14: Environments

Environments
Using Environments to Reflect Organization Patterns and Workflow

©2015 Chef Software Inc.
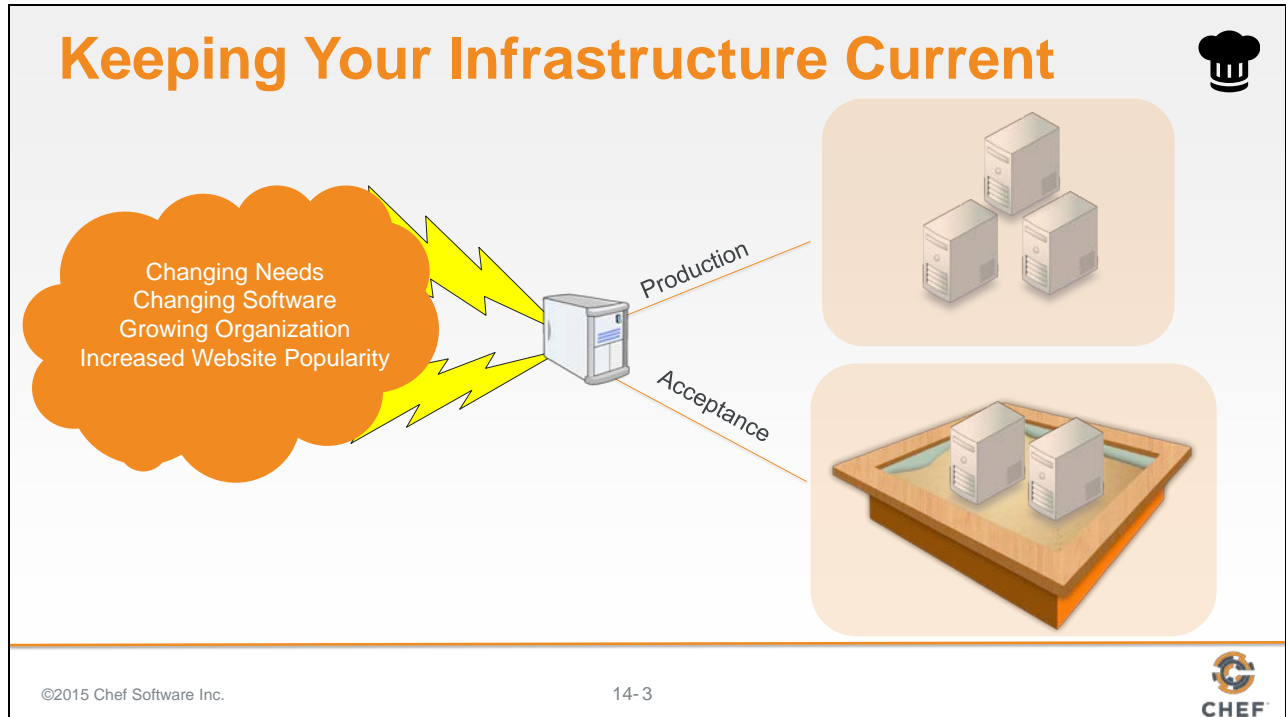
Slide 2

Objectives

After completing this module, you should be able to

➢ Create an environment

➢ Deploy a node to an environment

➢ Update a search query to be more exact

In this section, you will learn how to create an environment, deploy a node to an environment, and update a search query to be more exact.

Slide 3

Keeping Your Infrastructure Current

Changing Needs
Changing Software
Growing Organization
Increased Website Popularity

Production

Acceptance

©2015 Chef Software Inc.   14- 3

So, we have updated our myhaproxy cookbook to dynamically search for and update nodes. Everything is as it should be.

But our system is like a living, breathing thing that must grow and be updated to fit our changing needs. We need to find a way to update and test new tools, features and settings without impacting our current production system.

