4: chef-client
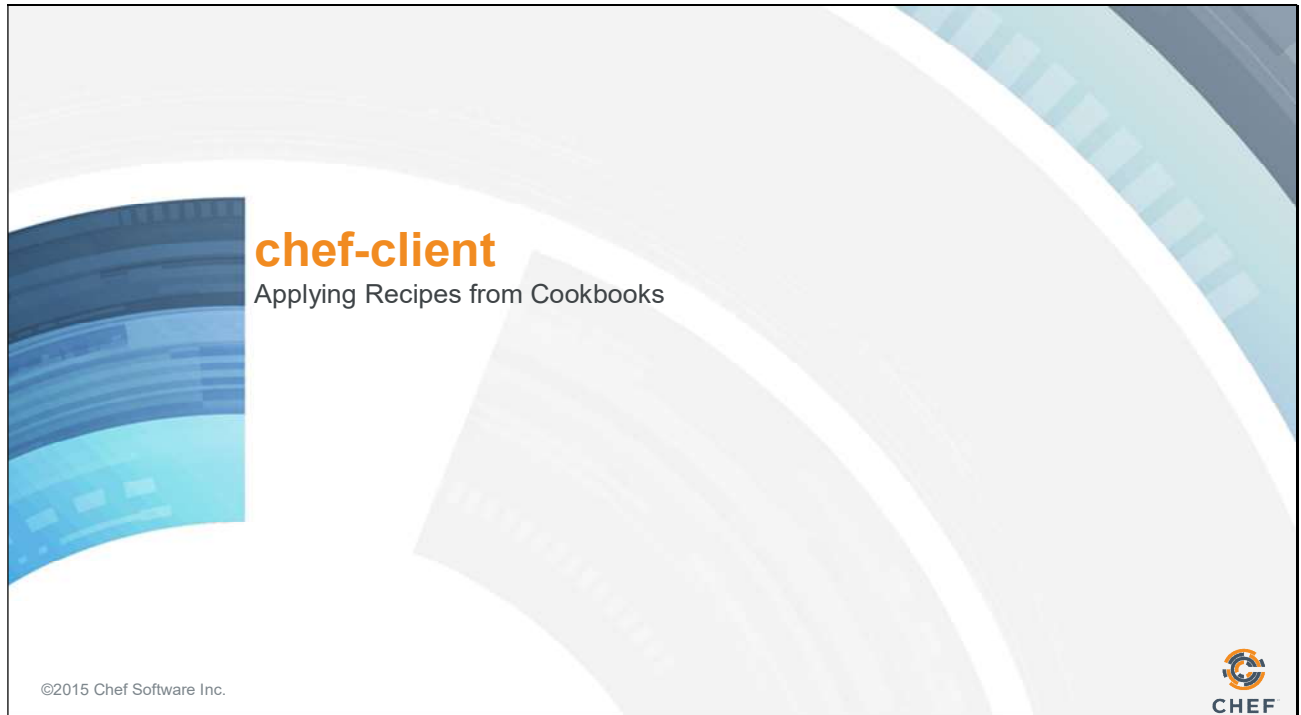

chef-client
Applying Recipes from Cookbooks

CHEF

Slide 2

Objectives

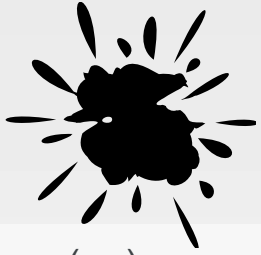After completing this module, you should be able to use chef-client to:

➢ Locally apply a cookbook's recipe with chef-client.

➢ Locally apply multiple cookbooks' recipes with chef-client.

➢ Include a recipe from within another recipe.

CHEF

In this module you will learn how to use the 'chef-client' command to apply recipes, and include a recipe within another recipe.

Slide 3

CONCEPT

chef-apply

chef-apply is a great tool for applying resources (-e) and for individual recipes but it doesn't know how to apply a cookbook.

CHEF

'chef-apply' is valuable tool for exploring resources within recipes without having to wrestle with all the folders and files associated with cookbooks. For the remainder of the modules we will not return to using 'chef-apply'. In the future you will most likely be using 'chef-client'. You may return to 'chef-apply' in your adventures when you find yourself wanting to test out an idea for a new recipe on a new platform or platform version. The speed of the tool is valuable.
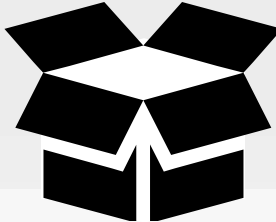
Slide 4

CONCEPT

chef-client

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.

