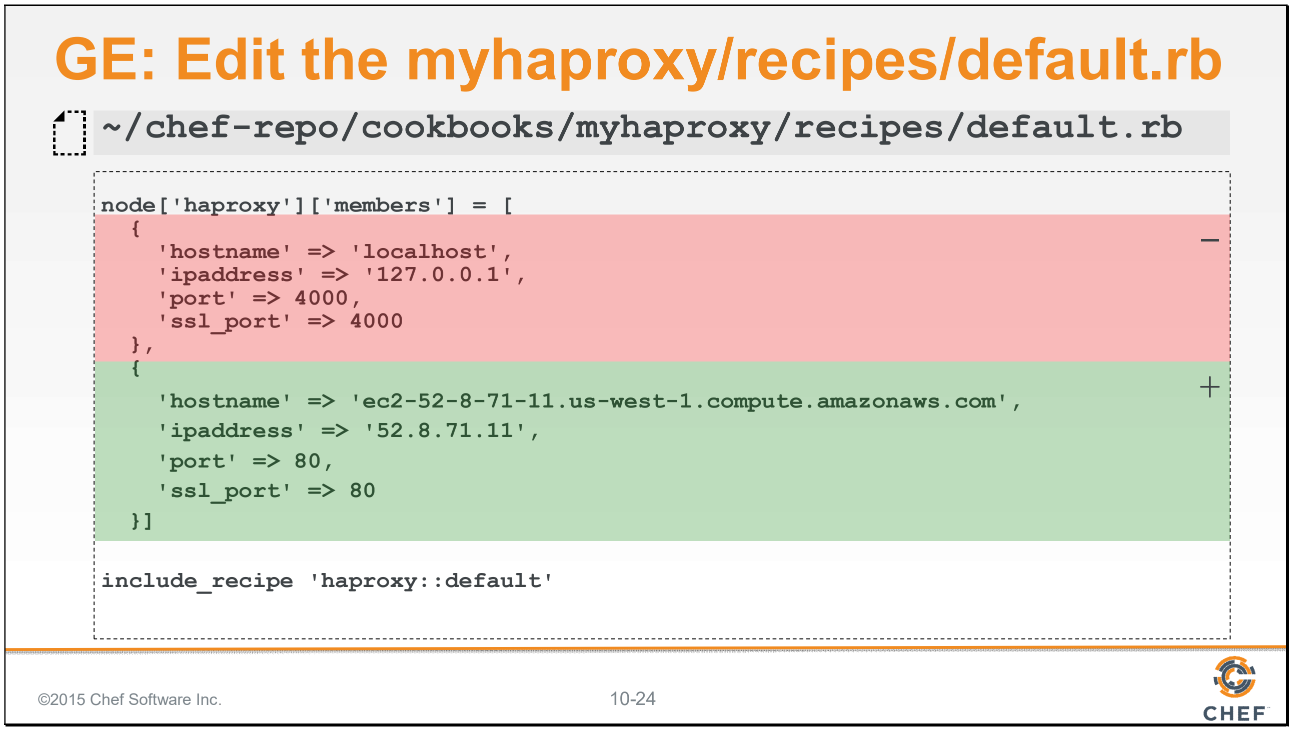
```
#

node['haproxy']['members'] = [
{
    'hostname' => 'localhost',
    'ipaddress' => '127.0.0.1',
    'port' => 4000,
    'ssl_port' => 4000
  }, {
    'hostname' => 'localhost',
    'ipaddress' => '127.0.0.1',
    'port' => 4001,
    'ssl_port' => 4001
}]

include_recipe 'haproxy::default'
```

C

Without changing anything any further, using this cookbook will simply execute the original cookbooks' recipe with all the same default values. Before you execute that recipe, you'll need to override the default values with your own.

Copy and paste the original default values into your recipe, as shown here.

Slide 24



GE: Edit the myhaproxy/recipes/default.rb

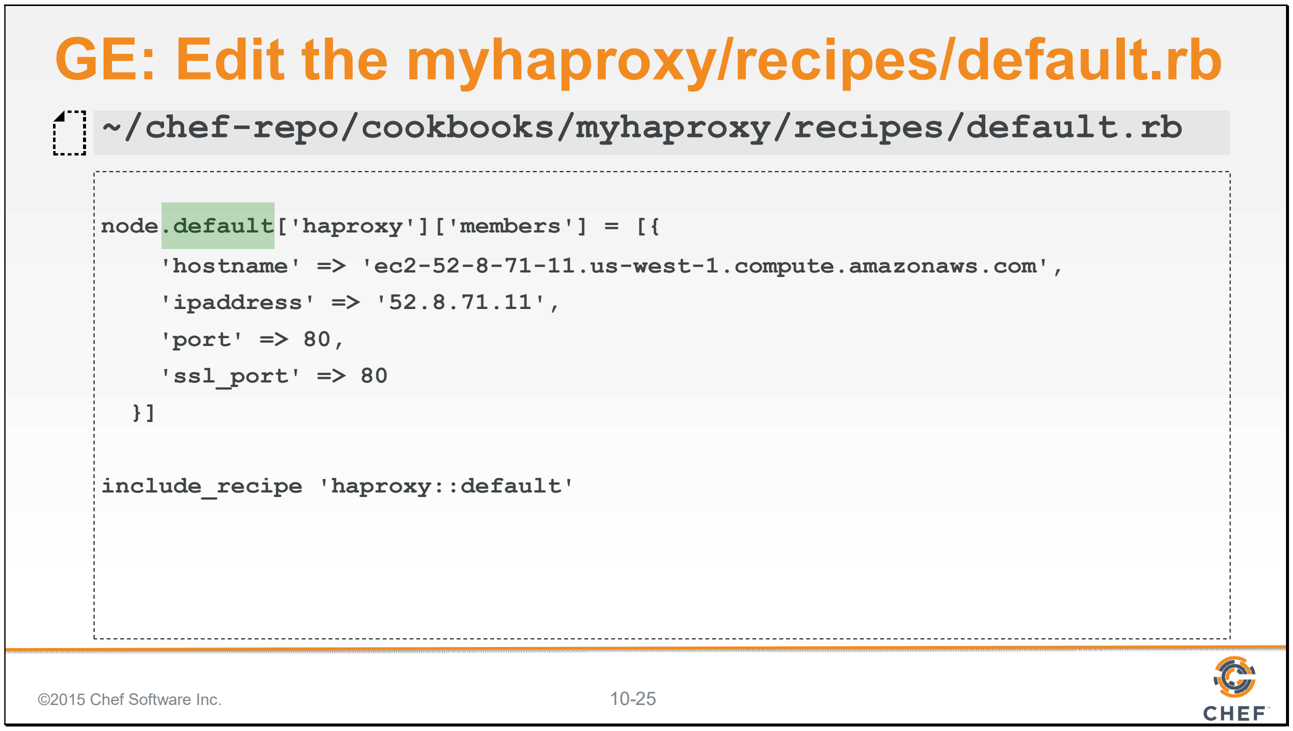
~/chef-repo/cookbooks/myhaproxy/recipes/default.rb

```
node['haproxy']['members'] = [
  {
    'hostname' => 'localhost',
    'ipaddress' => '127.0.0.1',
    'port' => 4000,
    'ssl_port' => 4000
  },
  {
    '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'
```

Remove one of the entries within the members array (shown in red).

Then update the information for the remaining member to include the public ipaddress and hostname for node1 (shown in green).

Slide 25



GE: Edit the myhaproxy/recipes/default.rb

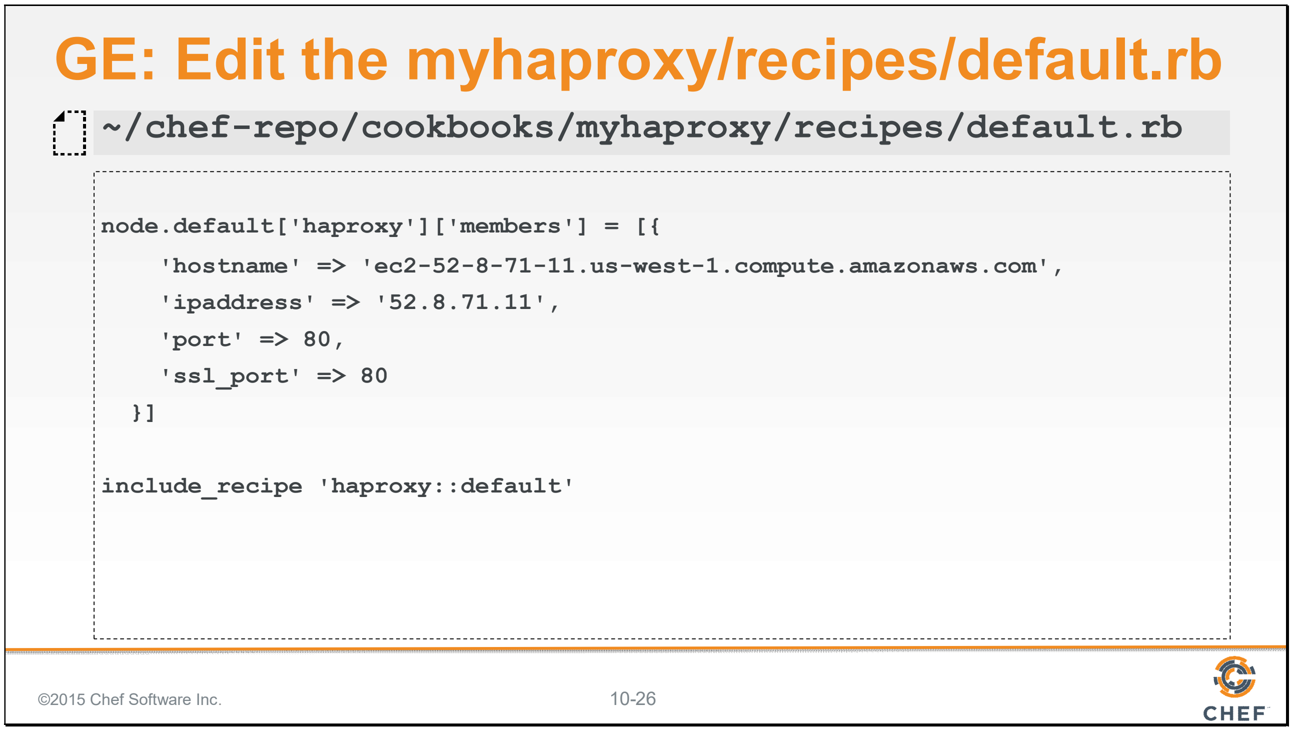
```
~/chef-repo/cookbooks/myhaproxy/recipes/default.rb
```

```
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'
```

To replace a default attribute in a recipe you have to use: 'node.**default**['haproxy']['members']...'

