A Taste of Chef

Introduction to Chef

DevFestDC

https://github.com/nathenharvey/devfestdc-2015/

CHEF

Nathen Harvey

- Community Director at Chef
- Co-host of the Food Fight Show
- Co-organizer of DevOpsDC meetup
- Occasional farmer – http://bit.ly/farmer-nathen
- Love Eggs – http://eggs.chef.io

- @nathenharvey
- nharvey@chef.io

CHEF

Hello!

- System Administrator?

Hello!

- System Administrator?
- Developer?

Hello!

- System Administrator?
- Developer?
- DevOp?

Hello!

- System Administrator?
- Developer?
- DevOp?
- Business Person?

Are you experienced?

- Experience with Infrastructure as Code or Configuration Management?

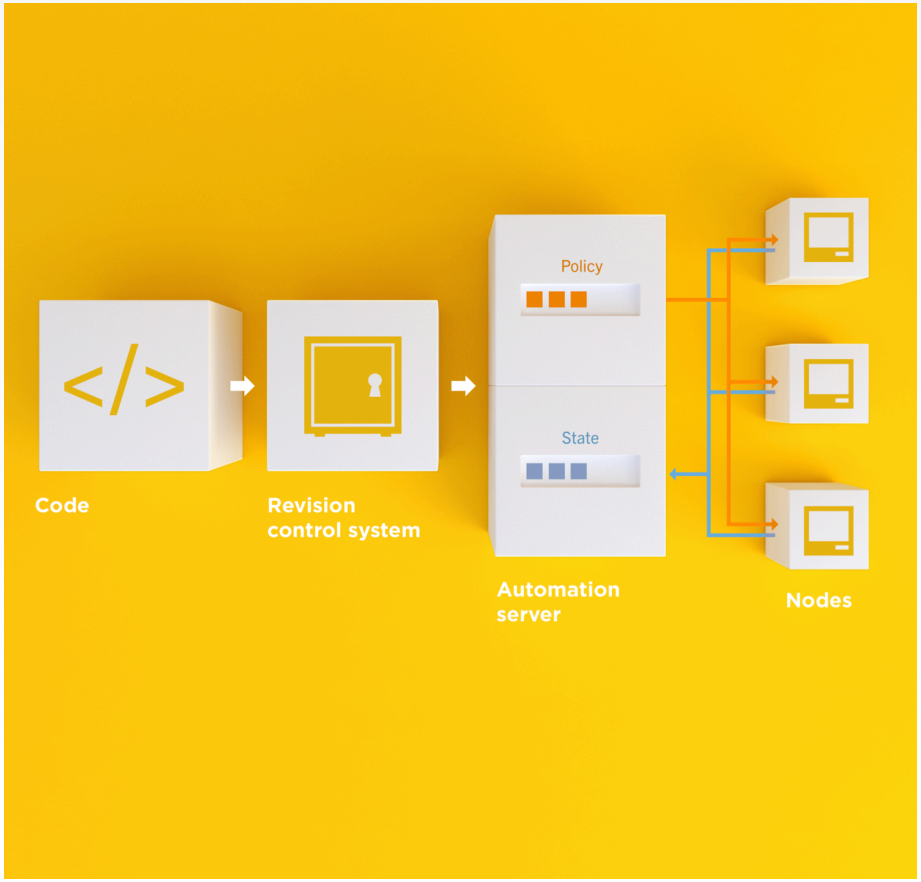
Are you experienced?

- Experience with Infrastructure as Code or Configuration Management?
- Experience with Chef?

Infrastructure as Code

CHEF

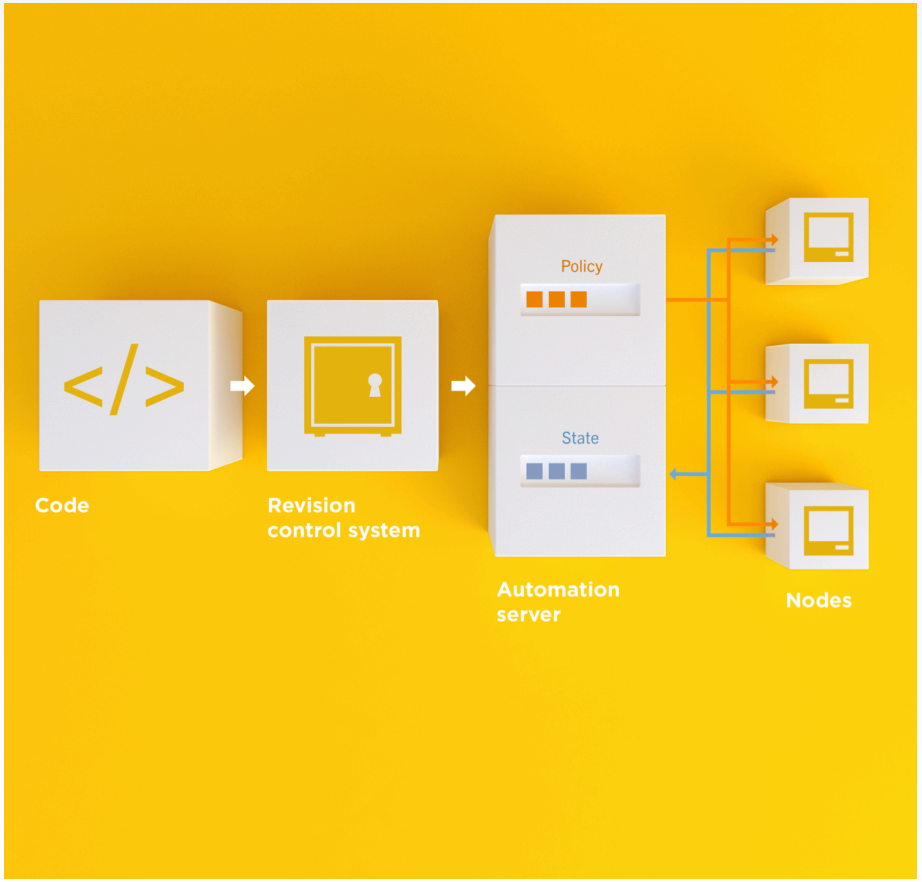
Infrastructure as Code



- Programmatically provision and configure components

CHEF

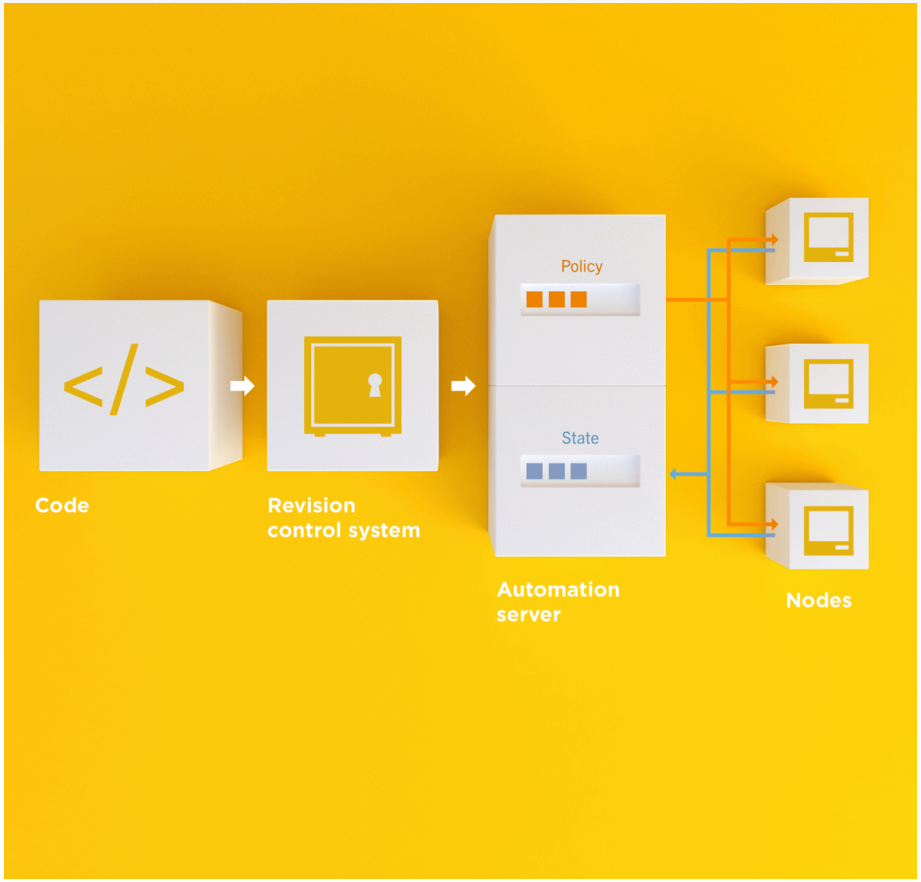
Infrastructure as Code



- Treat like any other code base

CHEF

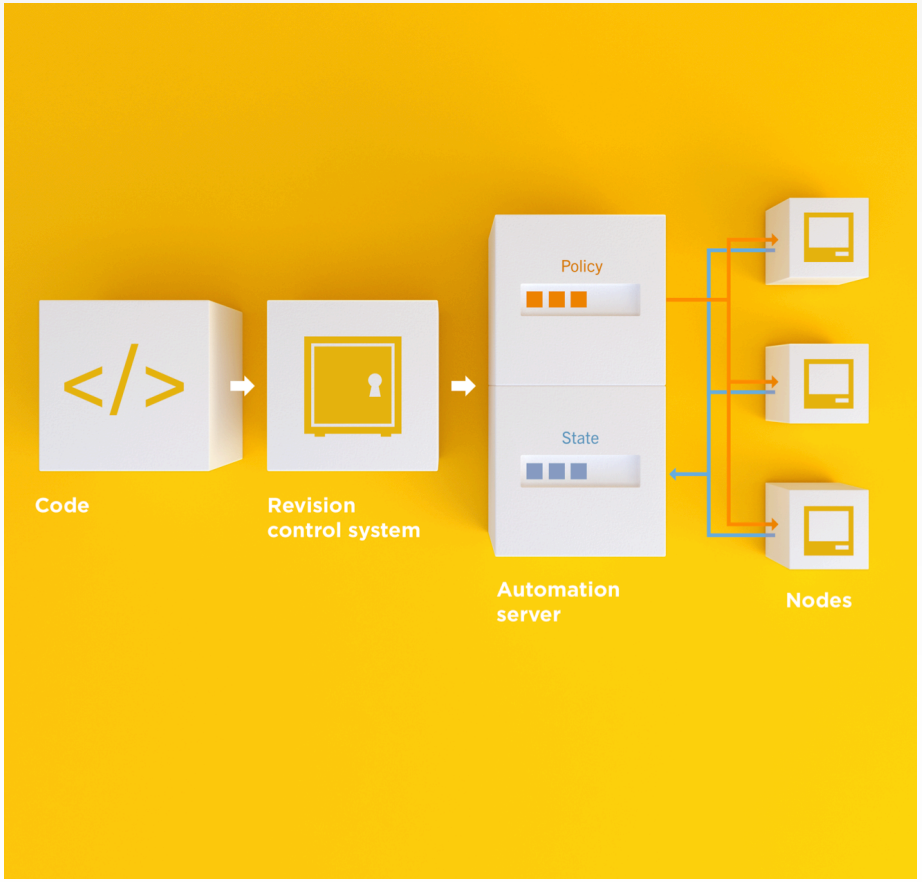
Infrastructure as Code



- Reconstruct business from code repository, data backup, and compute resources

CHEF

Infrastructure as Code



- Programmatically provision and configure components
- Treat like any other code base
- Reconstruct business from code repository, data backup, and compute resources