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

CHEF

In the ChefDK, we package another tool that is called 'chef-client'.

'chef-client' is a command-line application that can be used to apply a recipe or multiple recipes. It also has the ability to communicate with a Chef server – a concept we will talk about in another section. For now think of the Chef Server as a central, artifact repository where we will later store our cookbooks.

Slide 5

Here is an example of using 'chef-client' to locally apply a run list of recipes. In this case we are applying one recipe and that is the setup recipe within our workstation cookbook.

Slide 6

Demo: Using 'chef-client' to Locally Apply Recipes

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

Applying the following recipes locally:

The 'server' recipe from the 'apache' cookbook

CHEF

Here is an example of using 'chef-client' to locally apply the server recipe within our apache cookbook.

Slide 7

Demo: Using 'chef-client' to Locally Apply Recipes

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

Applying the following recipes locally:

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

CHEF

Here is an example of using 'chef-client' to locally apply two recipes -- the setup recipe from the workstation cookbook and the server recipe within our apache cookbook.

Slide 8

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 it work in a
local mode.

CHEF

Applying recipes with 'chef-client' is different than 'chef-apply' and that is because chef-client's default behavior is to communicate with a Chef server. So we use the '--local-mode' flag to ask 'chef-client' to look for the cookbooks locally.

Slide 9

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].

CHEF

When we apply a recipe with 'chef-client', we define a run list. This is an ordered list of recipes that we want to apply to the system. When you define a recipe from a cookbook on the run list, there is a particular convention:

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

COOKBOOK means the name of the Cookbook.

RECIPE means the name of the Recipe without the Ruby file extension.

Slide 10

Group Exercise: Return Home First

```
$ cd ~
```

CHEF

Before you start applying cookbooks through 'chef-client', make sure you are in your home directory.

Slide 11

GE: Apply the 'apache::server' Recipe Locally

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

[2015-09-15T14:52:45+00:00] WARN: No config file found or specified on command
line, using command line options.
[2015-09-15T14:52:45+00:00] WARN: No cookbooks directory found at or above
current directory.  Assuming /home/chef.
Starting Chef Client, version 12.3.0
resolving cookbooks for run list: ["apache::server"]


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

CHEF

Try applying our server recipe from the apache cookbook using `chef-client` in local mode.

Upon execution you unfortunately are presented with an error.

When executed we find that `chef-client` has an additional requirement. `chef-client` expects our cookbooks to be maintained in a directory named 'cookbooks'.

That seems simple enough to accommodate and a good way to start organizing the cookbooks that we are creating.

Slide 12

GE: Create Cookbooks Dir and Move the Cookbooks

```
$ mkdir cookbooks
$ mv workstation cookbooks
$ mv apache cookbooks
```

CHEF

Make a directory named 'cookbooks'. Then move the workstation cookbook and apache cookbook into the cookbooks directory.

Slide 13

GE: Apply the Cookbook Recipe Locally

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

[2015-09-15T14:54:45+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::server"]
Synchronizing Cookbooks:
  - apache
Compiling Cookbooks...
Converging 4 resources
Recipe: apache::server
  * yum_package[httpd] action install (up to date)
  * file[/var/www/html/index.html] action create (up to date)
  * service[httpd] action enable (up to date)
```

©2015 Chef Software Inc.   4-13   CHEF

Let's try that again--this time with all of our cookbooks in the cookbooks directory like `chef-client` expects.

Try applying the apache cookbook's recipe named server.

Slide 14

GE: Apply the Cookbook Recipe Locally

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

```
[2015-09-15T15:15:26+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: ["workstation::setup"]
Synchronizing Cookbooks:
  - workstation
Compiling Cookbooks...
Converging 6 resources
Recipe: workstation::setup
  * yum_package[nano] action install (up to date)
  * yum_package[vim] action install (up to date)
  * yum_package[emacs] action install (up to date)
```

CHEF

Try applying the workstation cookbook's recipe named 'setup'.

Slide 15

GE: Apply Both Recipes Locally

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

```
[2015-09-15T15:17:27+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::server","workstation::setup"]
Synchronizing Cookbooks:
  - apache
  - workstation
Compiling Cookbooks...