So you need to change: 'node['haproxy']['members']' to 'node.**default**['haproxy']['members']'

Slide 26



GE: Edit the myhaproxy/recipes/default.rb

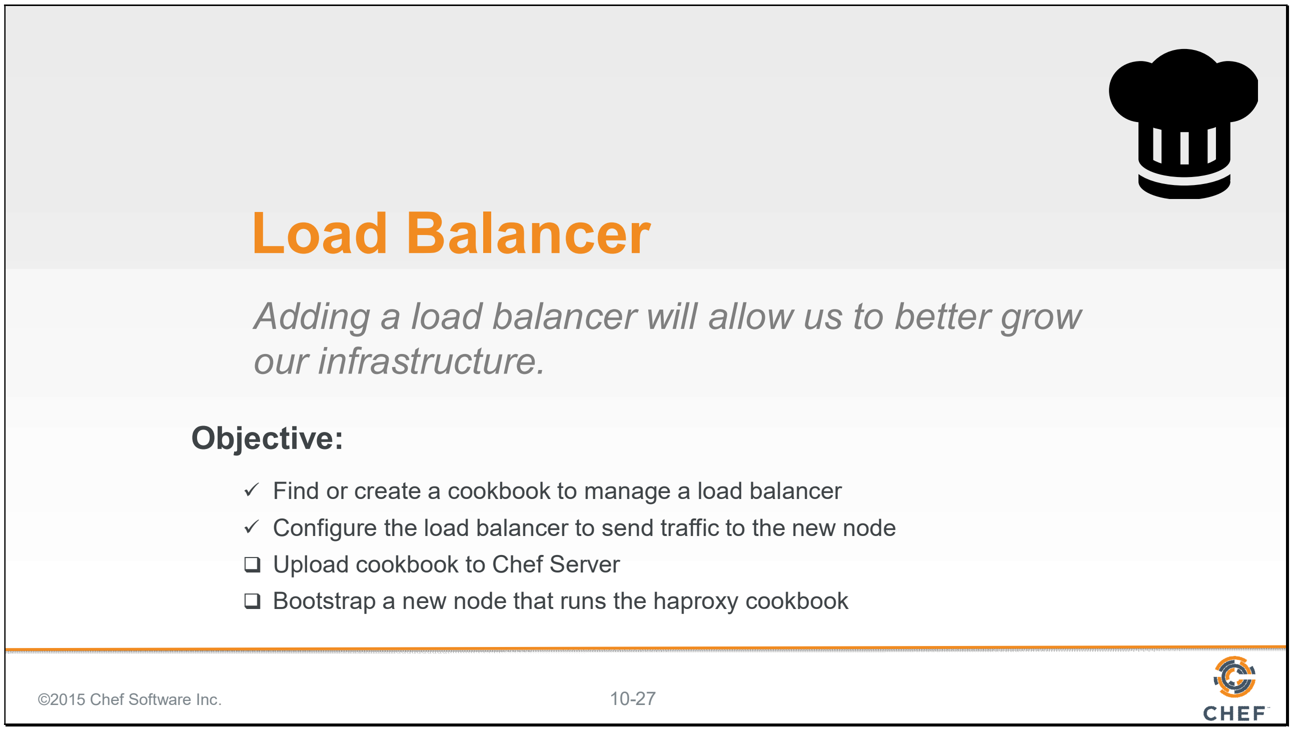
~/chef-repo/cookbooks/myhaproxy/recipes/default.rb

```
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'
```

The final default recipe for the wrapper cookbook 'myhaproxy' looks like the above.

Save your recipe file.

Slide 27



Load Balancer

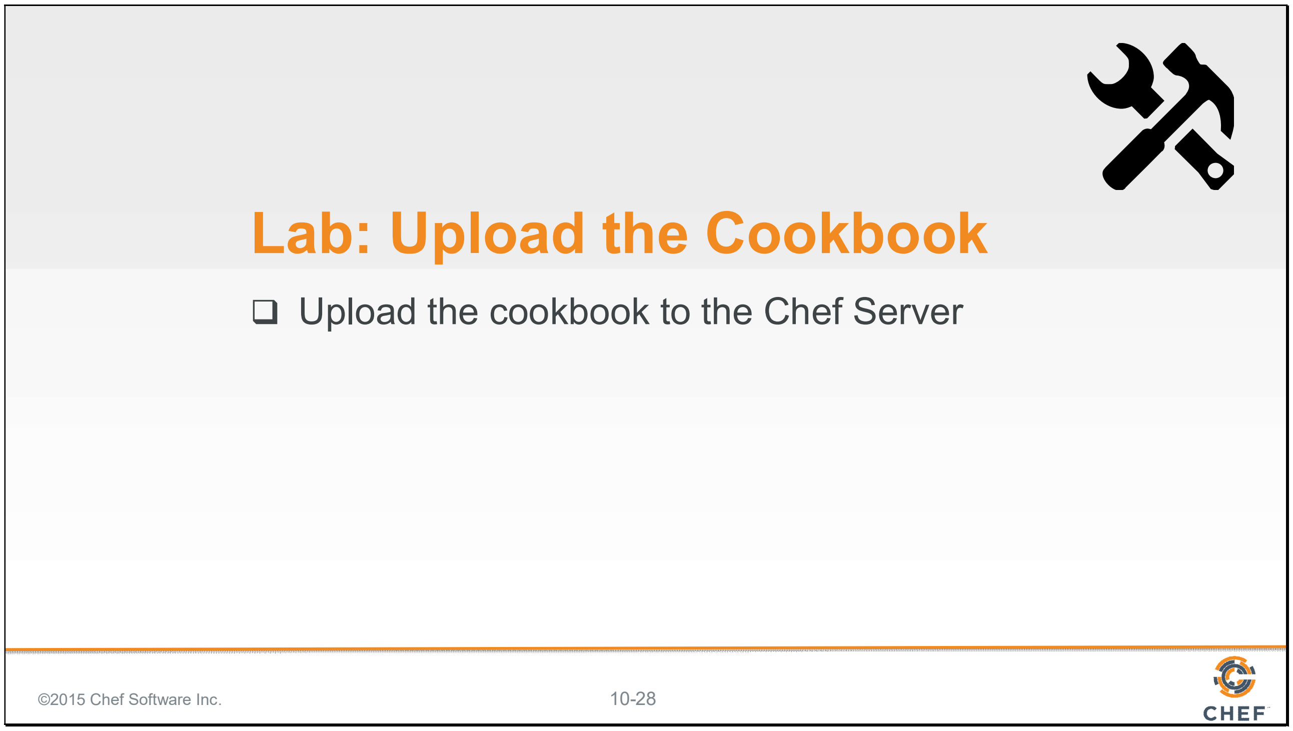
Adding a load balancer will allow us to better grow our infrastructure.

Objective:

- ✓ Find or create a cookbook to manage a load balancer
- ✓ Configure the load balancer to send traffic to the new node
- ❑ Upload cookbook to Chef Server
- ❑ Bootstrap a new node that runs the haproxy cookbook

You have completed creating the wrapper cookbook. It is time to upload to the Chef Server.

