6: Details About the System

Details About the System
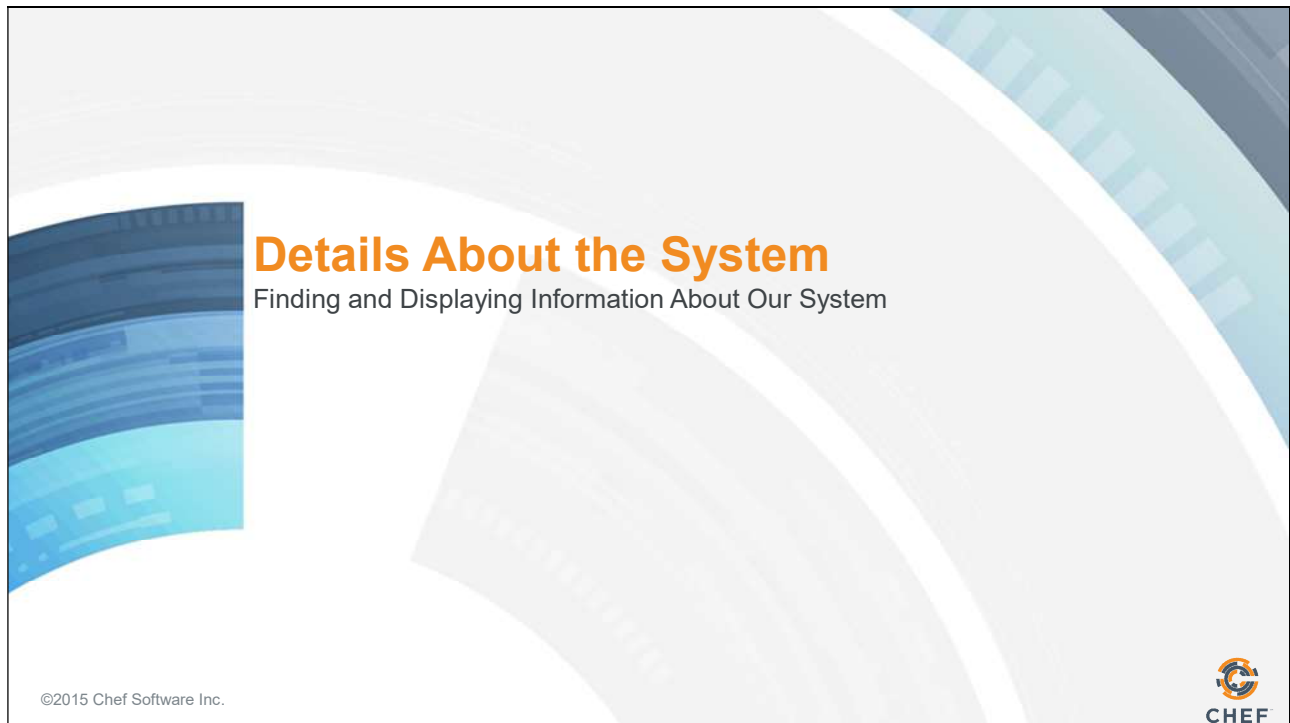Finding and Displaying Information About Our System

©2015 Chef Software Inc.

Slide 2

Objectives

After completing this module, you should be able to

- ➢ Capture details about a system
- ➢ Use the node object within a recipe
- ➢ Use Ruby's string interpolation
- ➢ Update the version of a cookbook

In this module you will learn how to capture details about a system, use the node object within a recipe, use Ruby's string interpolation, and update the version of a cookbook.

Have you ever had to manage a large number of servers that were almost identical?

How about a large number of identical servers except that each one had to have host-specific information in a configuration file?

The file needed to have the hostname or the IP address of the system.

Maybe you needed to allocate two-thirds of available system memory into HugePages for a database. Perhaps you needed to set your thread max to number of CPUs minus one.

The uniqueness of each system required you to define custom configuration files. Custom configurations that you need to manage by hand.

Details About the System

Displaying system details in the default web page definitely sounds useful.

Objective:

- ☐ Find out various details about the system
- ☐ Update the web page file contents, in the "iis-demo" cookbook, to include system details
- ☐ Use Test Kitchen to converge and verify the "iis-demo" cookbook
- ☐ Use chef-client to locally apply the "iis-demo" cookbook's default recipe

©2015 Chef Software Inc.   6- 4

When deploying our IIS server it would be nice if the test page displayed some details about the system. Together, let's walk through finding out these details and then updating the content attribute of the file resource to include this new content.

Slide 5

Some Useful System Data

o   IP Address
o   hostname
o   memory
o   CPU - MHz

Thinking about some of the scenarios that we mentioned at the start of the session makes us think that it would be useful to capture:

The IP address, hostname, memory, and CPU megahertz of our current system.