Running handlers:
[2015-09-15T15:17:30+00:00] ERROR: Running exception handlers
```

©2015 Chef Software Inc.   4-15

Try applying both recipes from both cookbooks again at one time.

Slide 16

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.

©2015 Chef Software Inc.   4-16

Actually, we didn't tell you everything about specifying the run list for the `chef-client` command.

When defining a recipe in the run list you may omit the name of the recipe, and only use the cookbook name, when that recipe's name is 'default'.

Similar to how resources have default actions and default attributes Chef uses the concept of providing sane defaults. This makes our faster when we understand the concepts.

A cookbook doesn't have to have a default recipe but most every cookbook has one. It's called default because when you think of a cookbook it is the recipe that defines the most common configuration policy.

When you think about the two cookbooks that we created -- the apache cookbook with the server recipe and the workstation cookbook with the setup recipe -- it seems like those recipes would be good default recipes for their respective cookbooks.

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.

https://docs.chef.io/recipes.html#include-recipes

4-17

CHEF

A simple solution would be to rename the setup recipe to the default recipe. However, a better practice would instead leave our recipes as they are and have the default recipe include the recipes with a method called `include_recipe`

This allows us to maintain all the current policies within its own recipe file and that way we can more easily switch our cookbooks default behavior, which can be useful when new requirements surface.

Slide 18

Demo: Including a Recipe

```
include_recipe 'workstation::setup'
```

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

CHEF

In this example we are including the 'workstation' cookbook's 'setup' recipe.

Demo: Including a Recipe

```
include_recipe 'apache::server'
```

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

CHEF

In this example, we are including the 'apache' cookbook's 'server' recipe.

Slide 20

GE: 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'
```

We are interested in having the default recipe for our workstation cookbook run the contents of the setup recipe.

Within the default recipe, define the `include_recipe` method and provide one parameter, which is the name of our recipe as it appears within a run list: cookbook_name::recipe_name.

GE: Apply the Cookbook's Default Recipe

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

```
 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: ["workstation"]
Synchronizing Cookbooks:
  - workstation
Compiling Cookbooks...
Converging 0 resources

Running handlers:
Running handlers complete
Chef Client finished, 0/0 resources updated in 3.300489827 seconds
```
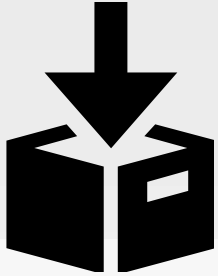
©2015 Chef Software Inc.   4-21

CHEF

Use 'chef-client' to locally apply the cookbook named workstation. This will load your workstation cookbook's default recipe, which in turn loads the workstation cookbook's setup recipe.

COMMIT

GE: Commit Your Work

```
$ cd workstation
$ git add .
$ git commit -m "Default recipe includes the setup
recipe"
```

CHEF

With everything working it is time to commit the latest changes.

Slide 23

Lab: 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]"

❑ Commit the changes with version control

In this lab you will update the apache cookbook's default recipe to include the apache cookbook's recipe named server.

Slide 24

We are interested in having the default recipe for our apache cookbook run the contents of the server recipe.

Within the default recipe, define the `include_recipe` method and provide one parameter, which is the name of our recipe as it appears within a run list: cookbook_name::recipe_name.

Lab: Applying the apache Default Recipe

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

```
[2015-09-15T15:23:18+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 0 resources

Running handlers:
Running handlers complete
Chef Client finished, 0/0 resources updated in 3.310768509 seconds
```
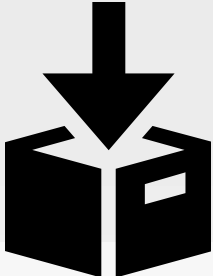
CHEF

Use 'chef-client' to locally apply the cookbook named apache. This will load your apache cookbook's default recipe, which in turn loads the apache cookbook's server recipe.

COMMIT

Lab: Commit Your Work

```
$ cd apache
$ git add .
$ git commit -m "Default recipe includes the server
recipe"
```

CHEF

With everything working it is time to commit the latest changes.

Slide 27

Discussion

Why would you want to apply more than one recipe at a time?

What are the benefits and drawbacks of using "include_recipe" within a recipe?

Do default values make it easier or harder to learn?

4-27

CHEF

Answer these questions.

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

Slide 28

Q&A

What questions can we help you answer?

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

4-28

CHEF

What questions can we help you answer?

Generally or specifically about chef-client, local mode, run lists, and include_recipe.

Slide 29

©2015 Chef Software Inc.