Slide 28

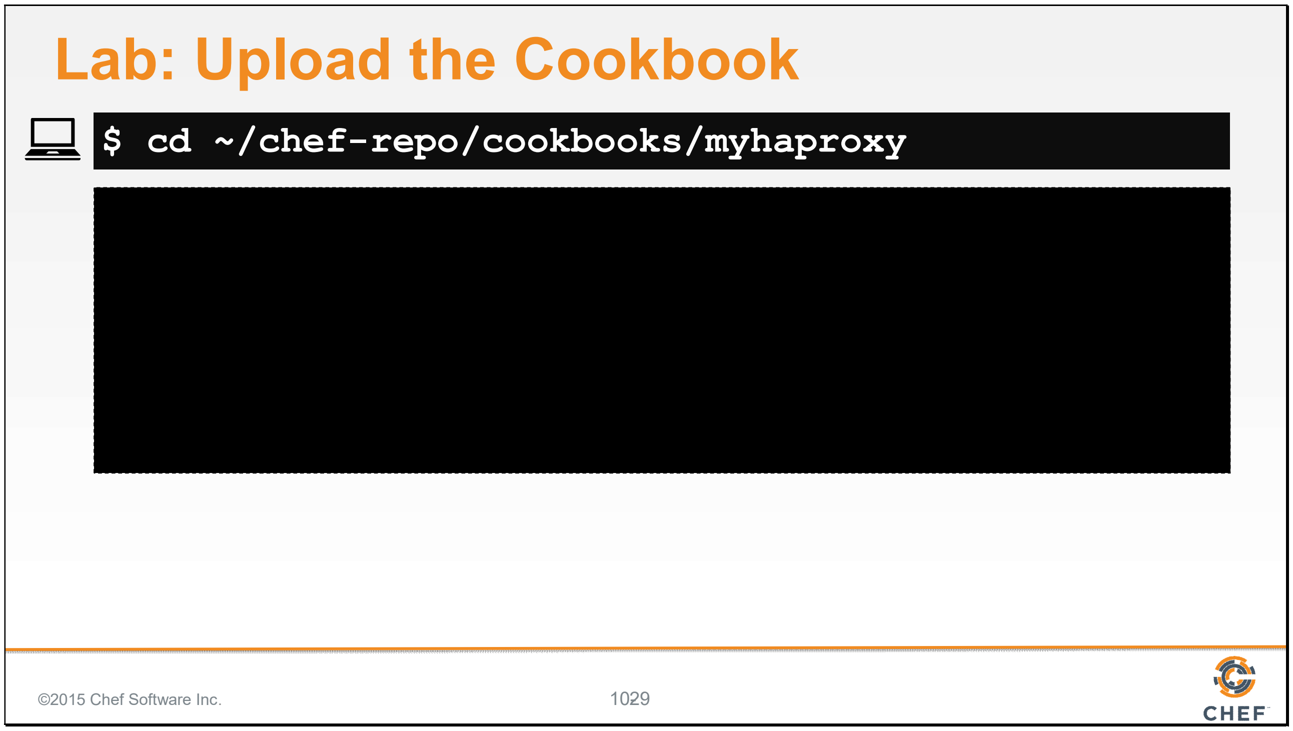


Lab: Upload the Cookbook

❑ Upload the cookbook to the Chef Server

As a lab exercise, upload the cookbook to the Chef Server

Slide 29



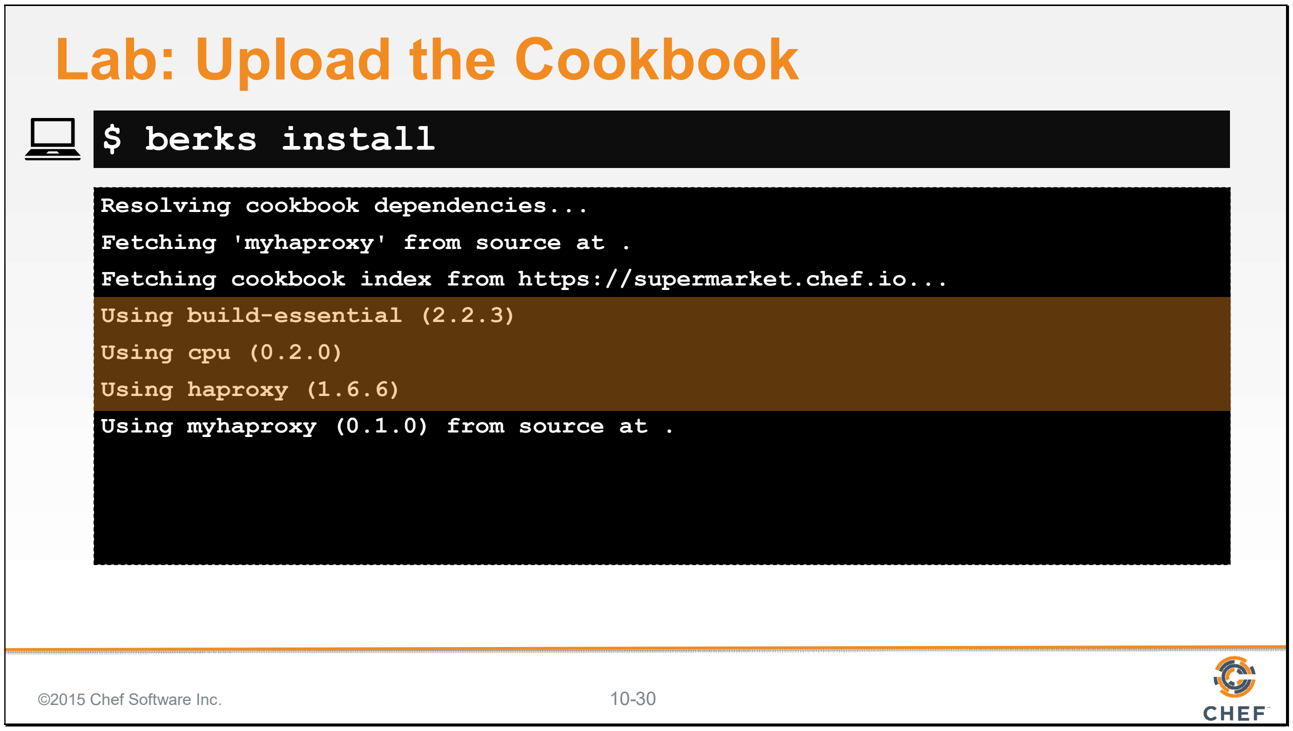
Lab: Upload the Cookbook

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

Let's review that lab.

You change into the directory for the 'myhaproxy' cookbook.

Slide 30



Lab: Upload the Cookbook

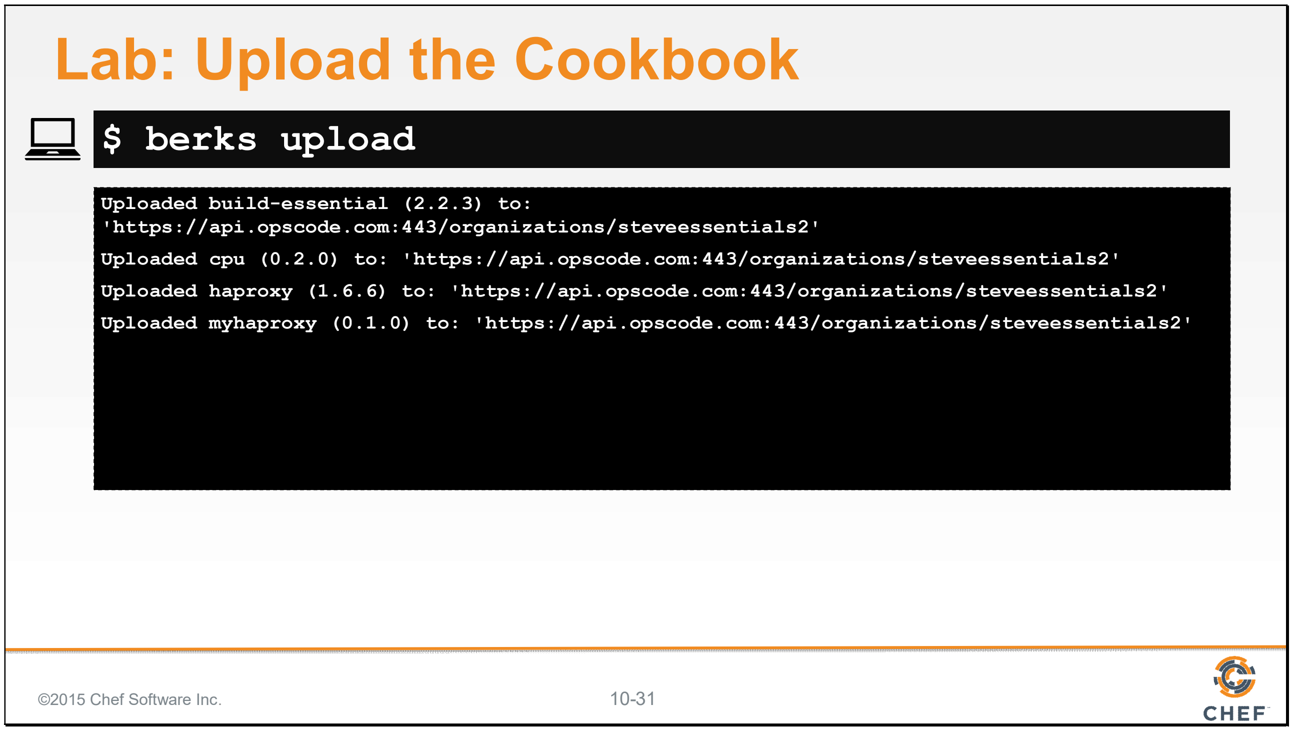
```
$ berks install
```

```
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 (0.1.0) from source at .
```

We use the Berkshelf to upload our cookbooks. This is where Berkshelf really shines as a tool.

Run the command "berks install". When you run this command for a cookbook that has a dependency, you'll see that Berkshelf will download the haproxy cookbook and its dependencies as well. The haproxy cookbook is dependent on the build-essential cookbook and the cpu cookbook. If any of those cookbooks had dependencies, berkshelf would find those and download them as well.