Of course, we have local testing tools like Test Kitchen to help us verify that our individual cookbooks work before we upload them to the Chef Server. But, that is not always enough. We may want to build, test, and release new features to our cookbooks but we do not immediately want all of our nodes to immediately use them.

For example, what if we had a requirement to update our apache cookbook with a new front page for our application? The release date of our new service with the sign up page does not go live for a week. So, we want to build, test, and upload that cookbook to the Chef Server without actually applying the cookbook until the release date. How would we accomplish that?
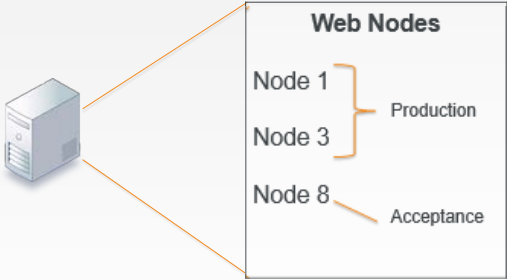
This is where environments are useful.

Slide 4

You likely are familiar with the concept of environments. An environment can best be defined as a logical separation of nodes that most often describe the life-cycle of an application.

Each environment signifies different behaviors and policies to which a node adheres for a given application or platform. For example, environments can be separated into 'acceptance' and 'production'.

"Acceptance" would be where we may make allowances for constant change and updates and for applications to be deployed with each release.

"Production" might be where we lock down our infrastructure and policies. Production would be what the outside world sees, and would remain unaffected by changes and upgrades until you specifically release them.
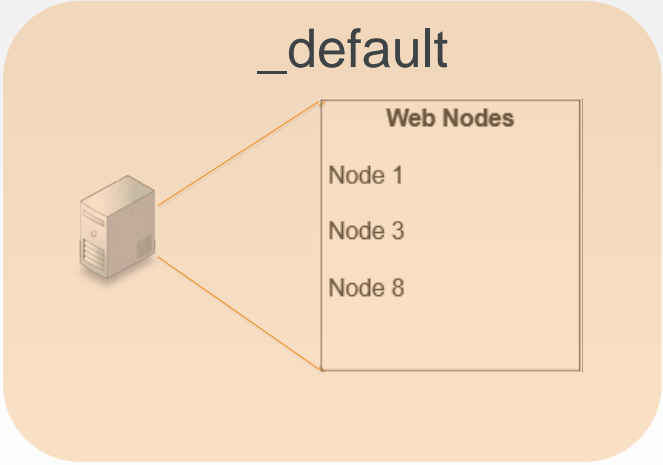
Chef also has a concept of an environment. A Chef environment allows us to define a list of policies that we will allow by defining a cookbook.

Slide 5

Environments

Every organization or infrastructure starts with the _default environment.

_default

Web Nodes

Node 1

Node 3

Node 8

©2015 Chef Software Inc.   14- 5

Chef also has a concept of an environment. Chef uses environments to map an organizations real-life workflow to what can be configured and managed using the Chef server.

Every organization begins with a single environment called the _default (underscore default) environment, which cannot be modified or deleted.

Therefore, you must create custom environments to define your organization's workflow.

Slide 6

GE: Production

Let's create a reliable environment for our nodes.

Objective:

❑ Deploy Our Site to Production

First, we need to create a Production environment. This is where we lock down our infrastructure and policies to a specific version of the myhaproxy cookbook.

Slide 7

GE: Using 'knife environment --help'

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

Because we still are communicating with the Chef server, let's ask Chef for help regarding available environment commands.

So change into chef-repo and then run 'knife environment --help'.

Slide 8

GE: View List of Defined Environments

```
$ knife environment list
```

```
_default
```

Remember, we use 'list' to view existing environments.

As previously stated, we see the _default environment has already been created.

Slide 9

GE: Viewing the _default Environment

```
$ knife environment show _default

chef_type:          environment
cookbook_versions:
default_attributes:
description:        The default Chef environment
json_class:         Chef::Environment
name:               _default
override_attributes:
```

Let's see how this environment looks.