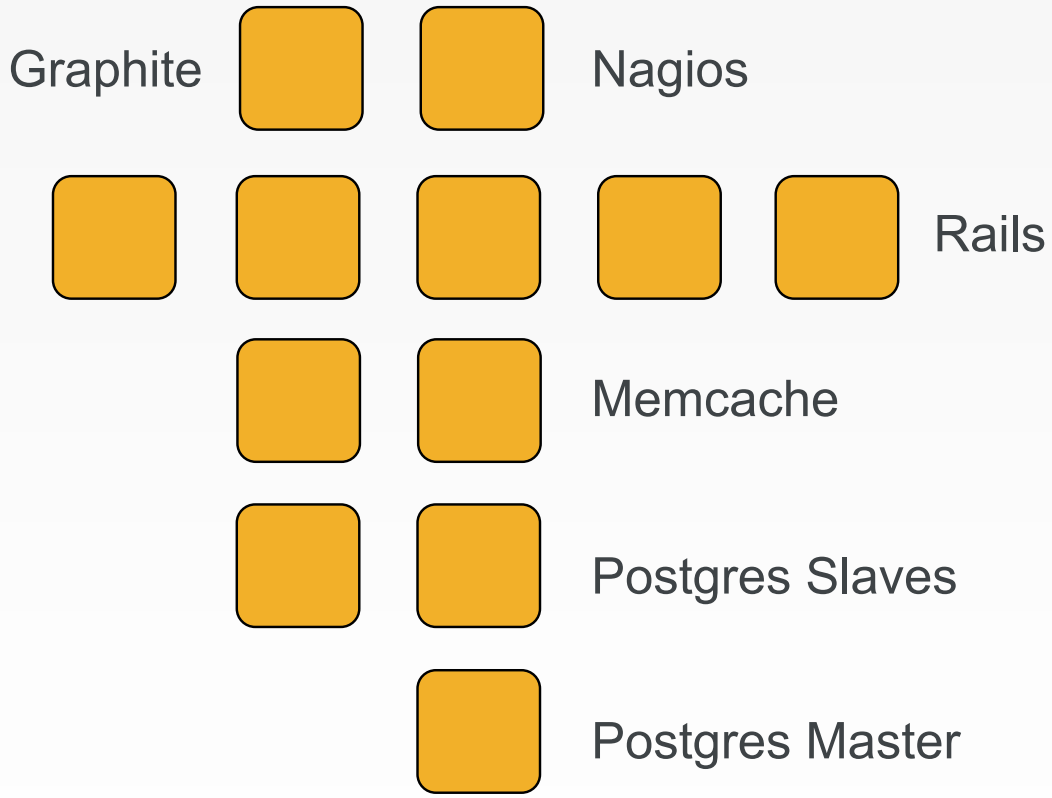
CHEF

Policy-based

- You capture the policy for your infrastructure in code
- A program ensures each node in your infrastructure complies with the policy
- A control loop keeps the system stable and allows for change when policy is updated

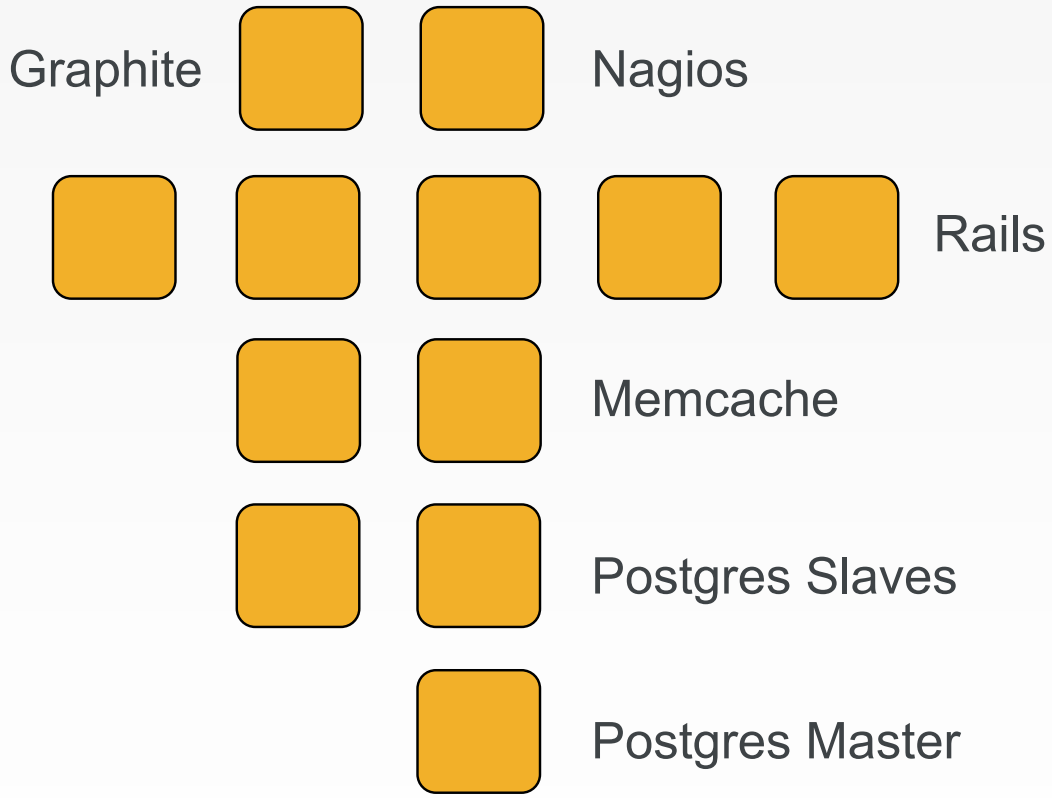
CHEF

Sample Infrastructure



CHEF

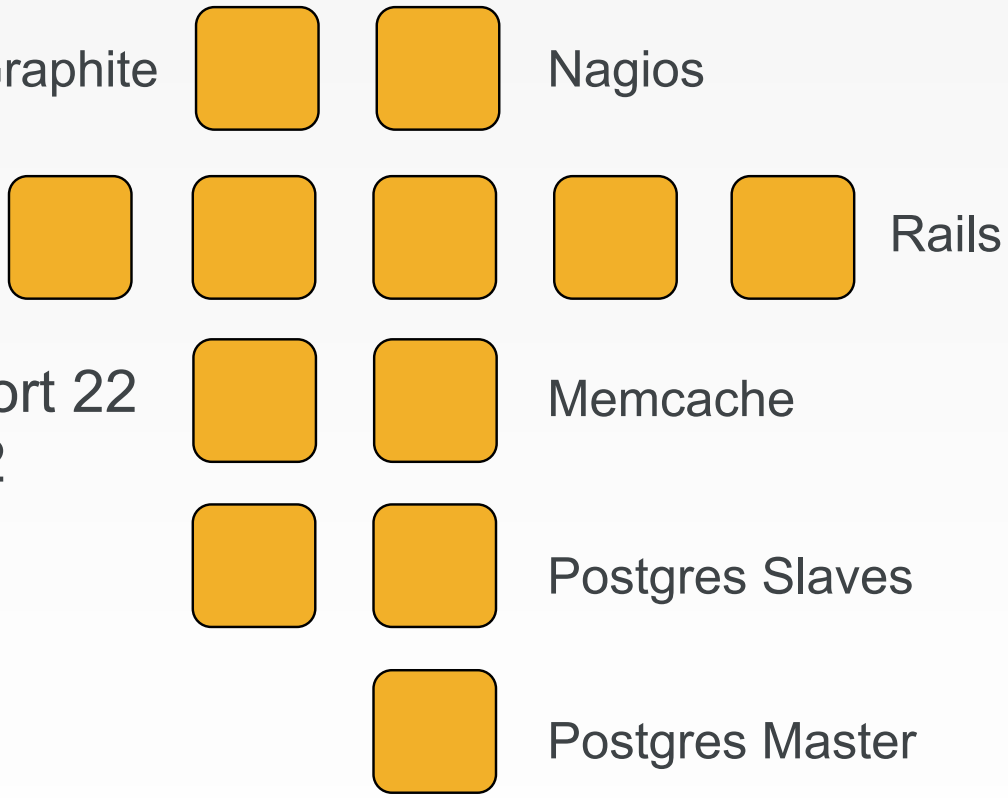
New Compliance Mandate!



CHEF

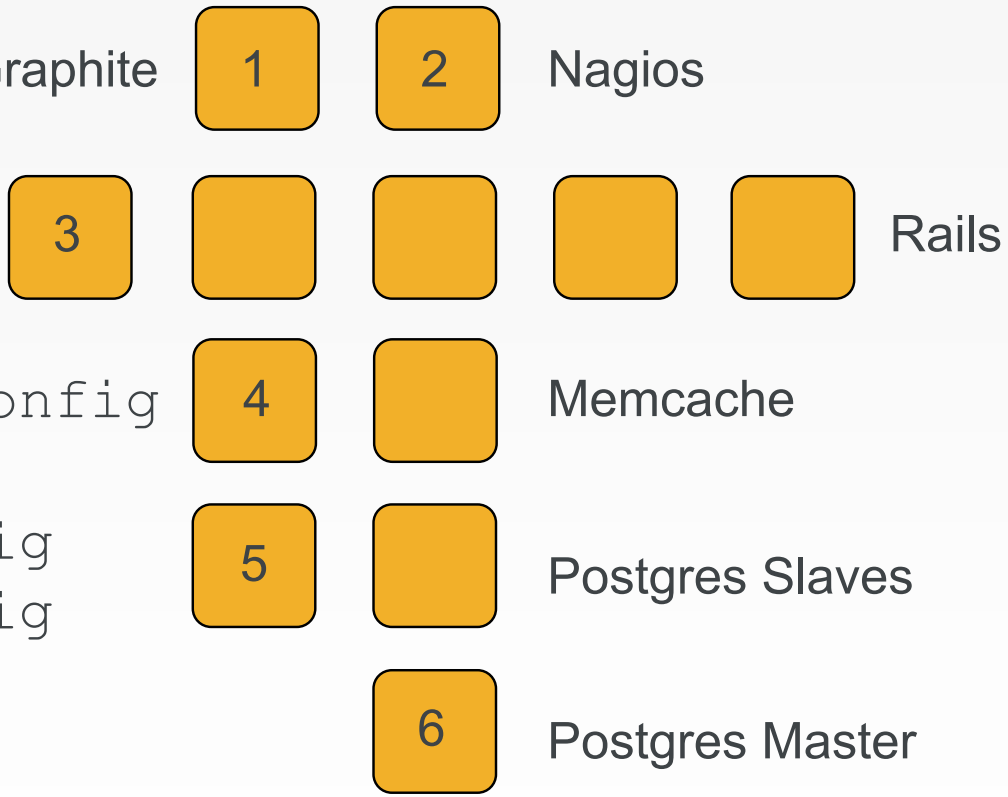
New Compliance Mandate!