Slide 31



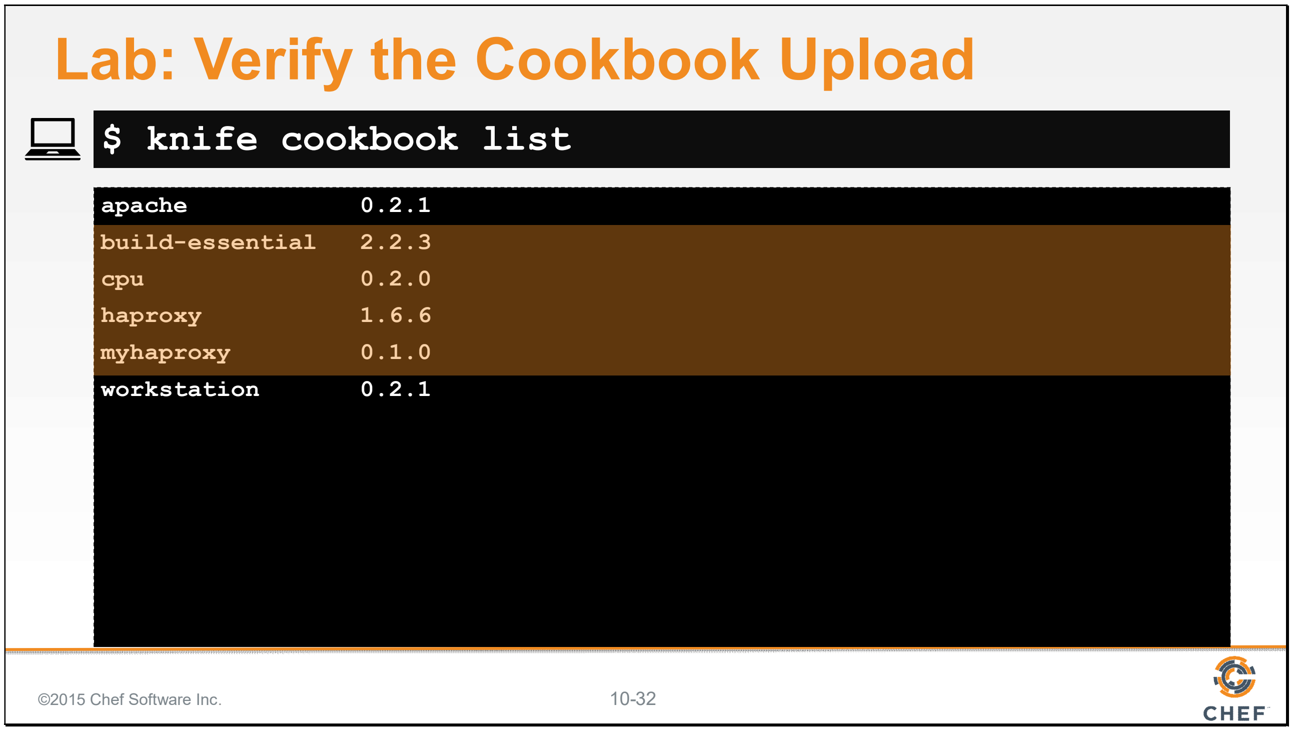
Lab: Upload the Cookbook

```
$ berks upload
```

```
Uploaded build-essential (2.2.3) to:
'https://api.opscode.com:443/organizations/steveessentials2'
Uploaded cpu (0.2.0) to: 'https://api.opscode.com:443/organizations/steveessentials2'
Uploaded haproxy (1.6.6) to: 'https://api.opscode.com:443/organizations/steveessentials2'
Uploaded myhaproxy (0.1.0) to: 'https://api.opscode.com:443/organizations/steveessentials2'
```

After installing all the necessary dependent cookbooks, we used 'berks upload' to send the cookbook and all its dependencies to the Chef Server. This is again an easier method to manage dependencies instead of manually identifying the dependencies and then uploading each single cookbook at a time.

Slide 32

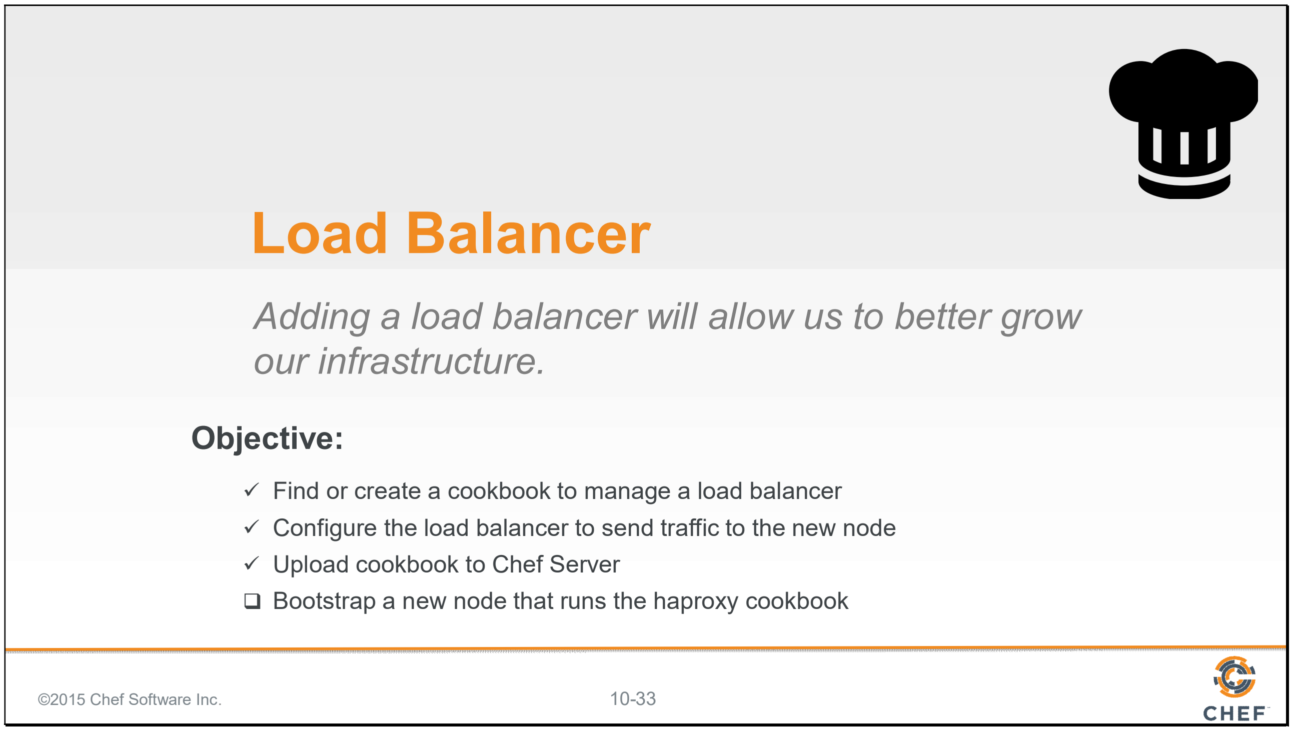


Lab: Verify the Cookbook Upload

```
$ 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
```

When that is complete you can verify that you've uploaded your cookbook and all of its dependencies.



Load Balancer

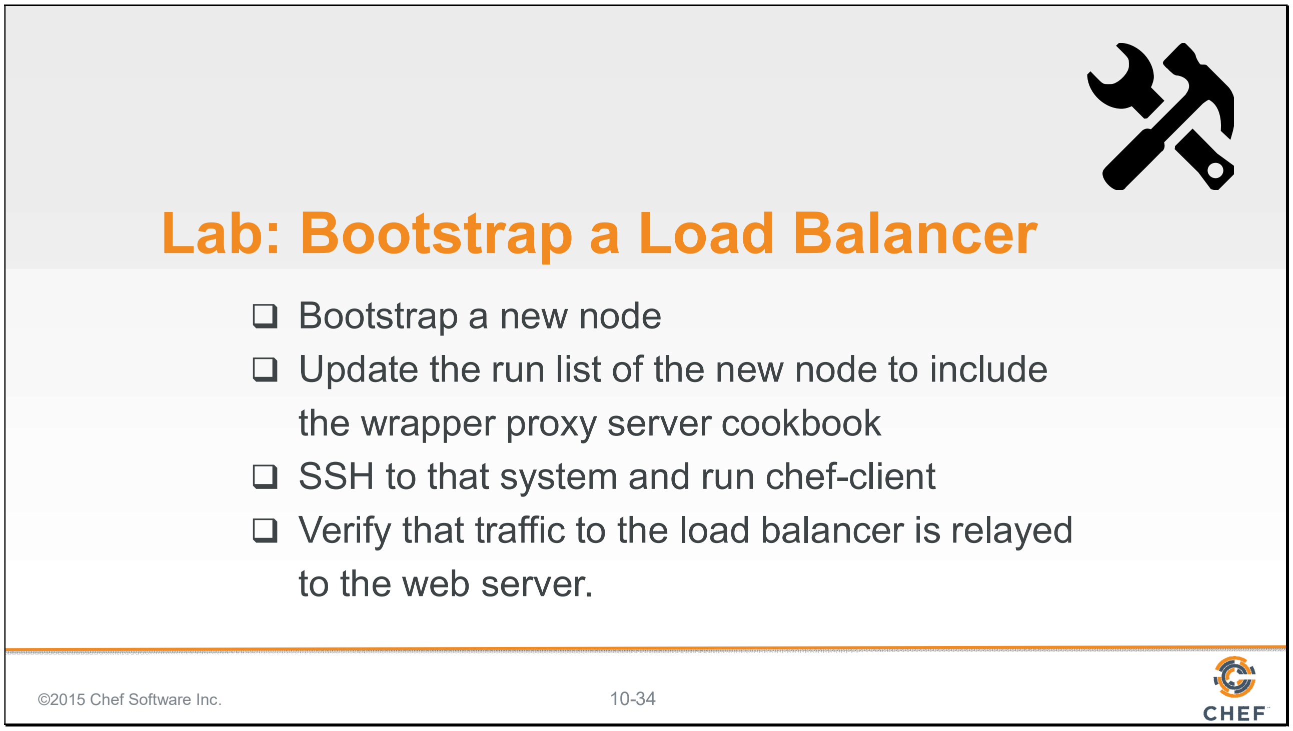
Adding a load balancer will allow us to better grow our infrastructure.

Objective:

- ✓ Find or create a cookbook to manage a load balancer
- ✓ Configure the load balancer to send traffic to the new node
- ✓ Upload cookbook to Chef Server
- ❑ Bootstrap a new node that runs the haproxy cookbook

The myhaproxy cookbook's default recipe is ready to be assigned to a run list of a node. So we'll need another node. The new load balancer node.