We'll walk through capturing that information using various system commands starting with the IP address.

Slide 6

GE: Finding the IP Address

```
$ ipconfig

Windows IP Configuration


Ethernet adapter Ethernet 2:

   Connection-specific DNS Suffix  . : ec2.internal
   Link-local IPv6 Address . . . . . : fe80::2da8:4ba7:45e2:e863%21
   IPv4 Address. . . . . . . . . . . : 172.31.21.21
   Subnet Mask . . . . . . . . . . . : 255.255.240.0
   Default Gateway . . . . . . . . . : 172.31.16.1
```

Running the `ipconfig` command will return to the IP Address of the system.

Slide 7

GE: Finding the Hostname

```
$ hostname
```

```
WIN-KRQSVD3RFM7
```

CHEF

Running the `hostname` command will return to the hostnane of the system.

Slide 8

Running the following command will return to the total physical memory of the system.

Slide 9

GE: Finding the CPU MHz

```
$ wmic cpu get name

Name
Intel(R) Xeon(R) CPU E5-2666 v3 @ 2.90GHz
```

CHEF

Running the following command will return to the cpu name which includes the clock speed of the CPU.

Details About the System

Displaying system details in the default web page definitely sounds useful.

Objective:

- ✓ Find out various details about the system
- ❑ Update the web page file contents, in the "iis-demo" cookbook, to include system details
- ❑ Use Test Kitchen to converge and verify the "iis-demo" cookbook
- ❑ Use chef-client to locally apply the "iis-demo" cookbook's default recipe

Now that we have those values about the system it is time to update the server recipe to include these details in the test page that we deploy.

Slide 11

GE: Adding the CPU

`~\cookbooks\iis-demo\recipe\server.rb`

```
# ... POWERSHELL_SCRIPT RESOURCE ...


file 'c:\inetpub\wwwroot\Default.htm' do
  content '<h1>Hello, world!</h1>
<h2>ipaddress: 172.31.21.21</h2>
<h2>hostname: WIN-KRQSVD3RFM7</h2>
<h2>total memory: 8052654080</h2>
<h2>CPU Mhz: 2.90GHz</h2>'
end


# ... SERVICE RESOURCE ...
```

Edit the server recipe and update the content attribute of the file resource to include some new HTML that includes the fields that we are interested in seeing in our test page.

Notice that the first line of the content attribute is no longer terminated with a single-quote. The single-quote is moved until the very end of the content after the element that contains the CPU Mhz.

Details About the Node

Displaying system details in the default web page definitely sounds useful.

Objective:

- ✓ Find out various details about the system
- ✓ Update the web page file contents, in the "iis-demo" cookbook, to include system details
- ❑ Use Test Kitchen to converge and verify the "iis-demo" cookbook
- ❑ Use chef-client to locally apply the "iis-demo" cookbook's default recipe

©2015 Chef Software Inc.   6-12

The recipe has been updated. Now it is time to test the changes that we have made. We have not written any new tests but our existing tests will ensure that the functionality we have come to depend on will not be broken.

Slide 13

Introducing a Change

By creating a change we have introduced risk.

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

Even though that seemed like a small change, it was a change. And with any changes there is a chance that we made a mistake or changed the behavior that was previously expected. If we decided to not run our tests we would be taking a risk when we attempted to apply it to our current system or if we were to give the cookbook to another individual.

This is an important practice to follow. After completing a change it is a good idea to execute the test suite.

Slide 14

GE: Change into Our Cookbook

```
$ cd ~\cookbooks\iis-demo
```

Remember, we are testing a specific cookbook with kitchen so we need to be within the directory of the cookbook. So change directory into the 'iis-demo' cookbook's directory.

Slide 15

Again we have not defined any new tests but the tests that we currently test that the recipe converges successfully and that the existing functionality is still present.

First run `kitchen converge` to apply the updated policy against the same test instance. Because we are not destroying the instance, when we converge it will act much faster as the resources will be updating the existing features, files, and services already in place.

Second run `kitchen verify` to ensure that the tests that we have defined previously will still pass.

Slide 16

Details About the System

Displaying system details in the default web page definitely sounds useful.

Objective:

- ✓ Find out various details about the system
- ✓ Update the web page file contents, in the "iis-demo" cookbook, to include system details
- ✓ Use Test Kitchen to converge and verify the "iis-demo" cookbook
- ❑ Use chef-client to locally apply the "iis-demo" cookbook's default recipe

---

When everything converges successfully and all the tests pass it is time to apply the cookbook locally to the system.