- Move SSH off of port 22
- Let's put it on 2022



CHEF

6 Golden Images to Update

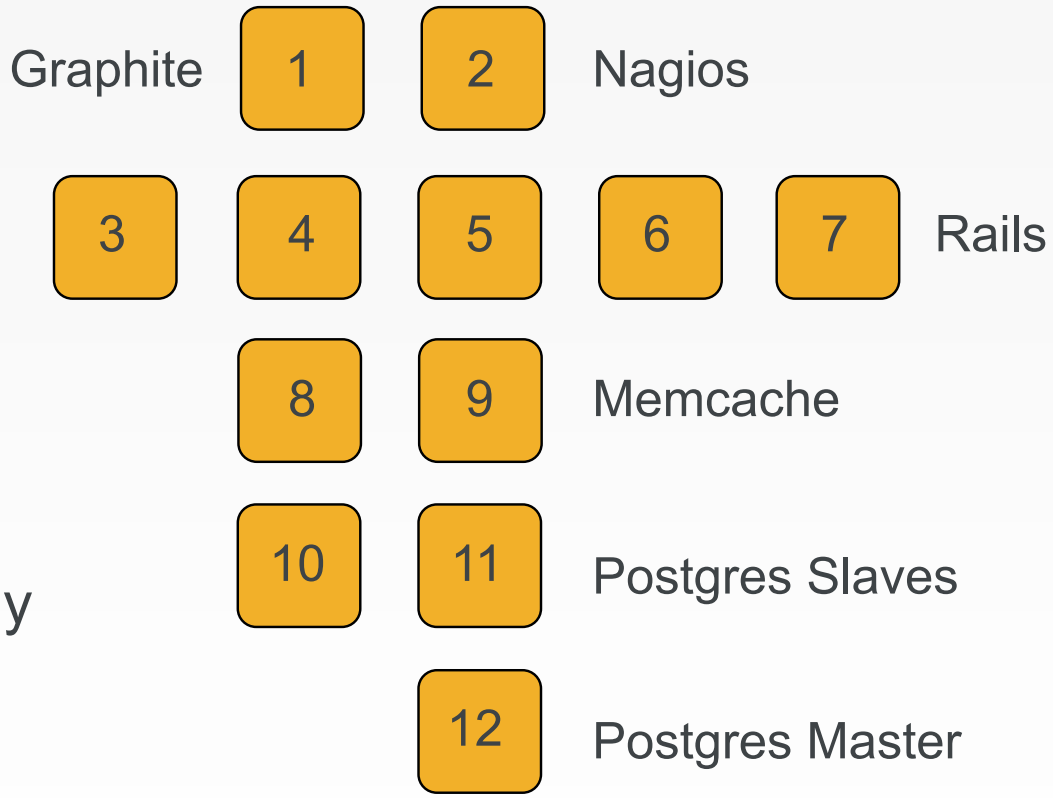



```
/etc/ssh/sshd_config

--- a/sshd_config
+++ b/sshd_config

-Port 22
+Port 2202
```

CHEF

12 Instances to replace

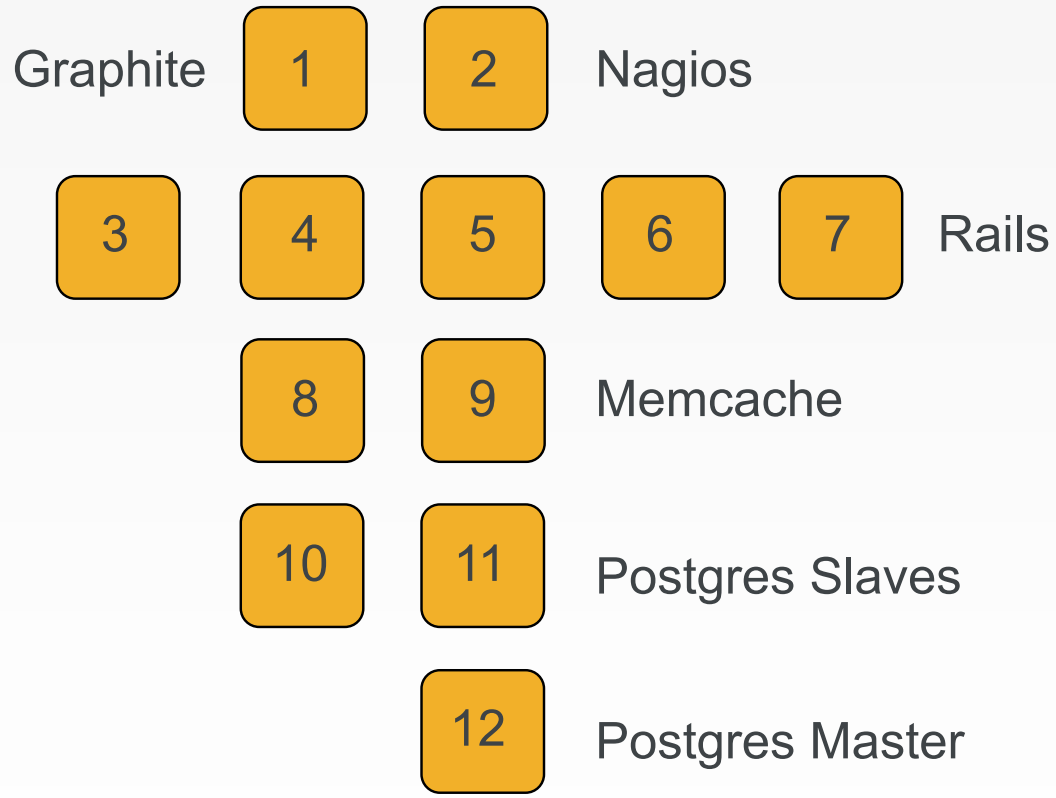
- Launch
- Delete
- Repeat
- Typically manually



CHEF

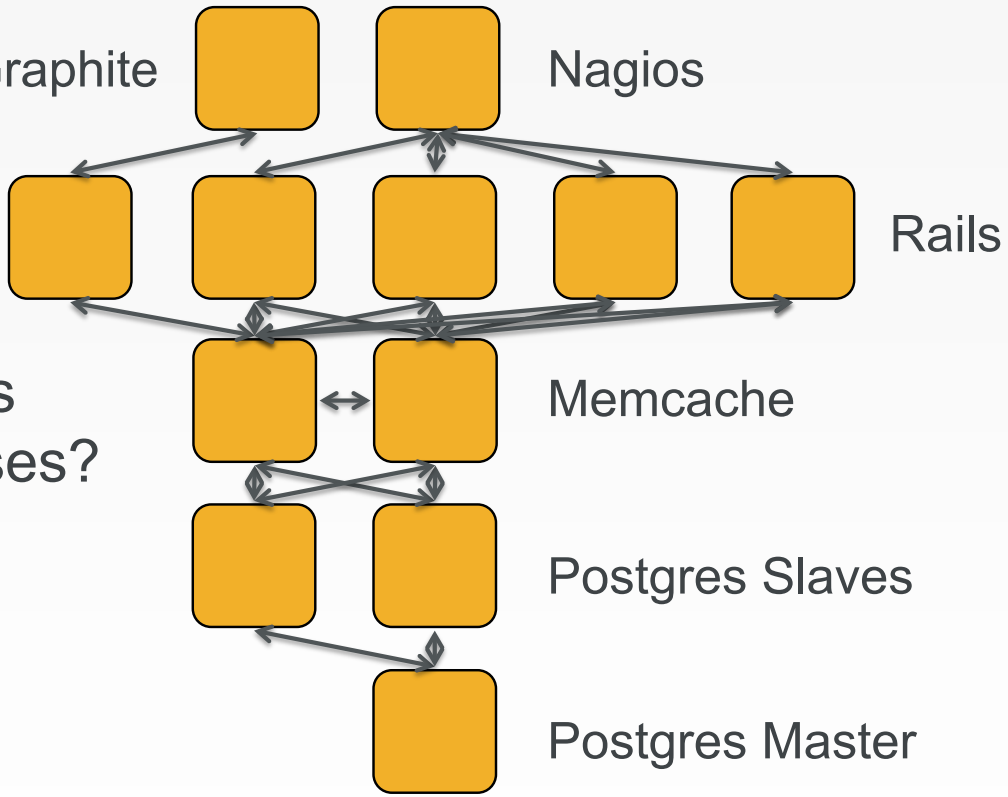
Done in maintenance window

- High stakes
- Late hours
- Risky change



CHEF

New configurations required?



Do the new instances have new IP Addresses?

* Not all connections shown

CHEF

Chef

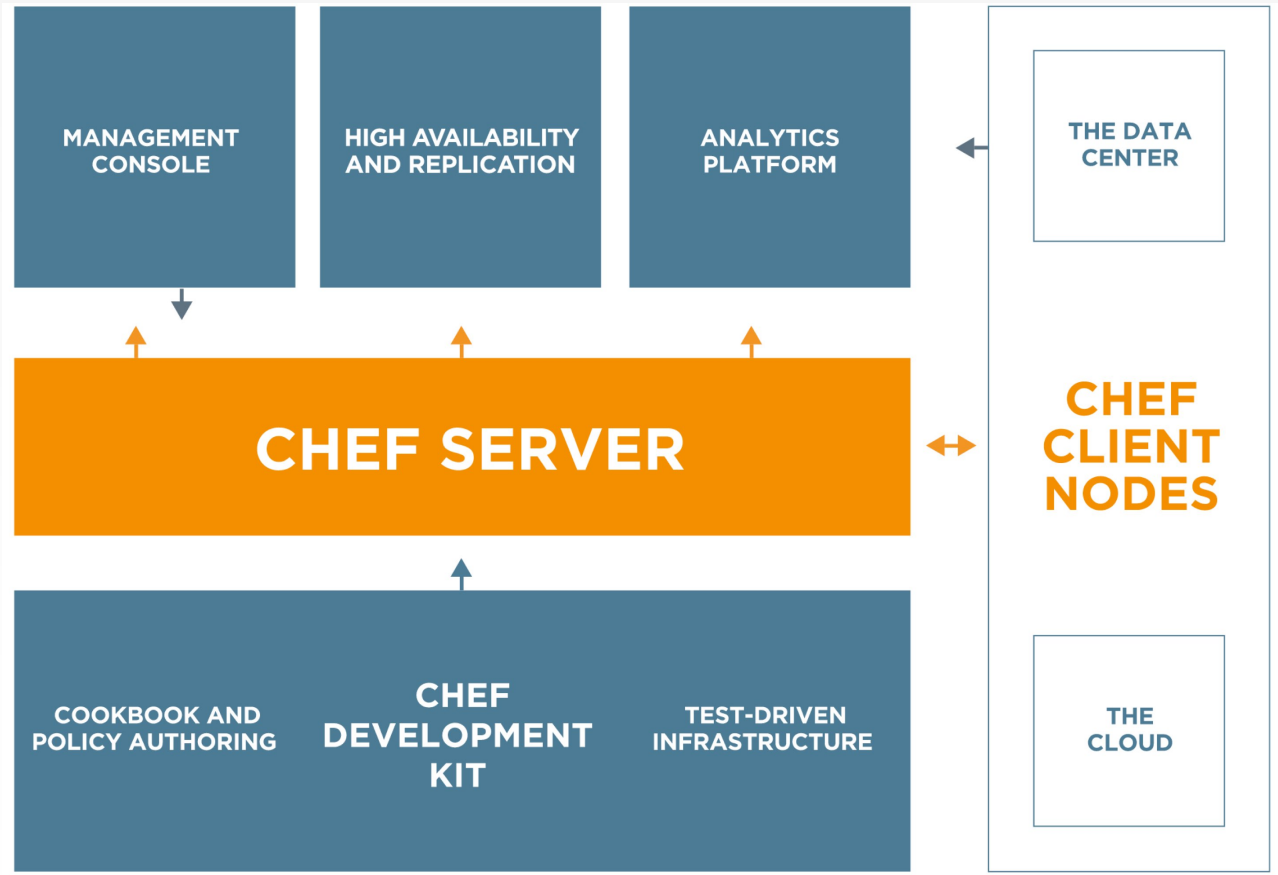
Fast, scalable, flexible IT automation

CHEF

What is Chef

- Open source framework for managing complexity in your infrastructure through policy-driven automation code
- A community of professionals
- A company