Slide 34

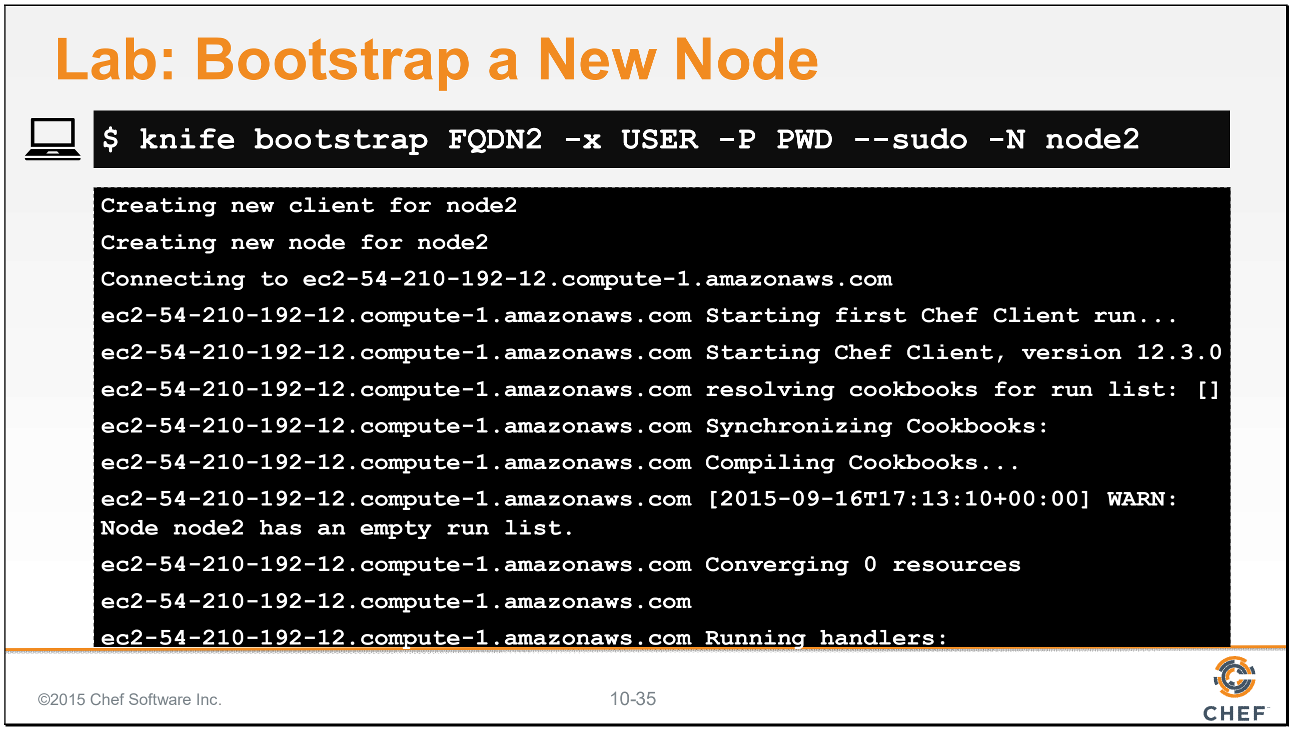


Lab: Bootstrap a Load Balancer

- ❑ Bootstrap a new node
- ❑ Update the run list of the new node to include the wrapper proxy server cookbook
- ❑ SSH to that system and run chef-client
- ❑ Verify that traffic to the load balancer is relayed to the web server.

Bootstrap this node the same as you did before but this time define the run list to converge the myhaproxy's default recipe. After setting that value, SSH into that node with the provided user name and password. Then run 'sudo chef-client' to apply the recipes defined in this node's run list. Then verify that your new node's default website is properly redirecting traffic to the original web node you previously set up.

Slide 35



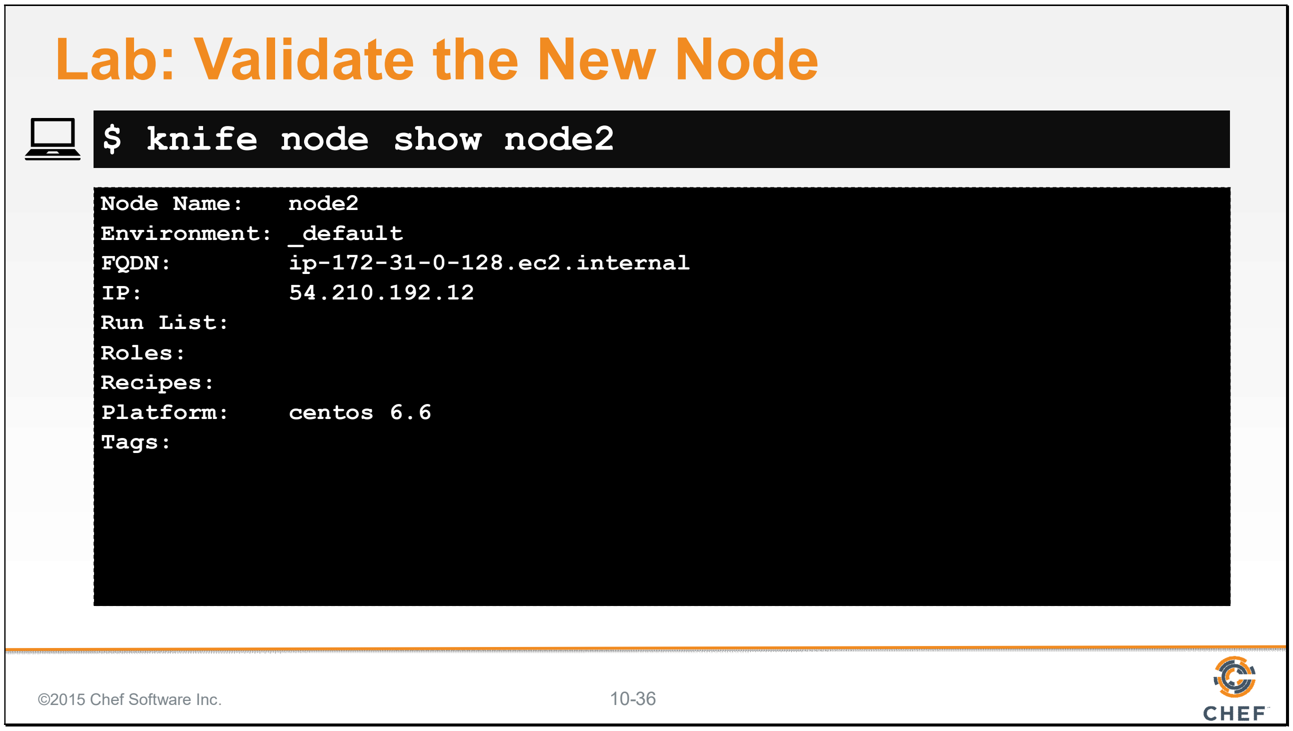
Lab: Bootstrap a New Node

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

```
Creating new client for node2
Creating new node for node2
Connecting to ec2-54-210-192-12.compute-1.amazonaws.com
ec2-54-210-192-12.compute-1.amazonaws.com Starting first Chef Client run...
ec2-54-210-192-12.compute-1.amazonaws.com Starting Chef Client, version 12.3.0
ec2-54-210-192-12.compute-1.amazonaws.com resolving cookbooks for run list: []
ec2-54-210-192-12.compute-1.amazonaws.com Synchronizing Cookbooks:
ec2-54-210-192-12.compute-1.amazonaws.com Compiling Cookbooks...
ec2-54-210-192-12.compute-1.amazonaws.com [2015-09-16T17:13:10+00:00] WARN:
Node node2 has an empty run list.
ec2-54-210-192-12.compute-1.amazonaws.com Converging 0 resources
ec2-54-210-192-12.compute-1.amazonaws.com
ec2-54-210-192-12.compute-1.amazonaws.com Running handlers:
```

First you bootstrap a new node named node2.

Slide 36

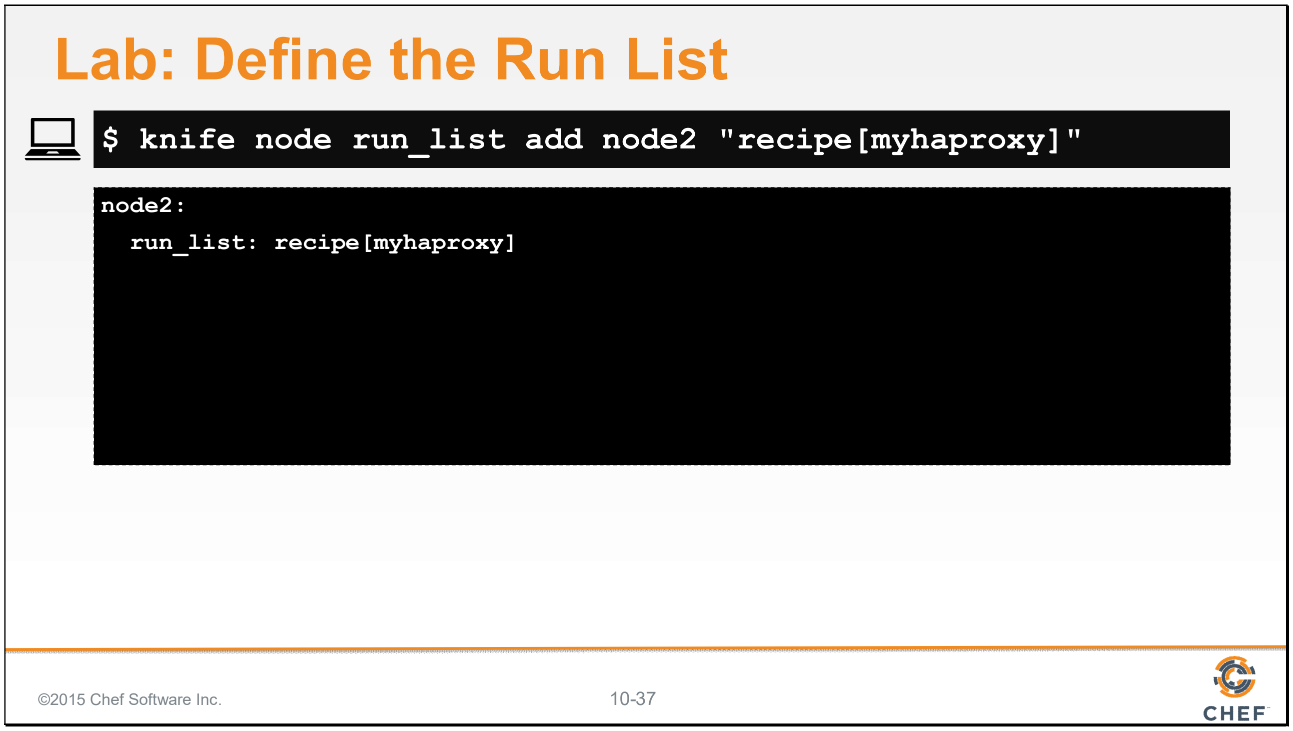


Lab: Validate the New Node

```
$ knife node show node2
```

```
Node Name:    node2
Environment:  _default
FQDN:         ip-172-31-0-128.ec2.internal
IP:           54.210.192.12
Run List:
Roles:
Recipes:
Platform:     centos 6.6
Tags:
```

After the node is bootstrapped, validate that it was added correctly to the organization.