Slide 17

GE: Return Home and Apply iis-demo Cookbook

```
$ cd ~
$ chef-client --local-mode -r "recipe[iis-demo]"
```

```
[2015-12-23T22:45:48+00:00] WARN: No config file found or specified on command
line, using command line options.
Starting Chef Client, version 12.5.1
resolving cookbooks for run list: ["iis-demo"]
Synchronizing Cookbooks:
  - iis-demo (0.1.0)
Compiling Cookbooks...
Converging 3 resources
Recipe: iis-demo::server
  * powershell_script[Install IIS] action run
    - execute "C:\Windows\system32\WindowsPowerShell\v1.0\powershell.exe" -NoLogo
-NonInteractive -NoProfile -ExecutionPolicy Bypass -InputFormat None -File
```

CHEF

If everything looks good, then we want to use `chef-client`. `chef-client` is not run on a specific cookbook--it is a tool that allows us to apply recipes for multiple cookbooks that are stored within a cookbooks directory.

1. So we need to return home to the parent directory of all our cookbooks.

2. Then use `chef-client` to locally apply the run list defined as: the 'iis-demo' cookbook's default recipe.

Slide 18

GE: Verify the Default Page Returns the Details

```
$ curl localhost
```

```
StatusCode        : 200
StatusDescription : OK
Content           : <h1>Hello, world!</h1>
                    <h2>ipaddress: 172.31.21.21</h2>
                    <h2>hostname: WIN-KRQSVD3RFM7</h2>
                    <h2>total memory: 8052654080</h2>
                    <h2>CPU Mhz: 2.90GHz</h2>
RawContent        : HTTP/1.1 200 OK
                    Accept-Ranges: bytes
                    Content-Length: 150
```

6-18

CHEF

Verify that the Default page is returning the new updated content with the details about the system.

Details About the System

Displaying system details in the default web page definitely sounds useful.

Objective:

- ✓ Find out various details about the system
- ✓ Update the web page file contents, in the "iis-demo" cookbook, to include system details
- ✓ Use Test Kitchen to converge and verify the "iis-demo" cookbook
- ✓ Use chef-client to locally apply the "iis-demo" cookbook's default recipe

DISCUSSION

Capturing System Data

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

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

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

©2015 Chef Software Inc.   6-20

Now that we've defined these values, lets reflect:

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

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

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!

If you have worked with systems for a while, the general feeling is that hard-coding the values in our file resource's attribute probably is not sustainable because the results are tied specifically to this system at this moment in time.

Slide 22

So how can we capture this data in real-time?

Capturing the data in real-time on each system is definitely possible. One way would be to execute each of these commands, parse the results, and then insert the dynamic values within the file resource's content attribute.

We could also figure out a way to run system commands within our recipes.

Before we start down this path, we'd like to introduce you to Ohai.

Slide 23

Ohai is a tool that detects and captures attributes about our system. Attributes like the ones we spent our time capturing already.