Slide 10

GE: Searching All of Our Nodes

```
$ knife search node "*:*"
```

```
3 items found

Node Name:   node1
Environment: _default
FQDN:        ip-172-31-8-68.ec2.internal
IP:          54.175.46.24
Run List:    role[web]
Roles:       web
Recipes:     apache, apache::default, apache::server
Platform:    centos 6.7
Tags:
```

If we search our nodes, we see that all three nodes have been set to the _default environment. How do we change this?

Slide 11

GE: Create an environments Directory

```
$ mkdir environments
```

First, we need to make a new environments directory. (Be sure you are still in the chef-repo before you do this.)

Slide 12

GE: Create the production.rb

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

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

cookbook "apache", "= 0.2.1"
cookbook "myhaproxy", "= 1.0.0"
```

Then we need to create a production.rb file. Like in the roles.rb files, we must provide a name and description.

Additionally, we need to define cookbook restrictions to lock down specific versions of both the apache and myhaproxy cookbooks. By adding this information to production.rb, we are telling our nodes to use these specific versions of these specific cookbooks.

Obviously, what this means is that as we work on newer versions of these cookbooks, we won't break anything in the production environment.

Okay, so now that we have captured our "good" environment in this file, let's save it and upload it.

Slide 13

GE: Upload the production.rb File

```
$ knife environment from file production.rb

Updated Environment production
```

Using the knife environment command, let's upload the production.rb file. This should be familiar because it is just like the command we used to upload roles.

Slide 14

GE: View the List of Environments

```
$ knife environment list
```

```
_default
production
```

Okay, let's use our list command to make sure the file uploaded correctly.

Slide 15

GE: View the Production Environment

```
$ knife environment show production

chef_type:          environment
cookbook_versions:
  apache:    = 0.2.1
  myhaproxy: = 1.0.0
default_attributes:
description:         Where we run production code
json_class:         Chef::Environment
name:               production
override_attributes:
```

If we use the knife environment show command, we can see how the production.rb file looks.

Note the cookbook versions that we set are shown here.

Slide 16

Now, we need to set the environments for our nodes. Let's ask Chef for help on that as well.

Let's use the knife node environment set command.

GE: Viewing 'knife node set --help'

```
$ 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
```

©2015 Chef Software Inc.   14-17

But how does that command work, exactly?

It looks like we just add the environment name at the end of the command to set that environment on a node.

Slide 18

GE: Using 'knife environment node set'

```
$ knife node environment set node1 production

node1:
  chef_environment:
```

So, let's do that for node1.

The results don't really tell us much, so let's take a look at node1.

Slide 19

GE: Viewing node1's Attributes

```
$ knife node show node1

Node Name:    node1
Environment:  production
FQDN:         ip-172-31-8-68.ec2.internal
IP:           54.175.46.24
Run List:     role[web]
Roles:        web
Recipes:      apache, apache::default, apache::server
Platform:     centos 6.7
Tags:
```

Using knife node show, we can see node1's attributes. Note that it has indeed been set to the production environment.

Slide 20

Lab: Set More Nodes to Production

❑ Set node2's environment to production

©2015 Chef Software Inc.   14-20

Let's do the same thing for node2.

Instructor Note: Allow 2 minutes to complete this exercise.

Slide 21

Lab: Set node2's Environment to Production

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

```
node2:
  chef_environment:
```

Slide 22

Lab: Verify node2 is Set to Production

```
$ knife node show node2

Node Name:   node2
Environment: production
FQDN:        ip-172-31-0-128.ec2.internal
IP:          54.210.192.12
Run List:    role[proxy]
Roles:       proxy
Recipes:     myhaproxy, myhaproxy::default, haproxy::default,
haproxy::install_package
Platform:    centos 6.6
Tags:
```

And, it looks like node2 was successfully set to the production environment.

Production

Let's create a reliable environment for our nodes.

Objective:

✓ Deploy our site to Production

Slide 24

Lab: Acceptance Environment

❑ Create an environment named "acceptance"
  that has no cookbook restrictions.
❑ Move node3 into the acceptance environment
❑ Run chef-client on all the nodes

Now, let's create the environment we can use to change and update the cookbooks without affecting our production environment. A sandbox, if you will.

Let's call this our "Acceptance environment".

Instructor Note: Allow 10 minutes to complete this exercise.

Slide 25

Lab: Create a New Environment File

`~/chef-repo/environments/acceptance.rb`