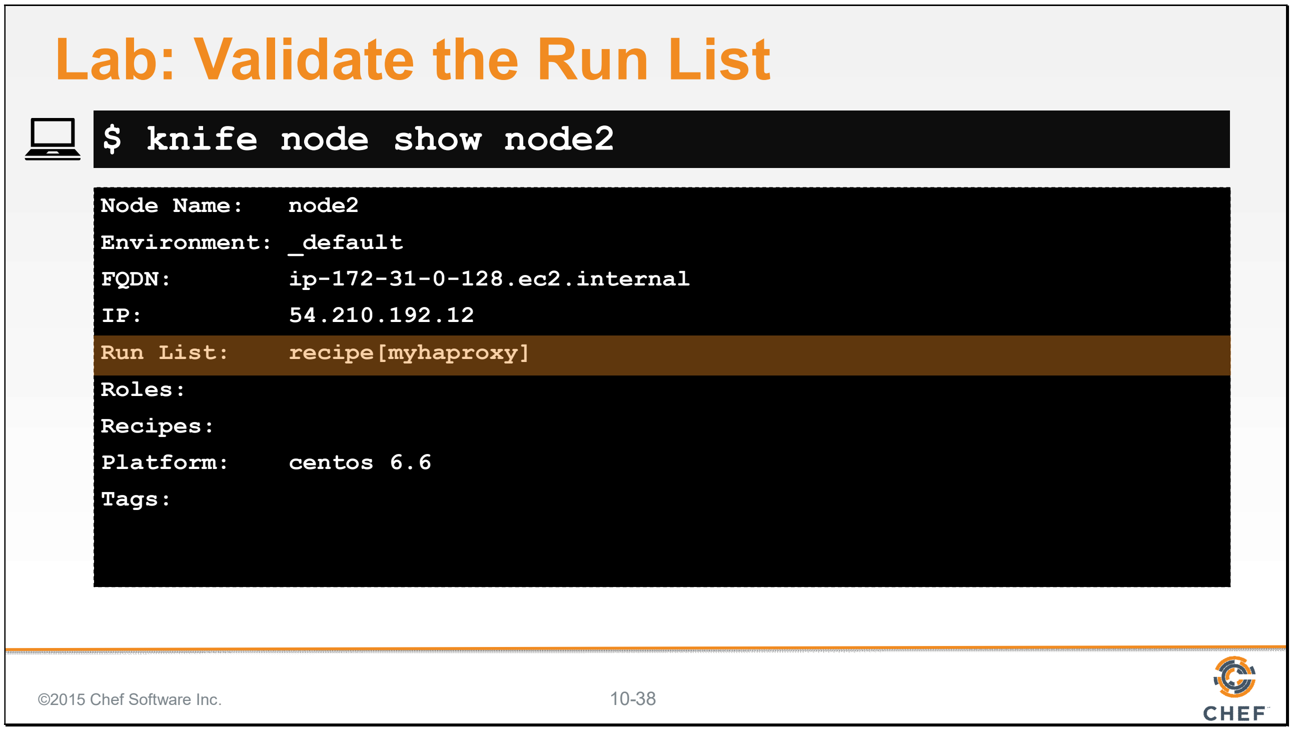
Lab: Define the Run List

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

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

Define an initial run list for that node to converge the default recipe of the myhaproxy cookbook.

Slide 38

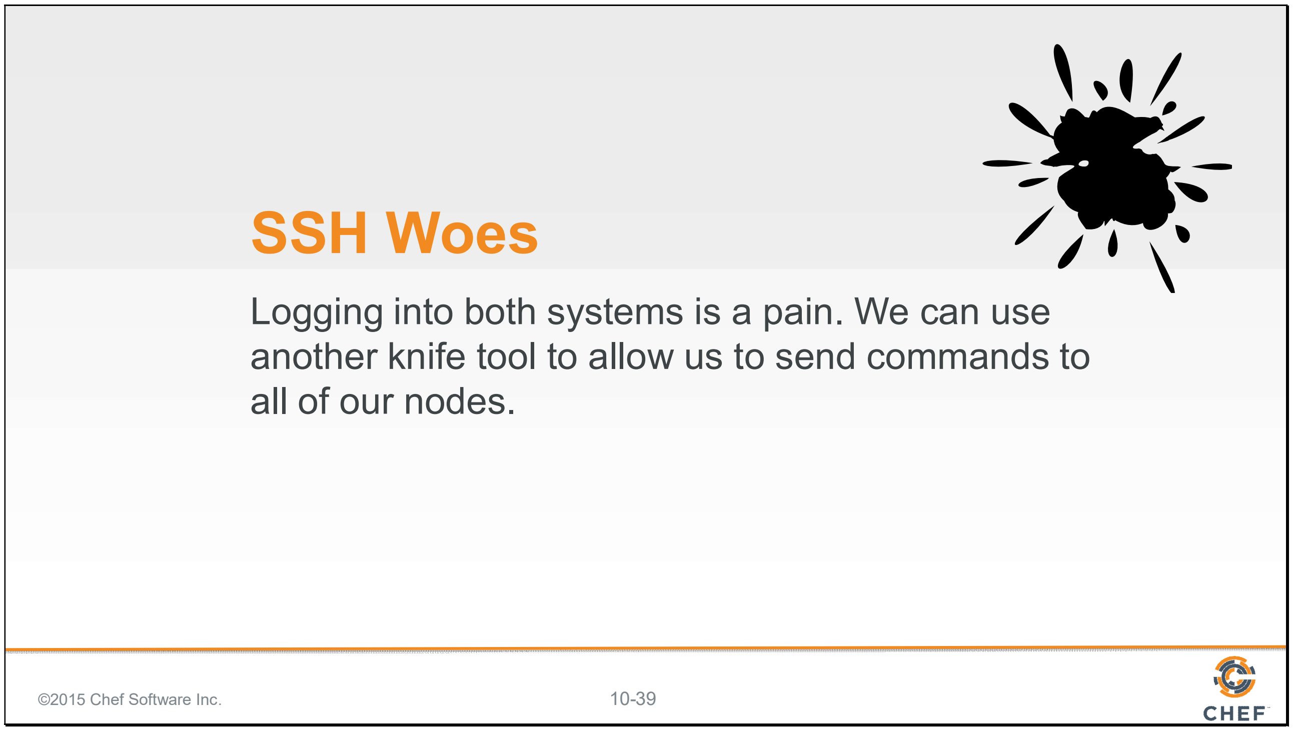


Lab: Validate the Run List

```
$ knife node show node2
```

```
Node Name:   node2
Environment: _default
FQDN:        ip-172-31-0-128.ec2.internal
IP:          54.210.192.12
Run List:    recipe[myhaproxy]
Roles:
Recipes:
Platform:    centos 6.6
Tags:
```

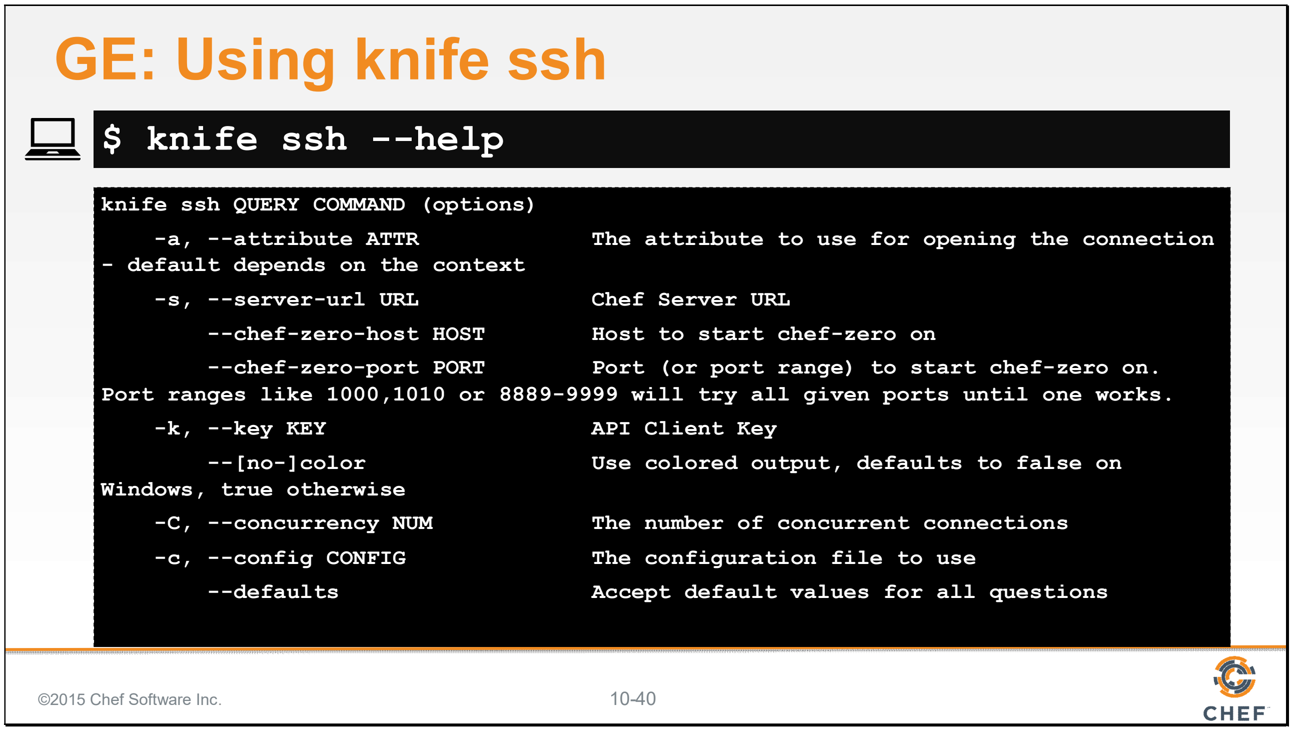
Ensure the run list has been set correctly for node2.