Slide 24

GE: Running 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": {
```

Ohai is also a command-line application that is part of the Chef Development Kit (ChefDK).

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).

http://docs.chef.io/ohai.html

Ohai, the command-line application, will output all the system details represented in JavaScript Object Notation (JSON).

Slide 26

These values are available in our recipes because `chef-client` and `chef-apply` automatically execute Ohai. This information is stored within a variable we call 'the node object'

Slide 27

CONCEPT

The Node Object

The node object is a representation of our system.
It stores all the attributes found about the system.

http://docs.chef.io/nodes.html#attributes

©2015 Chef Software Inc.   6-27   CHEF

The node object is a representation of our system. It stores all these attributes found about the system. It is available within all the recipes that we write to assist us with solving the similar problems we outlined at the start.

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.

Let's look at using the node object to retrieve the ipaddress, hostname, total memory, and cpu megahertz.

Slide 28
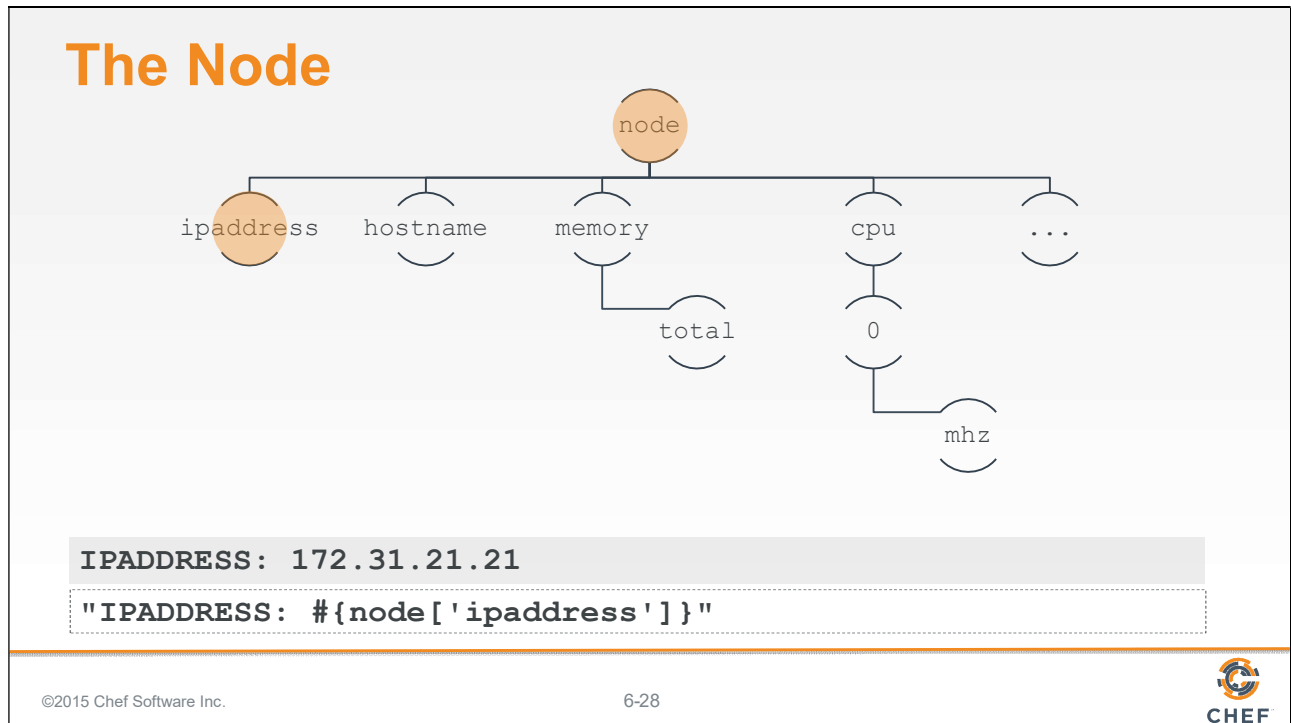
This is the visualization of the node attributes as a tree. That is done here to illustrate that the node maintains a tree of attributes that we can request from it.

The shaded text near the bottom of this slide is the hard-coded value we currently have in the file resource's content attribute.