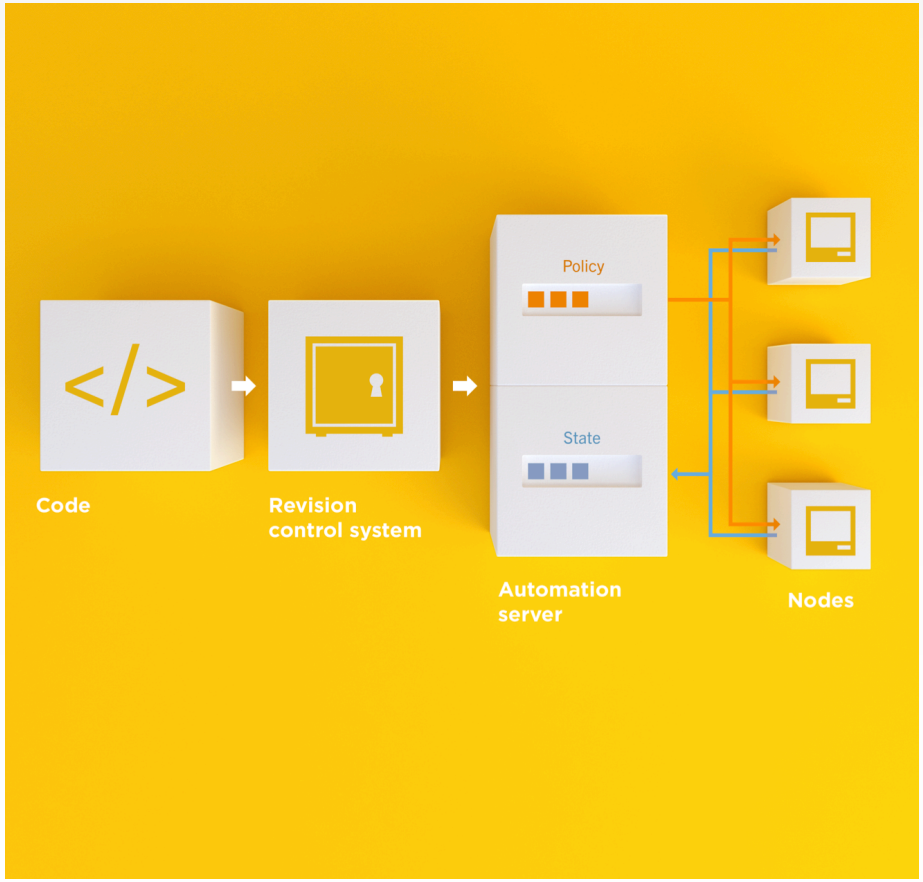
Chef



https://www.chef.io/chef/

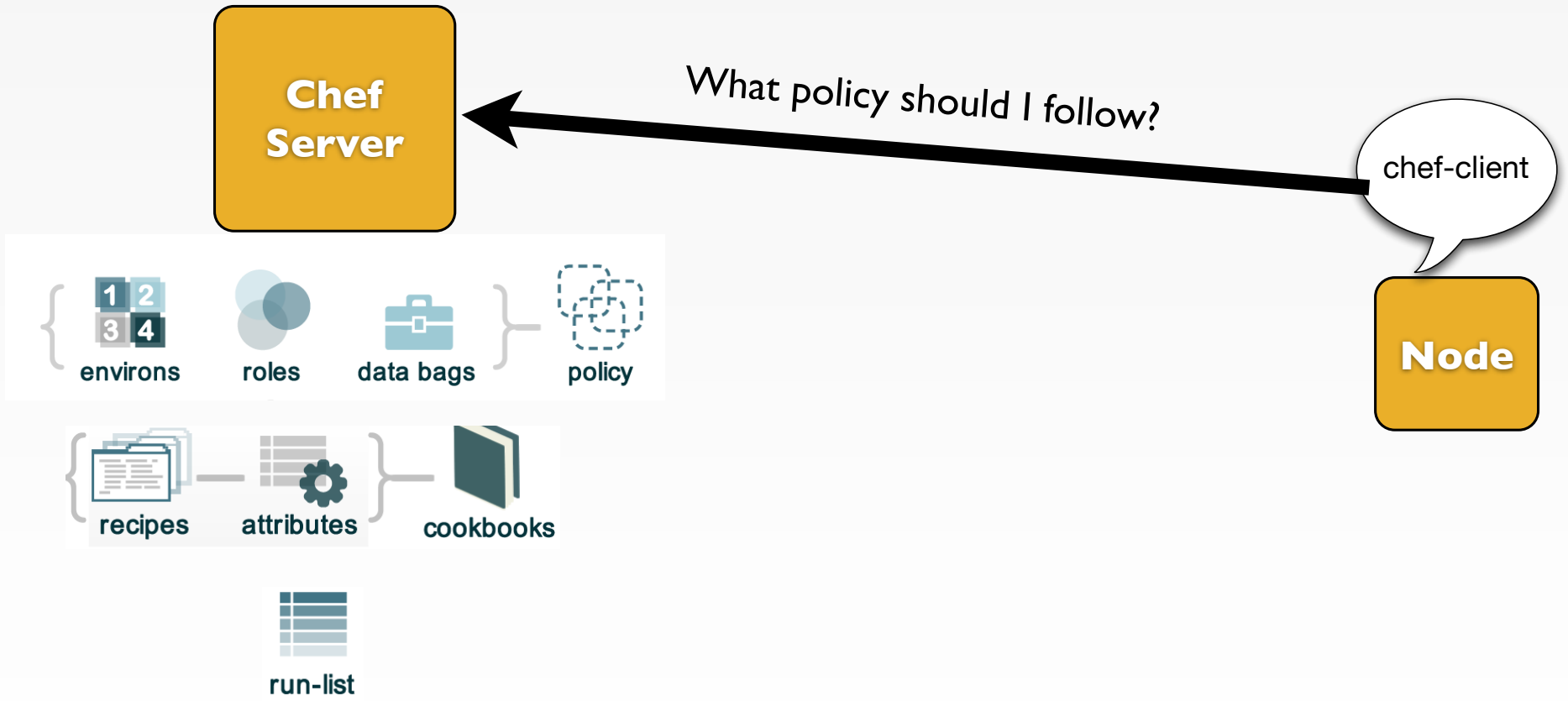
CHEF

Chef Server – Policy & State



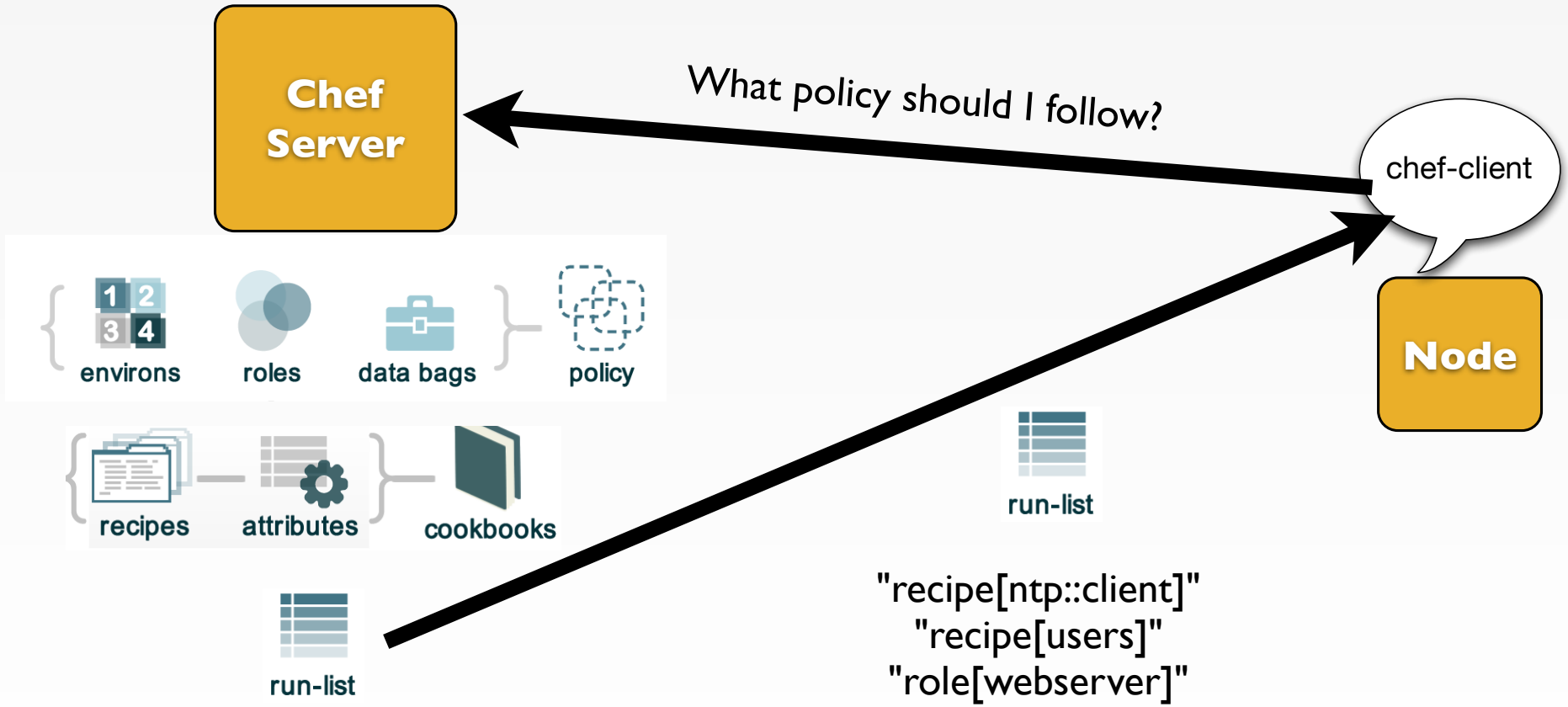
CHEF

Desired Configuration



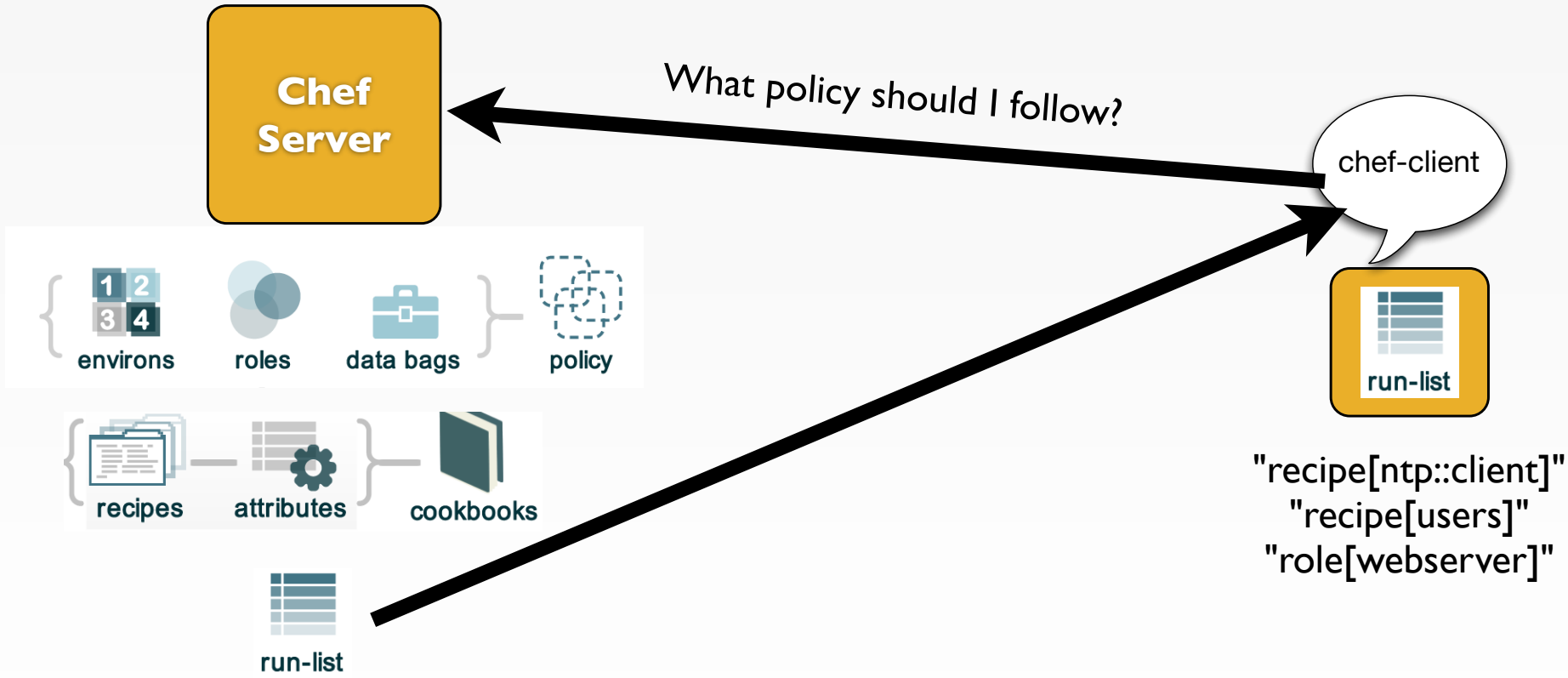
CHEF

Desired Configuration



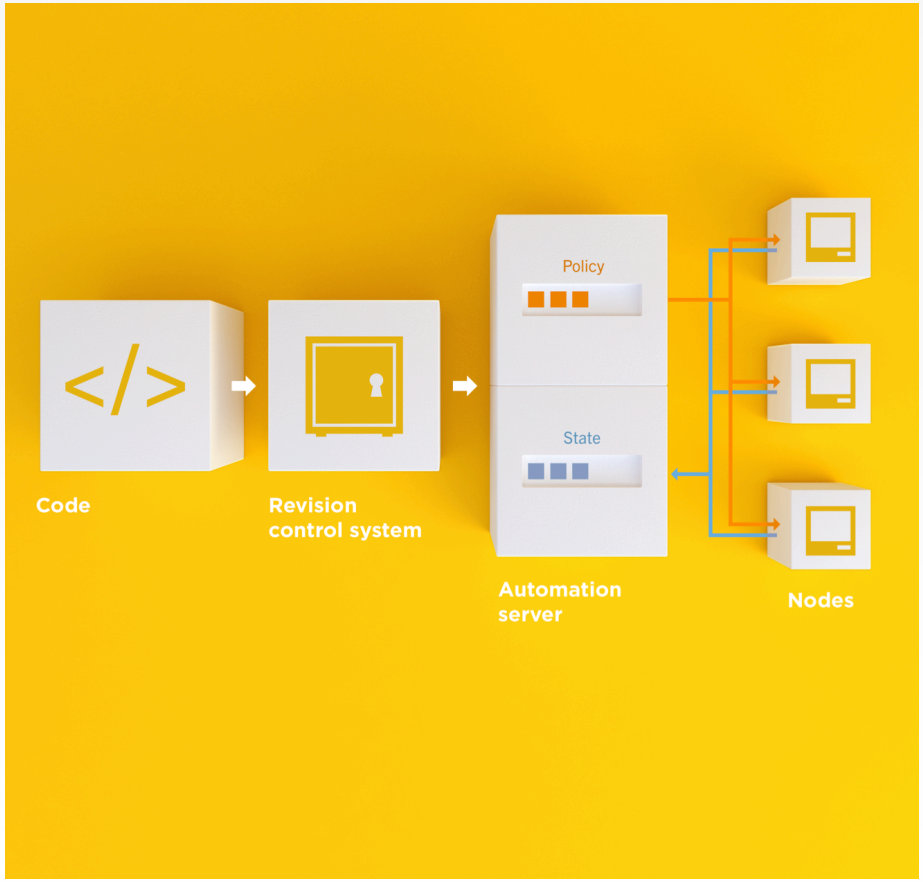
CHEF

Desired Configuration



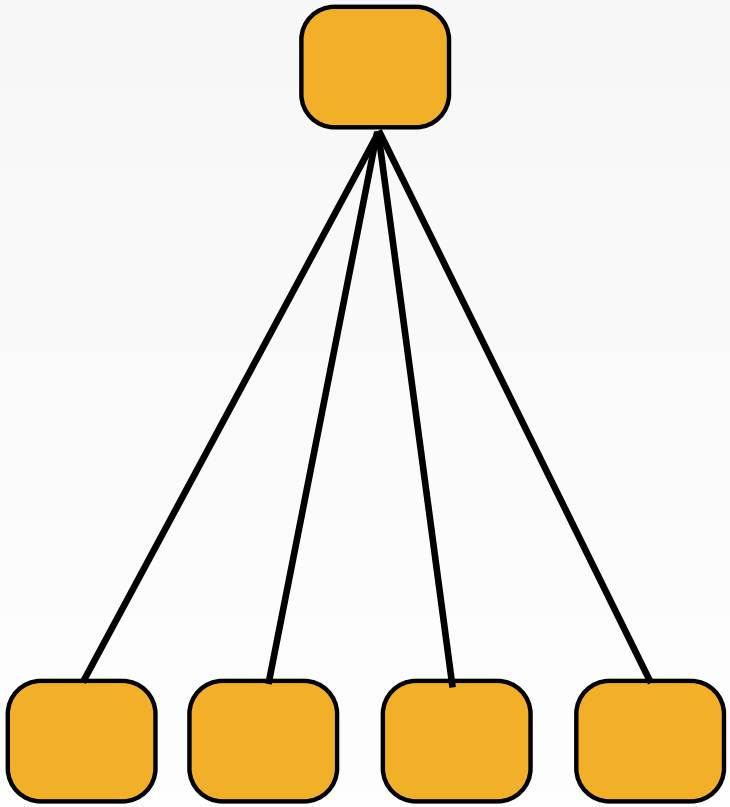
CHEF

Chef Server – Policy & State



CHEF

HA Proxy Configuration

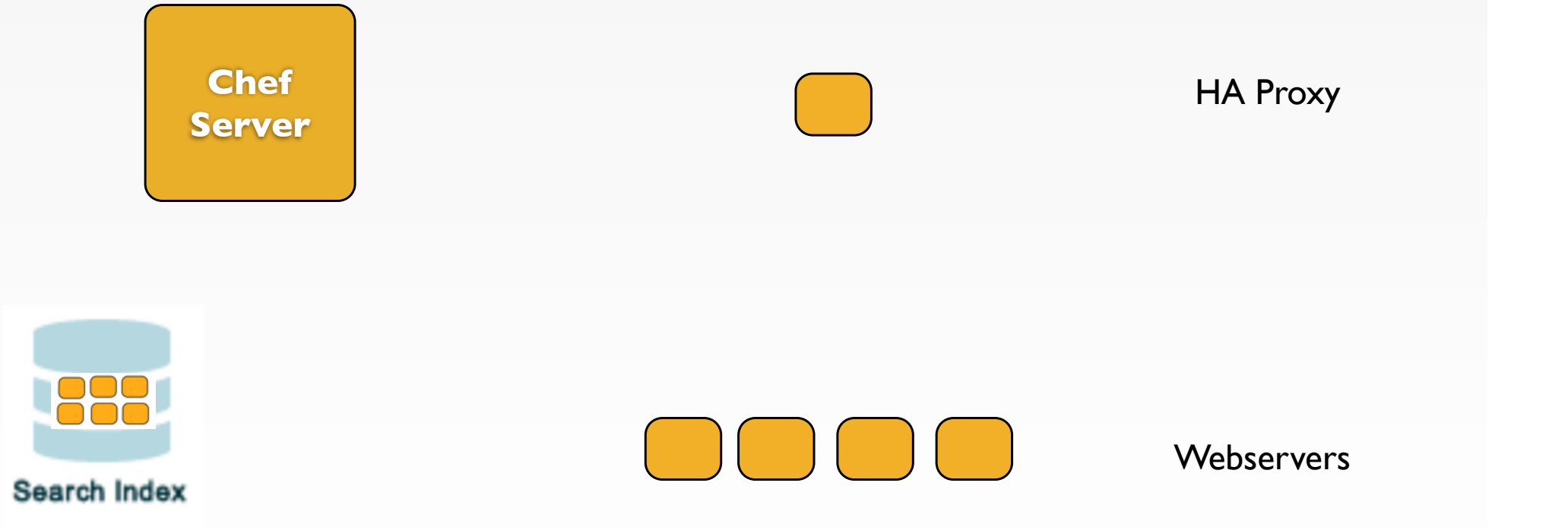