```
name "acceptance"
description "Where code and applications are
tested"
# No Cookbook Restrictions
```

First, let's create a new rb file in our chef-repo/environments directory. Let's name it acceptance.

In the Acceptance environment, we don't want to lock-down the cookbook versions, so we are not going to place restrictions on the cookbooks.

Slide 26

Lab: Upload the .rb File

```
$ knife environment from file acceptance.rb

Updated Environment acceptance
```

Let's upload that .rb file to the Chef server.

Slide 27

Lab: Verify that the Environment was Set

```
$ knife environment list

_default
production
acceptance
```

And let's make sure that this environment file was added properly.

Slide 28

Lab: Verify the Contents of the Environment

```
$ knife environment show acceptance

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

And last, but not least, let's ask the Chef Server to show us the acceptance environment.

Slide 29

Lab: Set node 3 to the Acceptance Environment

```
$ knife node environment set node3 acceptance

node3:
  chef_environment: acceptance
```

Okay, let's set node3 to the acceptance environment.

Slide 30

Lab: Verify that the Environment Was Set

```
$ knife node show node3
```

```
Node Name:    node3
Environment: acceptance
FQDN:         ip-172-31-0-127.ec2.internal
IP:           54.210.86.164
Run List:     role[web]
Roles:        web
Recipes:      apache, apache::default, apache::server
Platform:     centos 6.6
Tags:
```

And confirm that it has been set properly.

Slide 31

Lab: Converge All the Nodes

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

ec2-54-175-46-24.compute-1.amazonaws.com  Starting Chef Client, version 12.3.0
ec2-54-210-86-164.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
ec2-54-210-86-164.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"]
ec2-54-210-86-164.compute-1.amazonaws.com Synchronizing Cookbooks:
ec2-54-210-86-164.compute-1.amazonaws.com   - apache
ec2-54-210-86-164.compute-1.amazonaws.com Compiling Cookbooks...
ec2-54-210-86-164.compute-1.amazonaws.com Converging 3 resources
ec2-54-210-86-164.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
```

Using the knife ssh let's run chef client on all the nodes.

Slide 32

Separating Environments

Objective:

❑ Use Search to separate out the environments

❑ Update myhaproxy cookbook's version number

Now that we have created our two environments and set each node to a specific environment, we need to separate the environments to ensure that the proxy server only communicates with the production nodes.

Slide 33

So we set our web nodes to specific environments. As we manage our nodes, making changes to our cookbooks and recipes, what do you think is going to happen to Node1?

What about Node 3?

Setting the nodes is not enough. Chef does not automatically know to separate the environments. So, we have to tell it how to do that.

Slide 34

How do we do that?

First, let's answer a couple of question. As you think about the infrastructure we have created, which cookbook handles balance requests between nodes?

So if we want to make changes to that cookbook, which recipe would we change?

Answer 1: myhaproxy Answer 2: default.rb

Slide 35

In our last module, we talked about searching our nodes using Chef. Do you recall what we used to search for web nodes?

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

So, considering our search syntax, how can we further refine that syntax to search for a specific web node by environment?

Let's take a look.

Slide 36

Search Criteria

`~/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")                          —

members = []

#...
```

Looking at the default.rb file in the myhaproxy cookbook, we can review the original search syntax. If we want to search by environments, what would we need to add here?