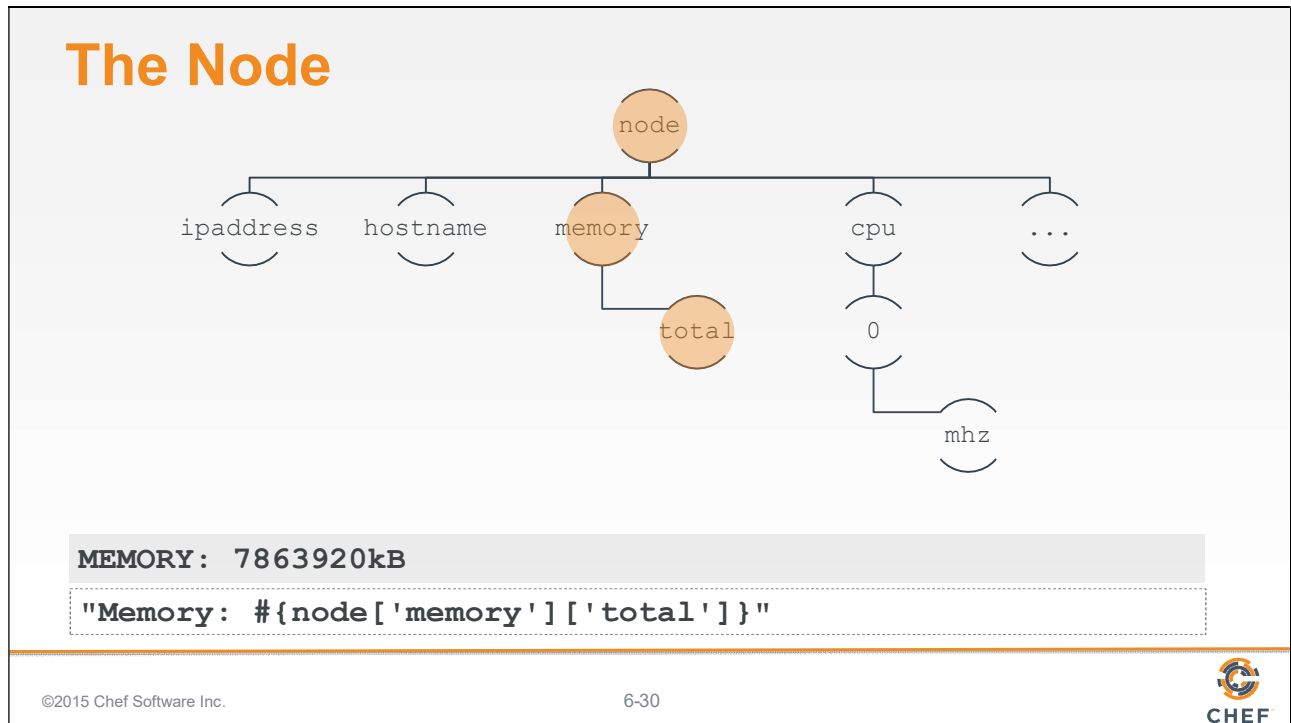
At the very bottom is an example of how we could use the node's dynamic value within a string instead of the hard-coded one.
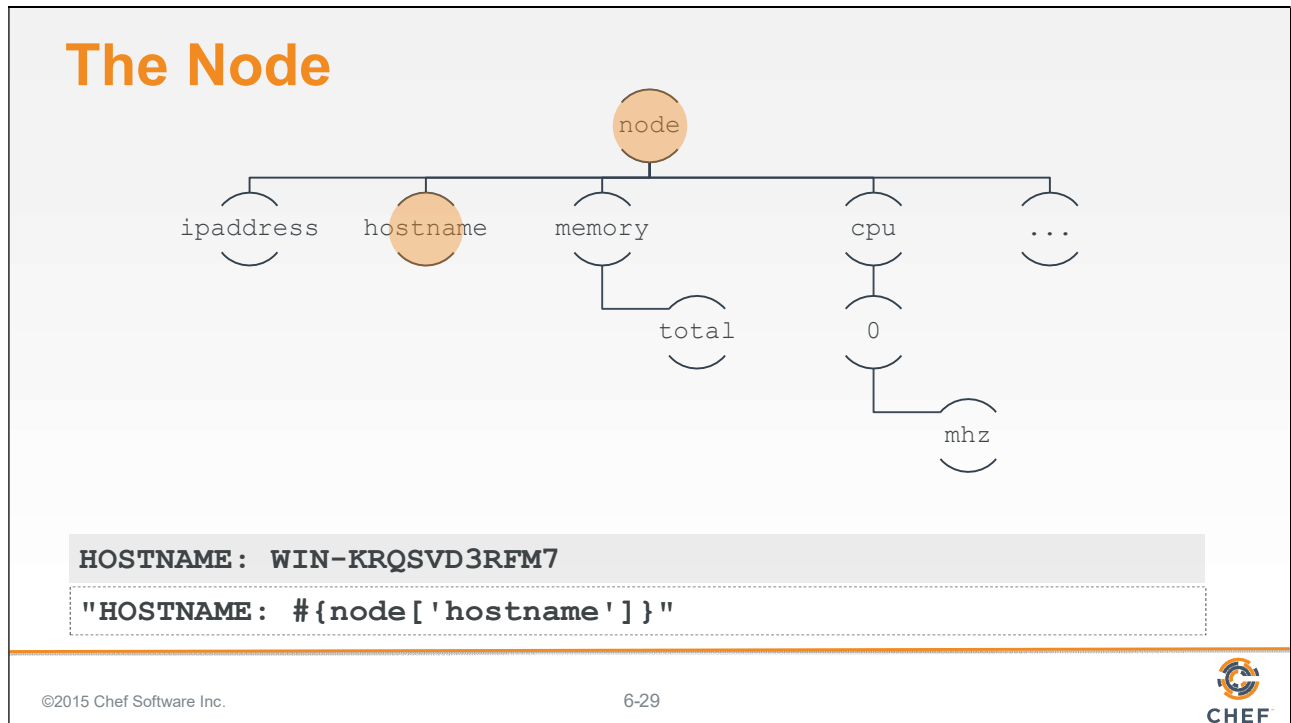
The node maintains a hostname attribute. This is how we retrieve and display it.

Slide 30

The node contains a top-level value memory which has a number of child elements. One of those child elements is the total amount of system memory.

Accessing the node information is different. We retrieve the first value 'memory', returning a subset of keys and values at that level, and then immediately select to return the total value.

Slide 31
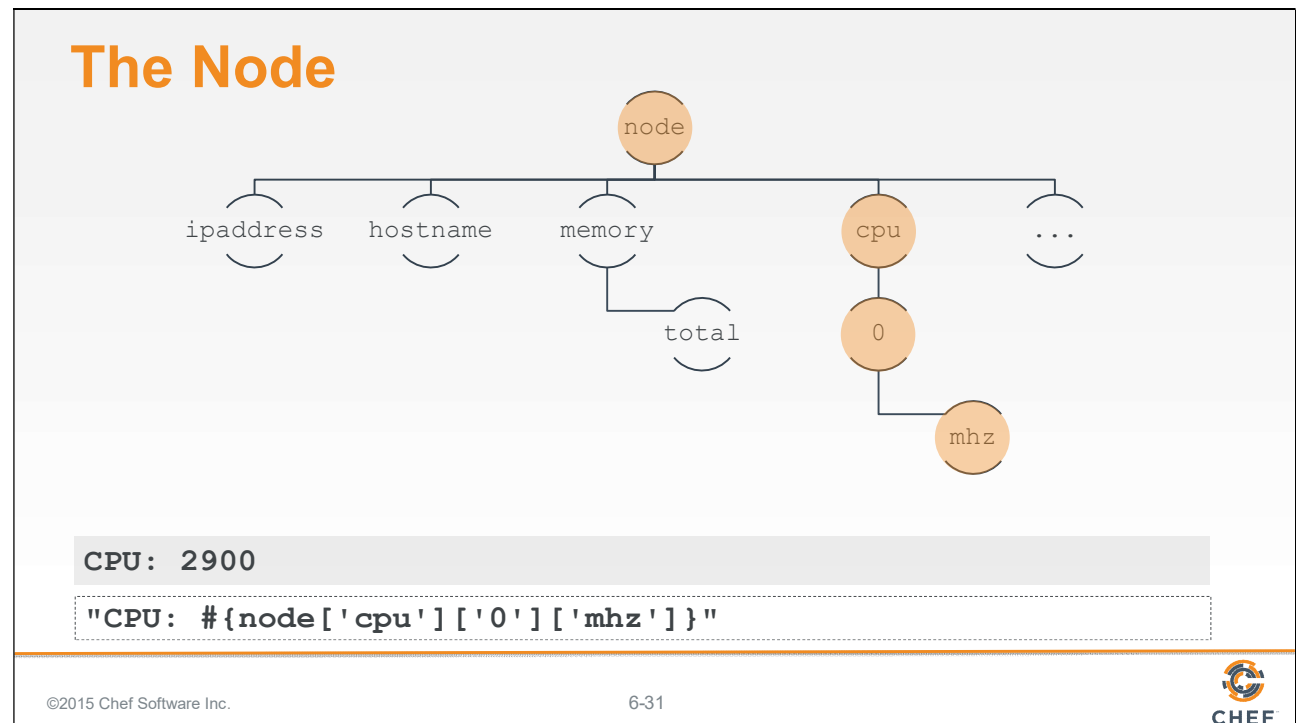
And finally, here we return the megahertz of the first CPU.

CONCEPT

String Interpolation