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.

We asked you to login to that remote node and run 'sudo chef-client' to apply the new run list defined for that node. This does in fact work but considering that we may need to execute this command for this node and many future nodes, it seems like a lot of windows and commands that we would need to execute.



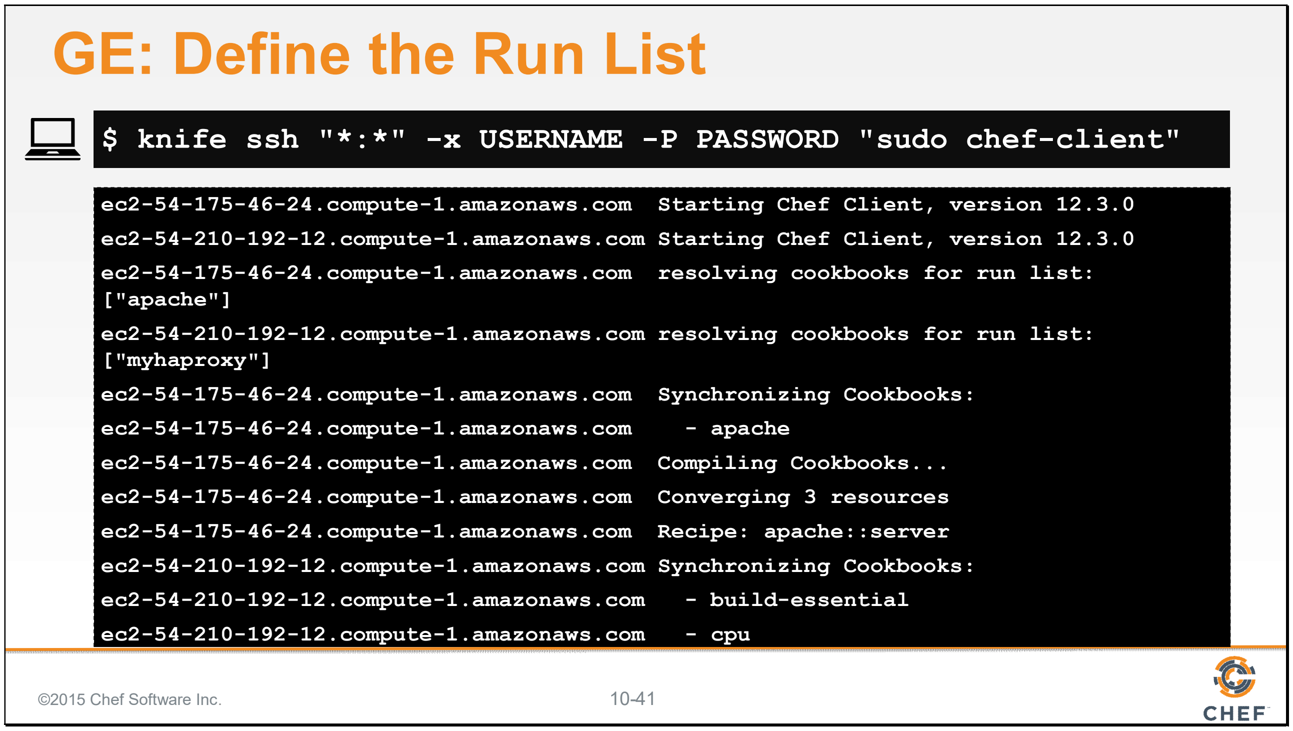
GE: Using knife ssh

```
$ 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
```

To make our lives easier, the 'knife' command provides a subcommand named 'ssh' that allows us to execute a command across multiple nodes that match a specified search query.

Slide 41



GE: Define the Run List

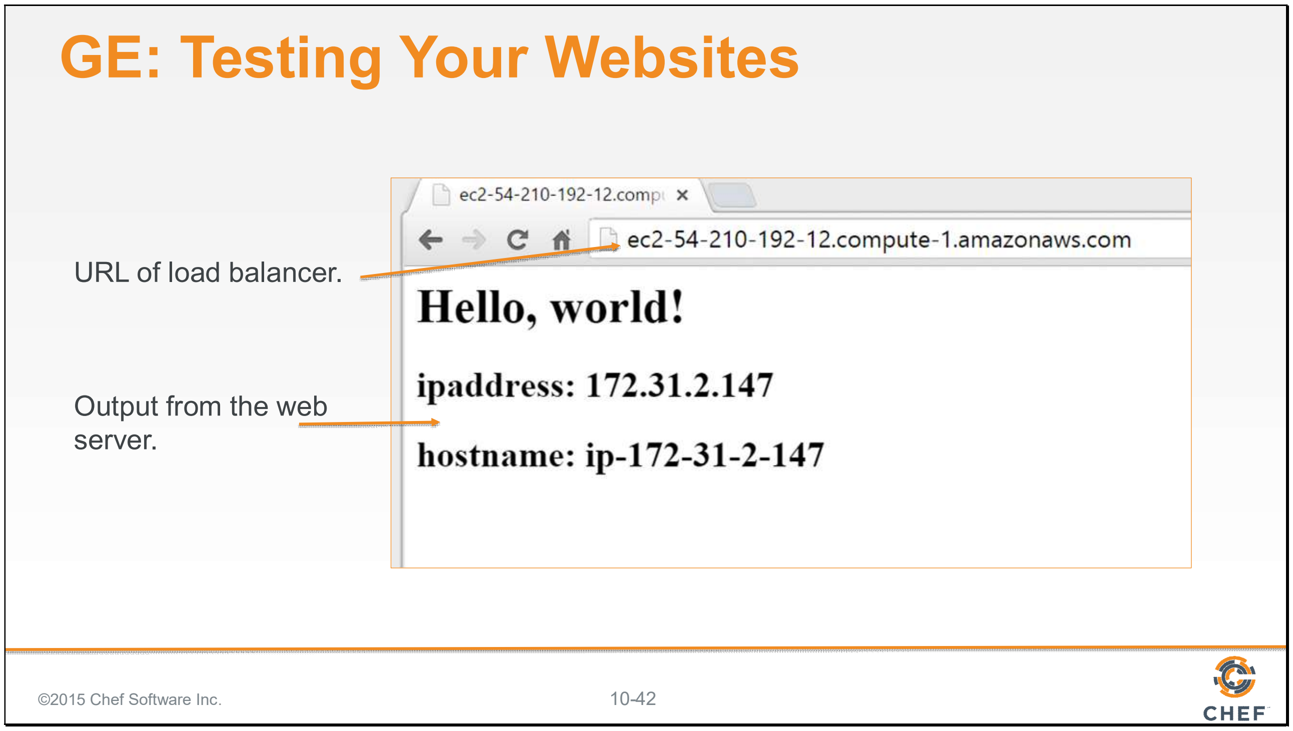
```
$ knife ssh "*:*" -x USERNAME -P PASSWORD "sudo chef-client"
```

```
ec2-54-175-46-24.compute-1.amazonaws.com  Starting Chef Client, version 12.3.0
ec2-54-210-192-12.compute-1.amazonaws.com Starting Chef Client, version 12.3.0
ec2-54-175-46-24.compute-1.amazonaws.com  resolving cookbooks for run list:
["apache"]
ec2-54-210-192-12.compute-1.amazonaws.com resolving cookbooks for run list:
["myhaproxy"]
ec2-54-175-46-24.compute-1.amazonaws.com  Synchronizing Cookbooks:
ec2-54-175-46-24.compute-1.amazonaws.com    - apache
ec2-54-175-46-24.compute-1.amazonaws.com  Compiling Cookbooks...
ec2-54-175-46-24.compute-1.amazonaws.com  Converging 3 resources
ec2-54-175-46-24.compute-1.amazonaws.com  Recipe: apache::server
ec2-54-210-192-12.compute-1.amazonaws.com Synchronizing Cookbooks:
ec2-54-210-192-12.compute-1.amazonaws.com   - build-essential
ec2-54-210-192-12.compute-1.amazonaws.com   - cpu
```

There are a lot of options for defining the search criteria that we will continue to explore. The most important criteria in this instance is star-colon-star. This means that we want to issue a command to all nodes.

So if you want to execute a "sudo chef-client" run for all of your nodes, you should write out this command. You would need to provide the user name to log into the system, the password for that system, and then finally the command to execute. In this way, you could easily ask your nodes to update from your current workstation as long as they all have the same login credentials. For more security, you should likely use SSH keys and forego specifying a username and password

Slide 42



GE: Testing Your Websites

URL of load balancer.