HA Proxy

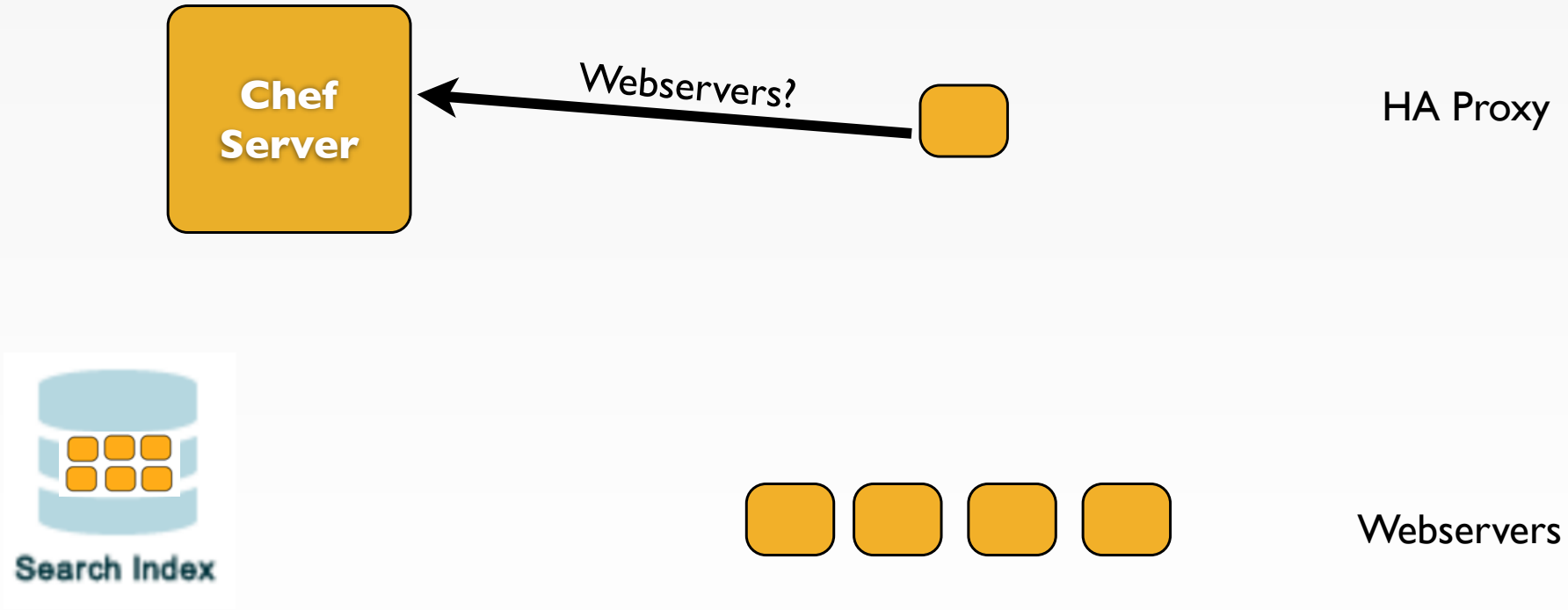
Webservers

CHEF

HA Proxy Configuration

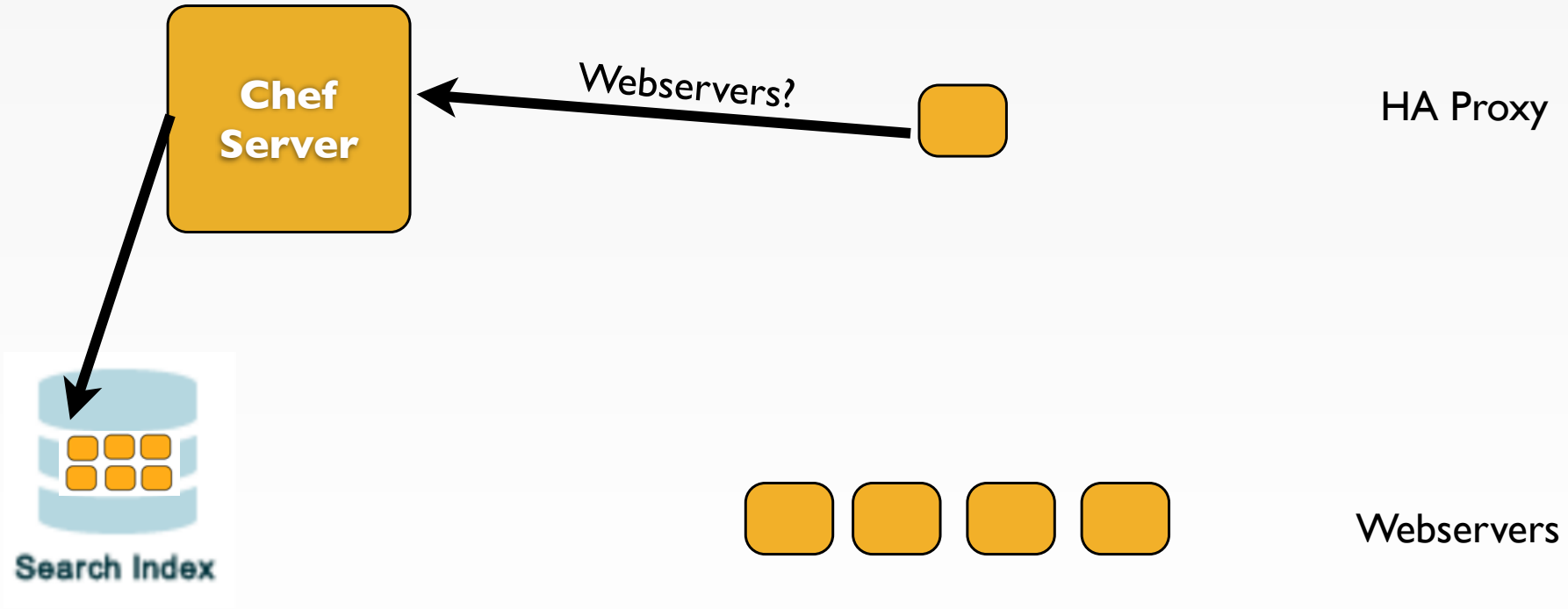


HA Proxy Configuration



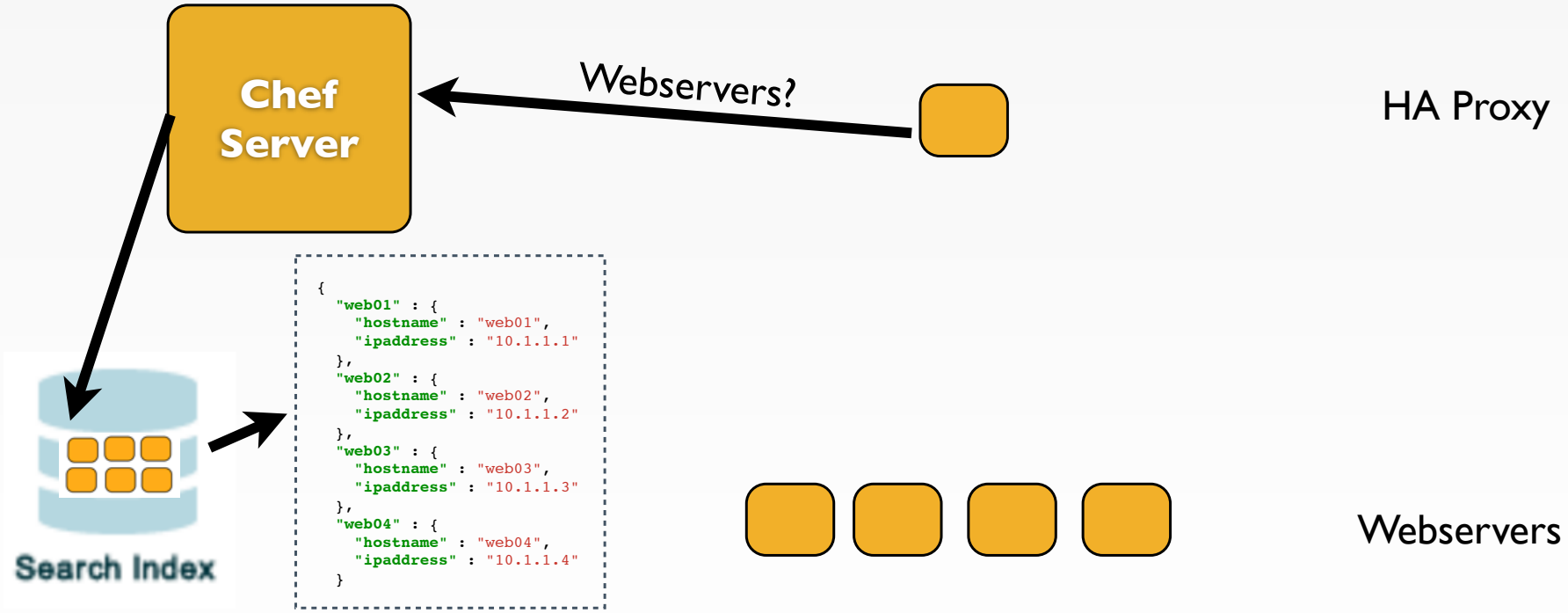
CHEF

HA Proxy Configuration



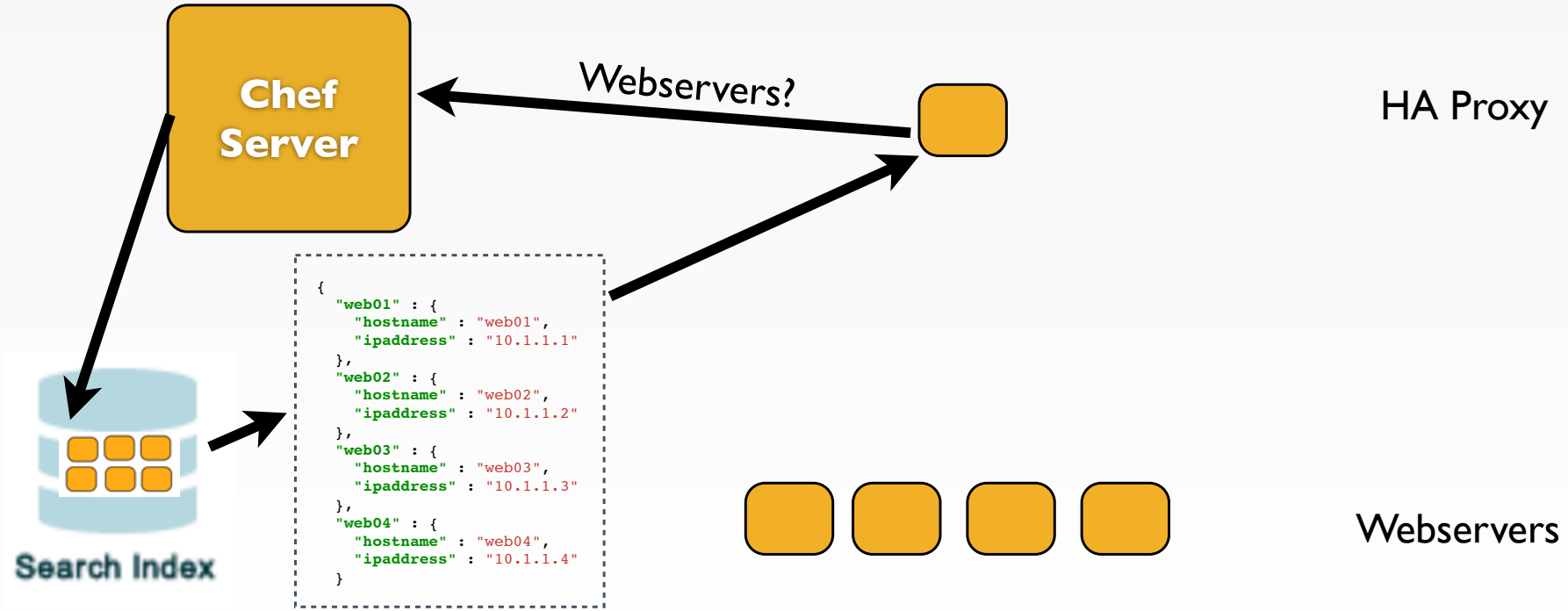
CHEF

HA Proxy Configuration



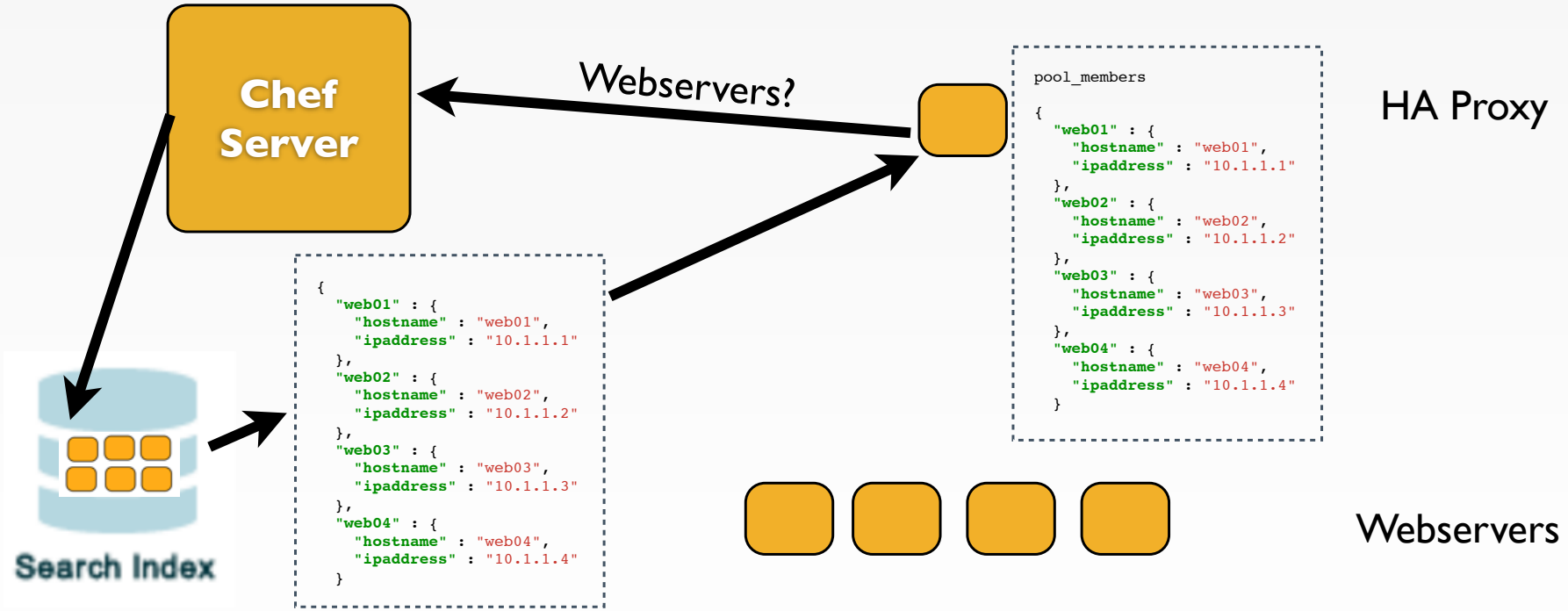
CHEF

HA Proxy Configuration



CHEF

HA Proxy Configuration

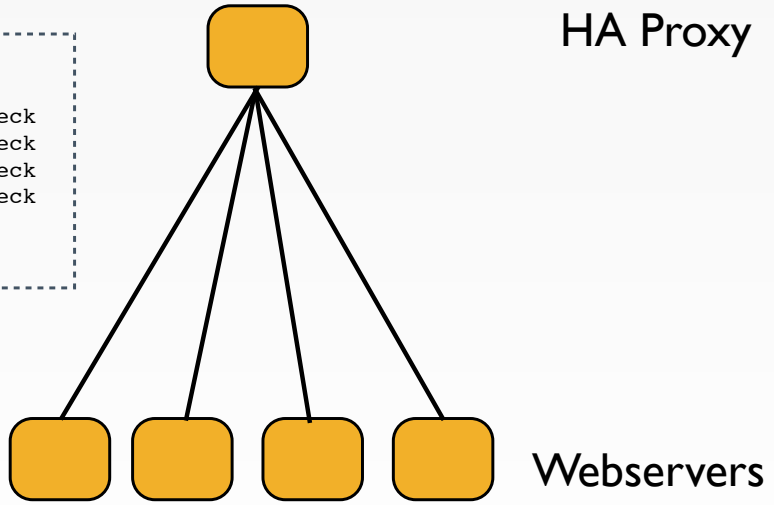


CHEF

HA Proxy Configuration

```
pool_members

{
  "web01" : {
    "hostname" : "web01",
    "ipaddress" : "10.1.1.1"
  },
  "web02" : {
    "hostname" : "web02",
    "ipaddress" : "10.1.1.2"
  },
  "web03" : {
    "hostname" : "web03",
    "ipaddress" : "10.1.1.3"
  },
  "web04" : {
    "hostname" : "web04",
    "ipaddress" : "10.1.1.4"
  }
```

```
haproxy.cfg

server web01 10.1.1.1 weight 1 maxconn 1 check
server web02 10.1.1.2 weight 1 maxconn 1 check
server web03 10.1.1.3 weight 1 maxconn 1 check
server web04 10.1.1.4 weight 1 maxconn 1 check
```