```
I have 4 apples
```

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

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

In all of the previous examples we demonstrated retrieving the values and displaying them within a string using a ruby language convention called string interpolation.

Slide 33

String interpolation is only possible with strings that start and end with double-quotes.

Slide 34
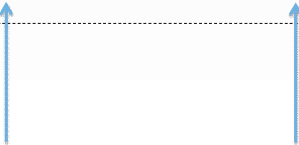
CONCEPT

String Interpolation

```
I have 4 apples
```

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

To escape out to display a ruby variable or ruby code you use the following sequence: number sign, left curly brace, the ruby variables or ruby code, and then a right curly brace.

Slide 35

Using Node Attributes

Hard-coding the values was a start. But a better approach would be to replace with the dynamic values found from Ohai.

Objective:

❑ Update the web page file contents, in the "iis-demo" cookbook, to include system details from node attributes.

So as a group let's return the recipe and update file resource to use the dynamic attributes found by Ohai.

Slide 36

GE: Using the Node's Attributes

`~\cookbooks\iis-demo\recipe\server.rb`

```
# ... POWERSHELL_SCRIPT RESOURCE ...


file 'c:\inetpub\wwwroot\Default.htm' do
  content "<h1>Hello, world!</h1>
<h2>ipaddress: #{node['ipaddress']}</h2>
<h2>hostname: #{node['hostname']}</h2>
<h2>total memory: #{node['memory']['total']}</h2>
<h2>CPU Mhz: #{node['cpu']['0']['mhz']}</h2>"
end


# ... SERVICE RESOURCE ...
```

CHEF

Update the content attribute of the file resource to include the node details. Remember to include the details about the node in the content string we need to use string interpolation. String interpolation requires that the string be a double-quoted string. Ensure you change the single-quotes to double quotes. Then use `#{}` to escape out of the double-quoted string. Between the curly braces you can define the node object and specify which attribute you would like to display.

Using Node Attributes

That feels much better!

Objective:

✓ Update the web page file contents, in the "iis-demo" cookbook, to include system details from node attributes.

Slide 38

Lab: Verify the Changes

❏ Change directory into the "iis-demo" cookbook's directory

❏ Run kitchen test for the "iis-demo" cookbook

❏ Change directory into the home directory

❏ Run chef-client locally to verify the "iis-demo" cookbook's default recipe.

©2015 Chef Software Inc.   6-38

Again we have created a change.

1. Move into the workstation cookbook's directory.

2. Verify the changes we made to the workstation cookbook's default recipe with kitchen.

3. Return to the home directory.

4. Use 'chef-client' to locally apply the workstation cookbook's default recipe.

Slide 39

Lab: Test the 'iis-demo' Cookbook