Slide 37

GE: Modify the myhaproxy default.rb

`~/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 = []


#...
```

Search the Chef Server for all node objects that have the role equal to 'web' and also share the same environment as the current node applying this recipe. The nodes currently applying this recipe are the nodes with the role set to proxy.

Now that we've made our changes, let's save this file.

Slide 38

GE: Separate Environments

✓ Use Search to separate out the environments
❑ Update myhaproxy version number

Now that we have created our two environments and set each node to a specific environment, we need to separate the environments to ensure that the proxy server only communicates with the production nodes.

Slide 39

GE: Version the myhaproxy metadata.rb

`~/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           '1.0.1'

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

©2015 Chef Software Inc.   14-39

Before we upload the new myhaproxy cookbook to the server, we probably want to update the version number. What type of change have we made here?

Answer: Patch

Because we are performing a patch, let's set the version number to 1.0.1.

Slide 40

GE: Run 'berks install'

```
$ cd cookbooks/myhaproxy
$ 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 (1.0.1) from source at .
```

We are going to need to use Berks to upload this cookbook because it has dependencies. So first we need to cd into the cookbook.
Then run 'berks install'.

Slide 41

GE: Run 'berks upload'

```
$ berks upload

Skipping build-essential (2.2.3) (frozen)
Skipping cpu (0.2.0) (frozen)
Skipping haproxy (1.6.6) (frozen)
Uploaded myhaproxy (1.0.1) to:
'https://api.opscode.com:443/organizations/vogue'
```

©2015 Chef Software Inc.   14-41

And finally berks upload.

Slide 42

GE: Separate Environments

- ✓ Use Search to separate out the environments
- ✓ Update myhaproxy version number

Slide 43

Before we run 'chef-client' to bring everything up to date, let's think about what we've done.

First, in the production environment, we restricted our cookbooks to a specific version.

Second, we created an acceptance environment with no cookbook restrictions.

Third, we set specific nodes to each of these environments.

Fourth, we updated the myhaproxy default.rb to include environment search criteria.

And lastly, we changed the version number in the myhaproxy metadata.rb file.

What problems do you think we may encounter, given all that we've done here?

Slide 44

Lab: Update Production

❑ Update the environment named production:

```
'myhaproxy' cookbook version equal to
                 '1.0.1'
```

©2015 Chef Software Inc.   14-44

Since we changed the version of the myhaproxy cookbook, we need to revise the production.rb file to incorporate the new version.

Slide 45

Lab: Update production.rb

~/chef-repo/environments/production.rb

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

cookbook "apache", "= 0.2.1"
cookbook "myhaproxy", "= 1.0.1"
```

So let's go back into our production.rb and update it to include the new version number.

Slide 46

Lab: cd and Run 'knife environment...'

```
$ cd ~/chef-repo
$ knife environment from file production.rb
```

```
Updated Environment production
```

Change to ~/chef-repo and then run 'knife environment from file production.rb'.

Slide 47

Lab: Verify the Version Number

```
$ knife environment show production

chef_type:          environment
cookbook_versions:
  apache:    = 0.2.1
  myhaproxy: = 1.0.1
default_attributes:
description:        Where we run production code
json_class:         Chef::Environment
name:               production
override_attributes:
```

And let's make sure that the production.rb on Chef server has the correct version of myhaproxy designated.

Slide 48

Lab: Converge All Nodes

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

```
ec2-54-175-46-24.compute-1.amazonaws.com  Starting Chef Client, version 12.3.0
ec2-54-210-86-164.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
ec2-54-210-86-164.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"]
ec2-54-210-86-164.compute-1.amazonaws.com Synchronizing Cookbooks:
ec2-54-210-86-164.compute-1.amazonaws.com   - apache
ec2-54-210-86-164.compute-1.amazonaws.com Compiling Cookbooks...
ec2-54-210-86-164.compute-1.amazonaws.com Converging 3 resources
ec2-54-210-86-164.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
```

And use 'sudo chef-client' to converge all nodes.

Slide 49

Answer these questions.

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

Slide 50

DISCUSSION

Q&A

What questions can we help you answer?

Slide 51