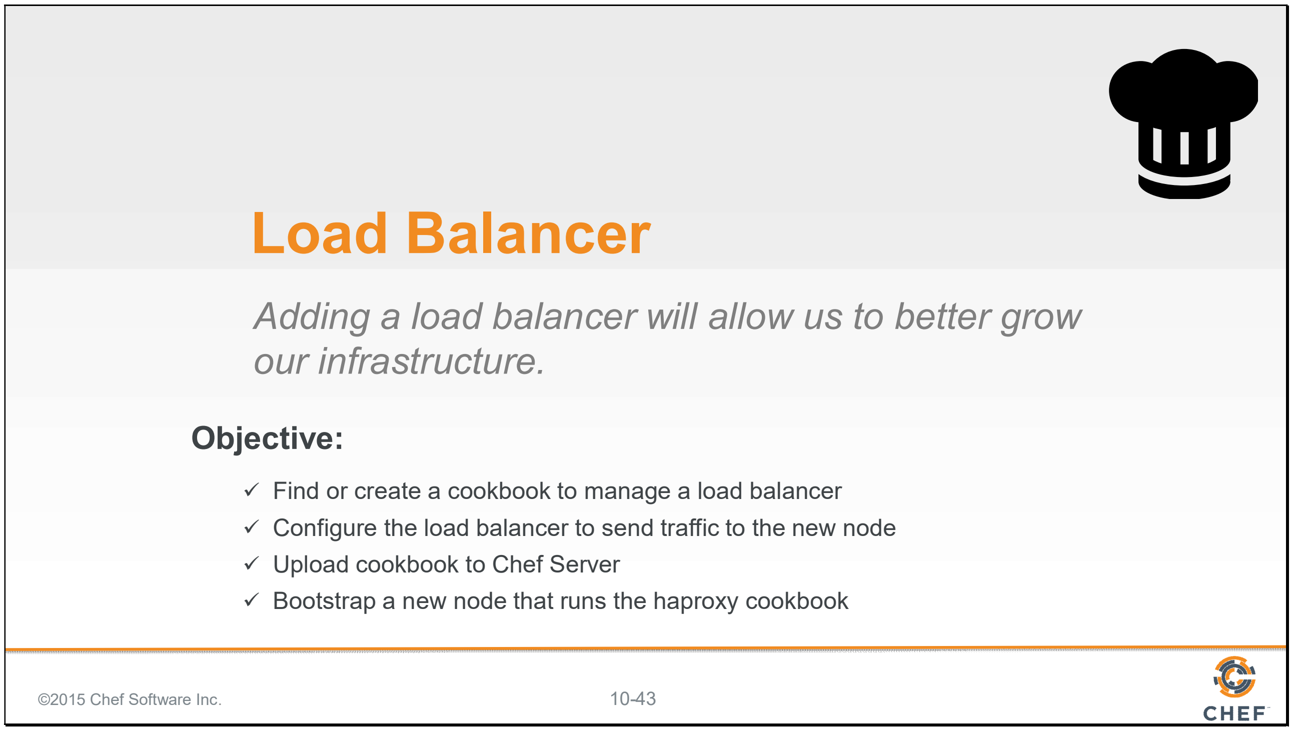
Output from the web server.

ec2-54-210-192-12.compute-1.amazonaws.com

```
Hello, world!

ipaddress: 172.31.2.147

hostname: ip-172-31-2-147
```

Point a web browser to the URL or public IP address of your load balancer. It should display the web page of the web server node that the load balancer is configured to serve.

Slide 43



Load Balancer

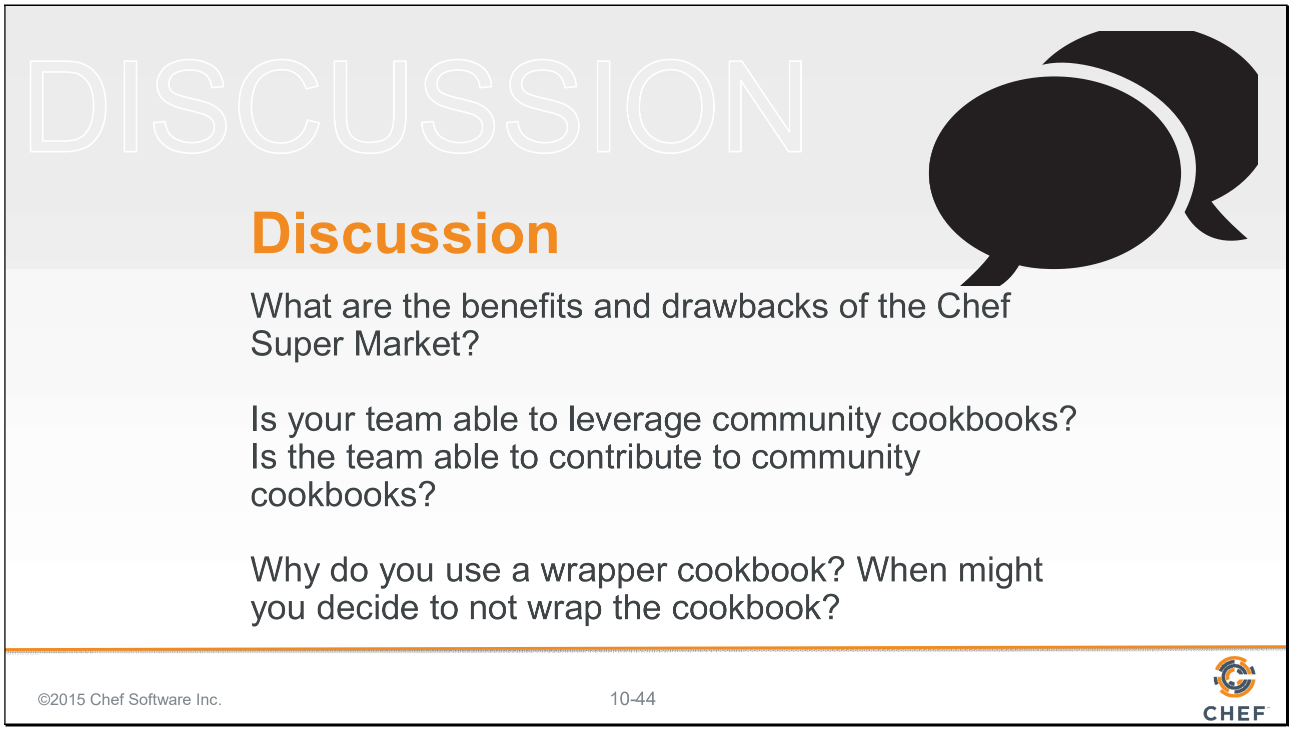
Adding a load balancer will allow us to better grow our infrastructure.

Objective:

- ✓ Find or create a cookbook to manage a load balancer
- ✓ Configure the load balancer to send traffic to the new node
- ✓ Upload cookbook to Chef Server
- ✓ Bootstrap a new node that runs the haproxy cookbook

With your node running the myhaproxy's cookbook's default recipe--relaying traffic to your first node running the apache cookbook's default recipe--you have moved closer to creating the original topology we set out to define today.

Slide 44



DISCUSSION

Discussion

What are the benefits and drawbacks of the Chef Super Market?

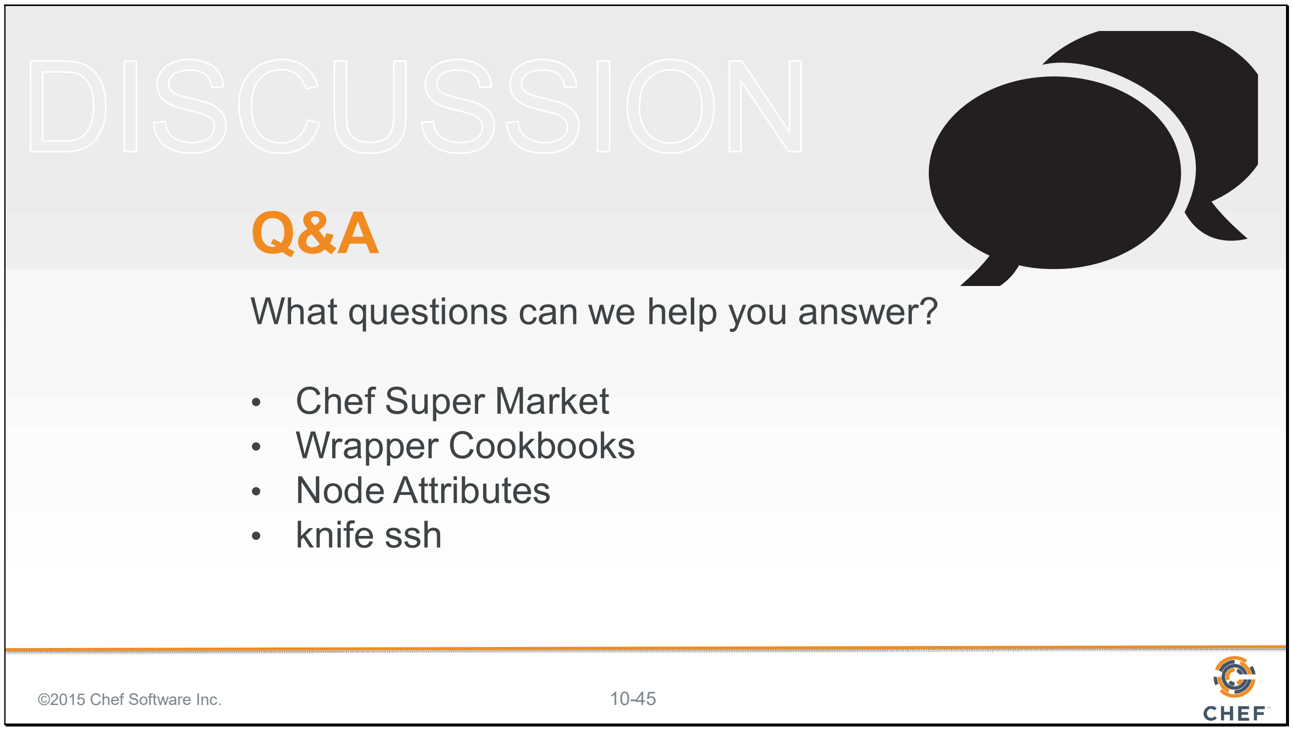
Is your team able to leverage community cookbooks? Is the team able to contribute to community cookbooks?

Why do you use a wrapper cookbook? When might you decide to not wrap the cookbook?

Answer these questions.

With your answers, turn to another person and alternate asking each other asking these questions and sharing your answers.

Slide 45


DISCUSSION

Q&A

What questions can we help you answer?

- Chef Super Market
- Wrapper Cookbooks
- Node Attributes
- knife ssh

CHEF

What questions can we help you answer?

In general or about specifically about Chef Super Market, wrapper cookbooks, node attributes, the 'knife ssh' command.

Slide 46

CHEF™