```
$ cd ~\cookbooks\iis-demo
$ kitchen converge
$ kitchen verify
```

```
-----> Starting Kitchen (v1.4.2)
-----> Setting up <default-windows-2012r2>...
       Finished setting up <default-windows-2012r2> (0m0.00s).
-----> Verifying <default-windows-2012r2>...
..


Finished in 1.69 seconds (files took 1.01 seconds to load)
2 examples, 0 failures


       Finished verifying <default-windows-2012r2> (0m2.60s).
-----> Kitchen is finished. (0m7.44s)
```

Change into the 'iis-demo' cookbook's directory and then converge and verify the cookbook. This ensures that change we made was done without an error and did not break any of the existing functionality.

Slide 40

Lab: Apply the 'iis-demo' Cookbook's Default Recipe

```
$ cd ~
$ chef-client --local-mode -r "recipe[iis-demo]"
```

```
[2015-12-23T23:15:24+00:00] WARN: No config file found or specified on command
line, using command line options.
Starting Chef Client, version 12.5.1
resolving cookbooks for run list: ["iis-demo"]
Synchronizing Cookbooks:
  - iis-demo (0.1.0)
Compiling Cookbooks...
Converging 3 resources
Recipe: iis-demo::server
  * powershell_script[Install IIS] action run
    - execute "C:\Windows\system32\WindowsPowerShell\v1.0\powershell.exe" -
NoLogo -NonInteractive -NoProfile -ExecutionPolicy Bypass -InputFormat None -
```

If everything passes, change to the home directory and then run `chef-client` to apply the iis-demo cookbook locally to the system.

Slide 41

Change Means a New Version

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

Objective:

❑ Determine the new version number
❑ Update the version of the "iis-demo" cookbook
❑ Commit the changes to the "iis-demo" cookbook to version control

Whenever you make a change to the cookbook it is important to test it to verify that it works. It is also important to update the version of the cookbook to correspond with the change in the features provided by the cookbook.

Together, let's talk about version numbers, update the version number, and then commit the changes to version control.

Slide 42

CONCEPT

Cookbook Versions

A cookbook version represents a set of functionality that is different from the cookbook on which it is based.

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

6-42

CHEF

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.

The first version of the cookbook displayed a simple hello message. Now the page displays the hello message with additional details about the system. The changes that we finished are new features of the cookbook.

Slide 43

Cookbooks use semantic version. The version number helps represent the state or feature set of the cookbook. Semantic versioning allows us three fields to describe our changes: major; minor; and patch.

Major versions are often large rewrites or large changes that have the potential to not be backwards compatible with previous versions. This might mean adding support for a new platform or a fundamental change to what the cookbook accomplishes.

Minor versions represent smaller changes that are still compatible with previous versions. This could be new features that extend the existing functionality without breaking any of the existing features.

And finally Patch versions describe changes like bug fixes or minor adjustments to the existing documentation.

Slide 44

So what kind of changes did you make to the cookbook? How could we best represent that in an updated version?

Changing the contents of an existing resource--by adding the attributes of the node doesn't seem like a bug fix and it doesn't seem like a major rewrite. It is like a new set of features while remaining backwards compatible. That sounds like a Minor changes based on the definitions provided by the Semantic Versioning documentation.

Slide 45

Change Means a New Version

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

Objective:

- ✓ Determine the new version number
- ❑ Update the version of the "iis-demo" cookbook
- ❑ Commit the changes to the "iis-demo" cookbook to version control

With the idea in mind that we want to update the minor version number we need to find that version number within the cookbook. When we do, we will "bump" the number - which means to increase the value by 1.

Slide 46

GE: Update the Cookbook Version

`~\cookbooks\iis-demo\metadata.rb`

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

Edit the "iis-demo" cookbook's metadata file. Find the line that specifies the version and increase the minor value by one.

Slide 47

Change Means a New Version

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

Objective:

- ✓ Determine the new version number
- ✓ Update the version of the "iis-demo" cookbook
- ❑ Commit the changes to the "iis-demo" cookbook to version control
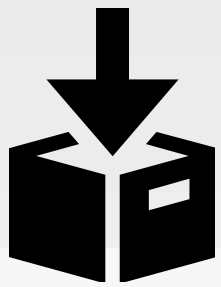

Now the last thing is to save those changes in version control. This will allow us in the future to return to this version of the code if we need to reproduce issues when someone uses the cookbook.

Slide 48

The last thing to do is commit our changes to source control. Change into the directory, add all the changed files, and then commit them.

Slide 49

Change Means a New Version

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

Objective:

- ✓ Determine the new version number
- ✓ Update the version of the "iis-demo" cookbook
- ✓ Commit the changes to the "iis-demo" cookbook to version control

Now that you have bumped the version number and committed your work you now have experienced the entire workflow when working with a cookbook.

Slide 50
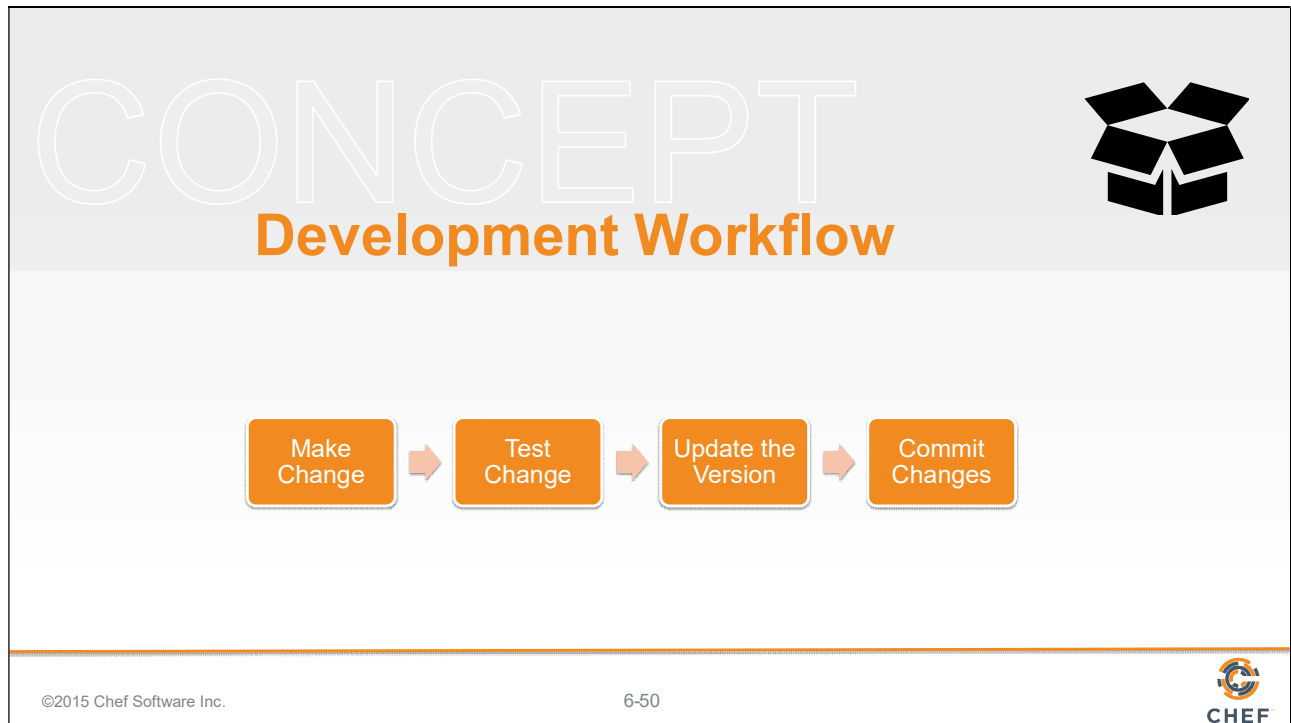
First you make the changes to the policies defined within your cookbook to meet the new requirements. Test the changes that you have by writing and executing the tests that are defined. When all the tests pass it is time to update the version of the cookbook. Then finally commit your changes. At this point you would then share those changes with your central source code repository and share the cookbook with your team.

It is important to note that some individuals like to write the tests before they make changes the system. This is called Test-Driven Development (TDD). This often helps you when developing cookbooks to explicitly define the expectations of the code you are about to write. This workflow is often adopted after one becomes more comfortable with this first workflow.

Slide 51

DISCUSSION

Discussion

What is the major difference between a single-quoted string and a double-quoted string?

How are the details about the system available within a recipe?

How does the version number help convey information about the state of the cookbook?

CHEF

Answer these questions.

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

With that we have added all of the requested features.

What questions can we help you answer?

In general or about specifically about ohai, the node object, node attributes, string interpolation, or semantic versioning.

Slide 53