CHEF

Building your policy

Resources and Recipes

CHEF

Resources

- Piece of the system and its desired state

Resources - Package

Package that should be installed

```
package "mysql-server" do
  action :install
end
```

Resources - Service

Service that should be running and restarted on reboot

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

Resources - Service

File that should be generated

```
file "/etc/motd" do
  content "Property of Chef Software"
end
```

Resources - Cron

Cron job that should be configured

```
cron "restart webserver" do
  hour '2'
  minute '0'
  command 'service httpd restart'
end
```

Resources - User

User that should be managed

```
user "nginx" do
  comment "Nginx user <nginx@example.com>"
  uid 500
  gid 500
  supports :manage_home => true
end
```

Resources - DSC

DSC resource that should be run

```
dsc_script 'emacs' do
  code <<-EOH
  Environment 'texteditor'
  {
    Name = 'EDITOR'
    Value = 'c:\\emacs\\bin\\emacs.exe'
  }
  EOH
end
```

Resources – Registry Key

Registry key that should be created

```
registry_key "HKEY_LOCAL_MACHINE\
\SOFTWARE\\Microsoft\\Windows\
\CurrentVersion\\Policies\\System"
do
  values [{
    :name => "EnableLUA",
    :type => :dword,
    :data => 0
    }]
  action :create
end
```

Resources

- Piece of the system and its desired state

- http://docs.chef.io/chef/resources.html

Lab – What does the cow say?

- **Problem**: Our workstation is boring, I love cows
- **Success Criteria**: Carry on a conversation with Henry

CHEF

What's up with the card?

- http://bit.ly/chef-dfdc

- Login: chef
- Password: [REDACTED]

Login to your lab machine

```
$ ssh chef@54.164.75.30
```

```
The authenticity of host '54.165.227.226
(54.165.227.226)' can't be established.
RSA key fingerprint is c1:ec:ab:66:fb:22:4a:
8f:c2:c5:9b:26:77:f3:dd:b3.
Are you sure you want to continue connecting
(yes/no)? yes
Warning: Permanently added
'54.165.227.226' (RSA) to the list of known
hosts.
chef@54.165.227.226's password:
```

Welcome to your workstation

- ChefDK version 0.7.0 is installed
  - `chef --version`
- Chef user has passwordless sudo access
  - `sudo cat /etc/shadow`

chef-apply

- chef-apply is an executable program that allows you to work with resources
- Is included as part of the ChefDK
- A great way to explore resources
- NOT how you'll eventually use Chef in production

Install cowsay

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

```
Recipe: (chef-apply cookbook)::(chef-apply recipe)
  * apt_package[cowsay] action install
    - install version 3.03+dfsg1-6 of package cowsay
```

Say hello

```
$ cowsay "hello, devfest"
```

```
 _________________
< hello, devfest >
 -----------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
```

CHEF

Resources

- Describe the desired state
- Do not need to tell Chef how to get there

- What happens if you re-run the chef-apply command?

Install cowsay again with chef-apply

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

```
Recipe: (chef-apply cookbook)::(chef-apply recipe)
 * apt_package[cowsay] action install (up to date)
```

Test and Repair

Resources follow a test and repair model

```
package "vim"
```

Test and Repair

Resources follow a **test** and repair model

```
package "vim"
```

Test Is vim installed?

CHEF

Test and Repair

Resources follow a **test** and repair model

```
package "vim"
```

Test Is vim installed?

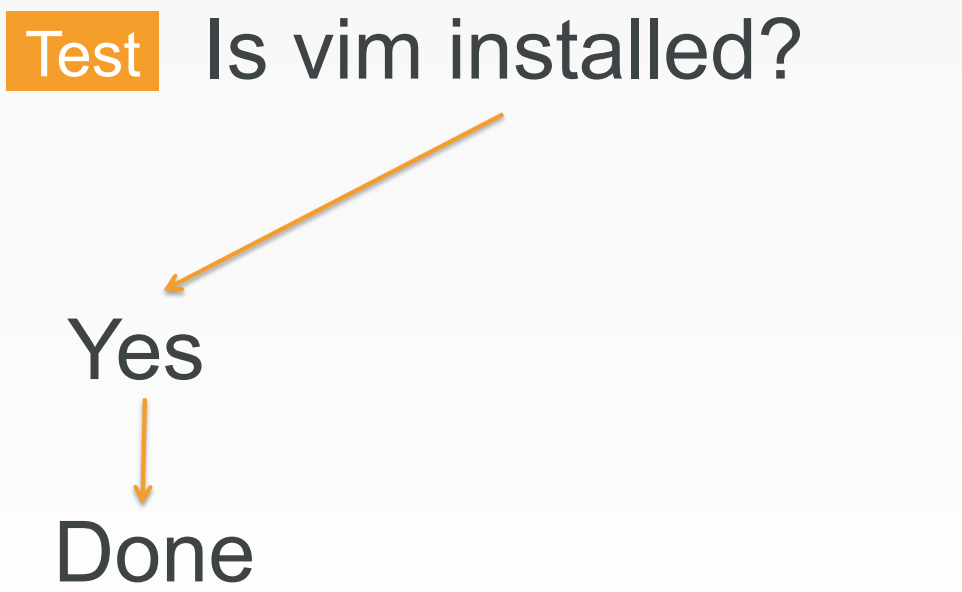
Yes

CHEF

Test and Repair

Resources follow a **test** and repair model

```
package "vim"
```

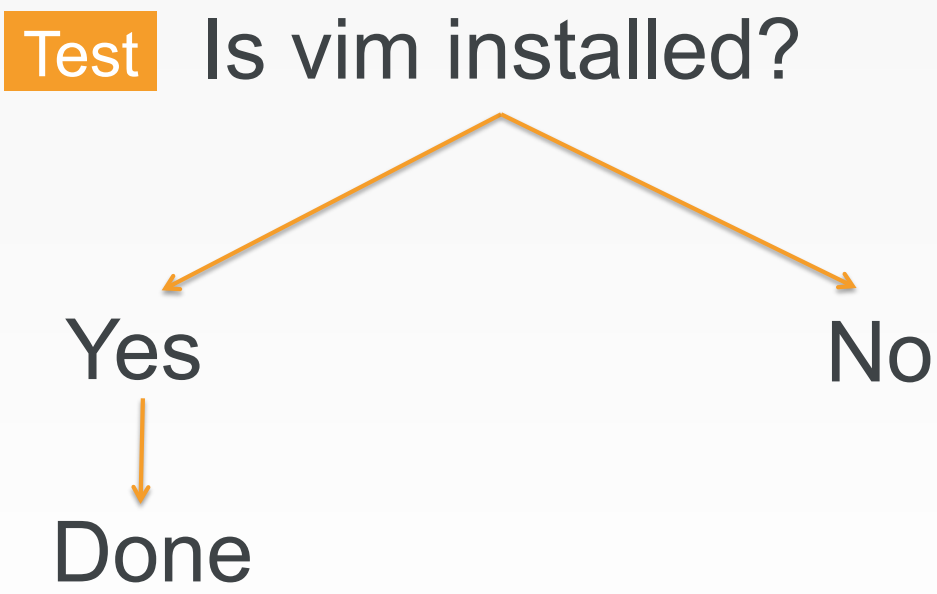


CHEF

Test and Repair

Resources follow a **test** and repair model

```
package "vim"
```

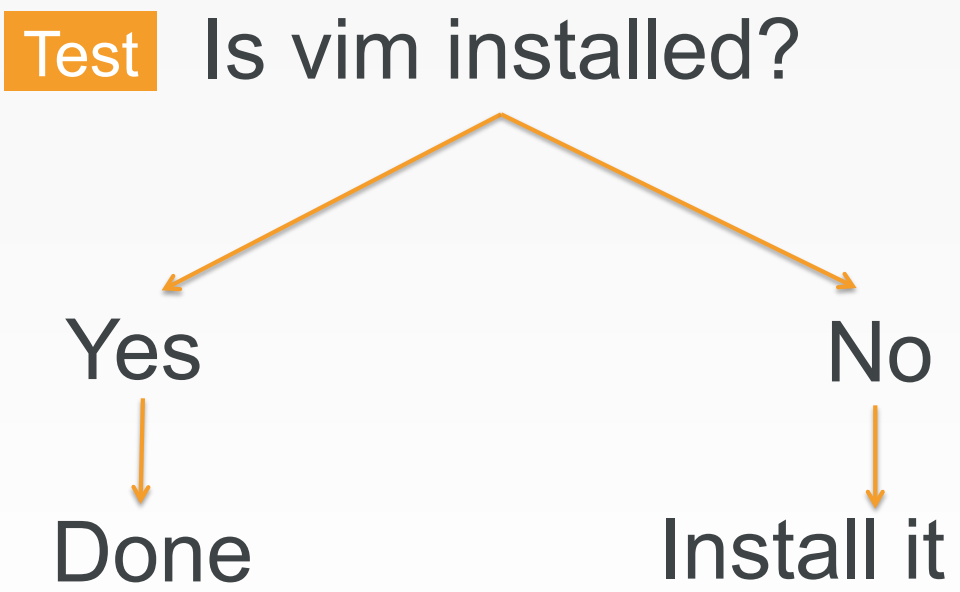


CHEF

Test and Repair

Resources follow a **test** and repair model

```
package "vim"
```

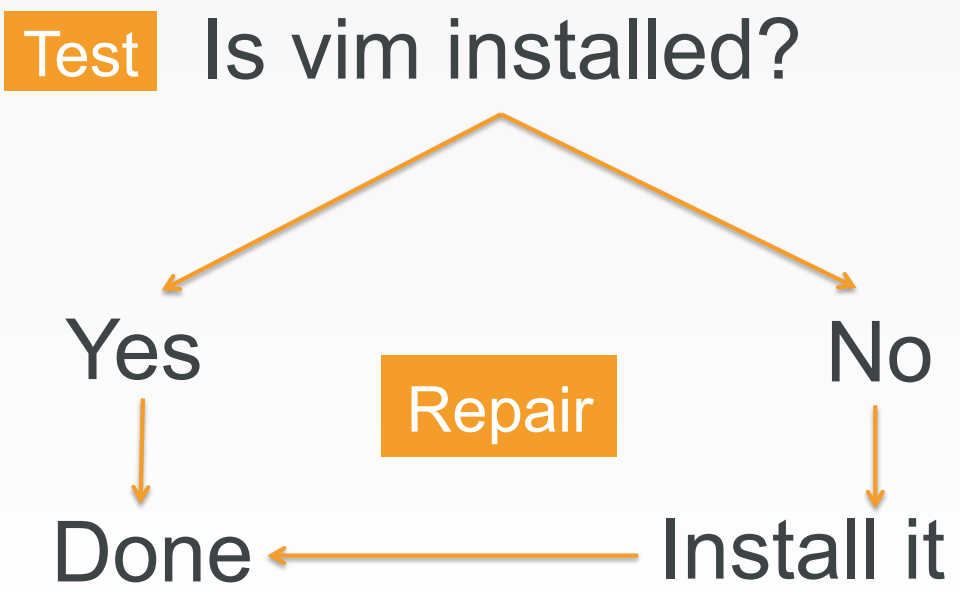


CHEF

Test and Repair

Resources follow a **test** and **repair** model

```
package "vim"
```



CHEF

Resources – Test and Repair

- Resources follow a test and repair model

- Resource currently in the desired state? (test)
  - Yes – Do nothing
  - No – Bring the resource into the desired state (repair)

Resources

- package
- template
- service
- directory
- user
- group
- dsc_script
- registry_key
- powershell_script
- cron
- mount
- route
- …and more!

Recipes

- Policy is defined as a collection of **resources** in **recipes**. There are lots of abstractions on top of this but **resources** are the basic building blocks.

Test-driven Infrastructure

Change policy with confidence

CHEF

Our process

- Write policy
- Apply policy
- Verify policy

- Not bad for the simple case, will quickly get untenable

Faster Feedback

- Speed-up the feedback loops with automated testing.
- Have confidence in your changes before you run them in production

Chef Testing

- Did chef-client complete successfully?
- Did the recipe put the node in the desired state?
- Are the resources properly defined?
- Does the code follow our style guide?

Test-driving infrastructure

- We are going to use a relatively simple scenario
- We are going to explore many facets of testing
- We are going to follow a test-first, test-driven model

Our Scenario

- We want a simple app deployed on a server.
- We will only have time to build some of it in today's workshop.
- You will leave with working code

Create an apache cookbook

```
$ chef generate cookbook apache
```

```
Compiling Cookbooks...
Recipe: code_generator::cookbook
  * directory[/home/chef/chef-repo/cookbooks/apache] action create
    - create new directory /home/chef/chef-repo/cookbooks/apache
  * template[/home/chef/chef-repo/cookbooks/apache/metadata.rb] action
create_if_missing
    - create new file /home/chef/chef-repo/cookbooks/apache/metadata.rb
    - update content in file /home/chef/chef-repo/cookbooks/apache/
metadata.rb from none to 4c0e2d
    (diff output suppressed by config)
  * template[/home/chef/chef-repo/cookbooks/apache/README.md] action
create_if_missing
```

Questions to ask when testing

- **Did chef-client complete successfully?**
- Did the recipe put the node in the desired state?
- Are the resources properly defined?
- Does the code following our style guide?

Chef client success status

- Requirements to verify chef-client success:
  - A place to store the cookbook artifact

Chef client success status

- Requirements to verify chef-client success:
  - A place to store the cookbook artifact
  - A chef-client with access to the cookbook

Chef client success status

- Requirements to verify chef-client success:
  - A place to store the cookbook artifact
  - A chef-client with access to the cookbook
  - A target server running the same OS as production

Test Kitchen

- Test harness to execute code on one or more platforms
- Driver plugins to allow your code to run on various cloud and virtualization providers
- Includes support for many testing frameworks
- Included with ChefDK

CHEF

Test Matrix

- Two operating systems

ubuntu-12.04
centos-6.4

CHEF

Test Matrix

- Two operating systems
- One recipe

	default
ubuntu-12.04	apache::default
centos-6.4	apache::default

Test Matrix

- Two operating systems
- Two recipes

	default	ssl
ubuntu-12.04	apache::default	apache::ssl
centos-6.4	apache::default	apache::ssl

Test Matrix

- Three operating systems
- Two recipes

	default	ssl
ubuntu-12.04	apache::default	apache::ssl
centos-6.4	apache::default	apache::ssl
ubuntu-14.04	apache::default	apache::ssl

Configuring the Kitchen

OPEN IN EDITOR: `cookbooks/apache/.kitchen.yml`

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-12.04
  - name: centos-6.4

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

SAVE FILE!

CHEF

.kitchen.yml

- driver - virtualization or cloud provider

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-12.04
  - name: centos-6.4

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

.kitchen.yml

- provisioner - application to configure the node

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-12.04
  - name: centos-6.4

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

.kitchen.yml

- platforms - target operating systems

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-12.04
  - name: centos-6.4

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

.kitchen.yml

- suites - target configurations

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-12.04
  - name: centos-6.4

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

.kitchen.yml

	default
ubuntu-12.04	apache::default
centos-6.4	apache::default

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-12.04
  - name: centos-6.4


suites:
  - name: default
    run_list:
      - recipe[apache::default]
```

.kitchen.yml

	default	ssl
ubuntu-12.04	apache::default	apache::ssl
centos-6.4	apache::default	apache::ssl

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-12.04
  - name: centos-6.4

suites:
  - name: default
    run_list:
      - recipe[apache::default]
  - name: ssl
    run_list:
      - recipe[apache::ssl]
```

.kitchen.yml

	default	ssl
ubuntu-12.04	apache::default	apache::ssl
centos-6.4	apache::default	apache::ssl
ubuntu-14.04	apache::default	apache::ssl

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-12.04
  - name: centos-6.4
  - name: ubuntu-14.04

suites:
  - name: default
    run_list:
      - recipe[apache::default]
  - name: ssl
    run_list:
      - recipe[apache::ssl]
```

Move to the apache cookbook directory

```
$ cd ~/apache
```

Update .kitchen.yml

OPEN IN EDITOR: `.kitchen.yml`

```
---
driver:
  name: docker
  use_sudo: false

provisioner:
  name: chef_zero

platforms:
  - name: ubuntu-14.04

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

SAVE FILE!

CHEF

List the Test Kitchens

```
$ kitchen list
```

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

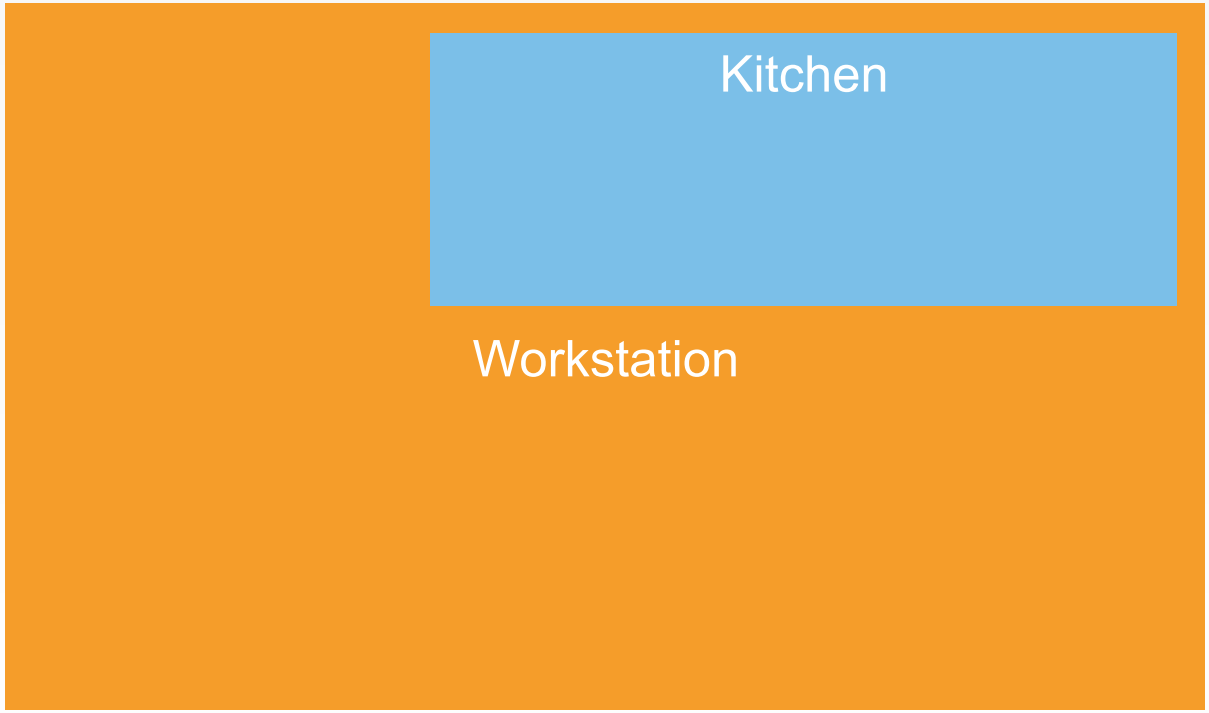
CHEF

Create the kitchen

```
$ kitchen create
```

```
-----> Starting Kitchen (v1.4.2)
-----> Creating <default-ubuntu-1404>...
       Sending build context to Docker daemon 57.34 kB
       Sending build context to Docker daemon
       Step 0 : FROM ubuntu:14.04
        ---> 91e54dfb1179
       Step 1 : RUN dpkg-divert --local --rename --add /sbin/initctl
        ---> Running in a2f331ff54cb
       Leaving 'local diversion of /sbin/initctl to /sbin/initctl.distrib'
        ---> abafd8b3362f
       Removing intermediate container a2f331ff54cb
       Step 2 : RUN ln -sf /bin/true /sbin/initctl
        ---> Running in 9973fb743c3d
```

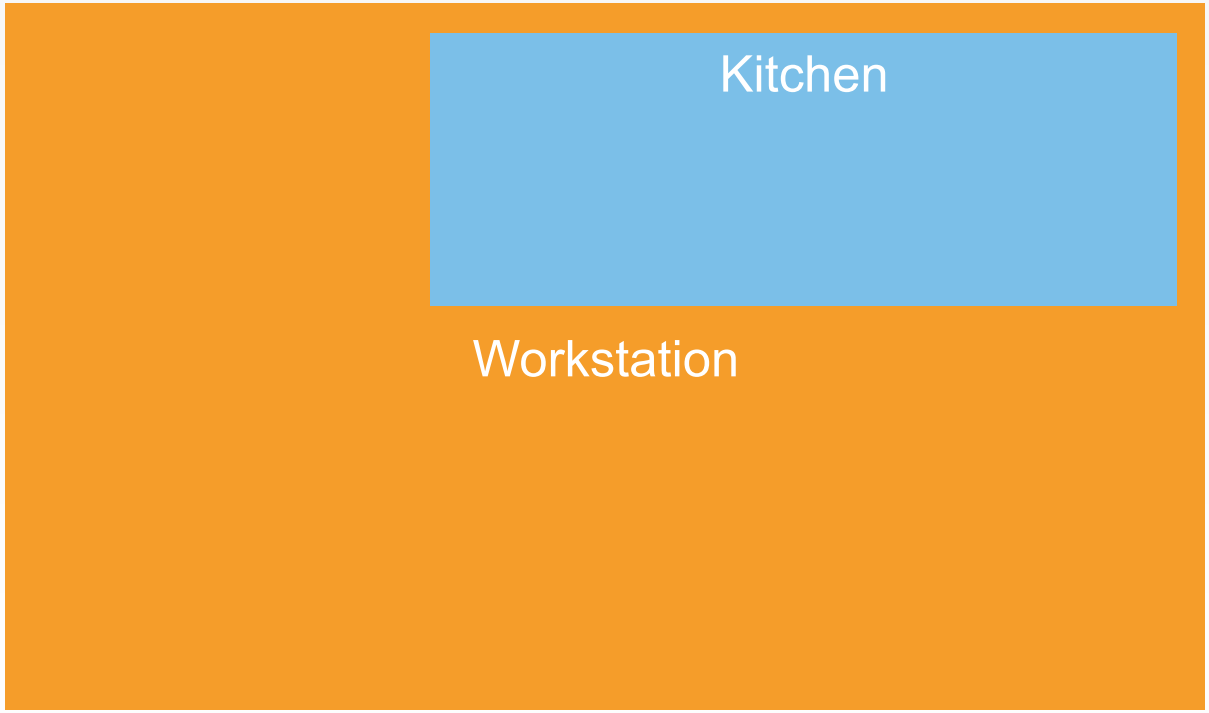
Kitchen created



Login to the kitchen

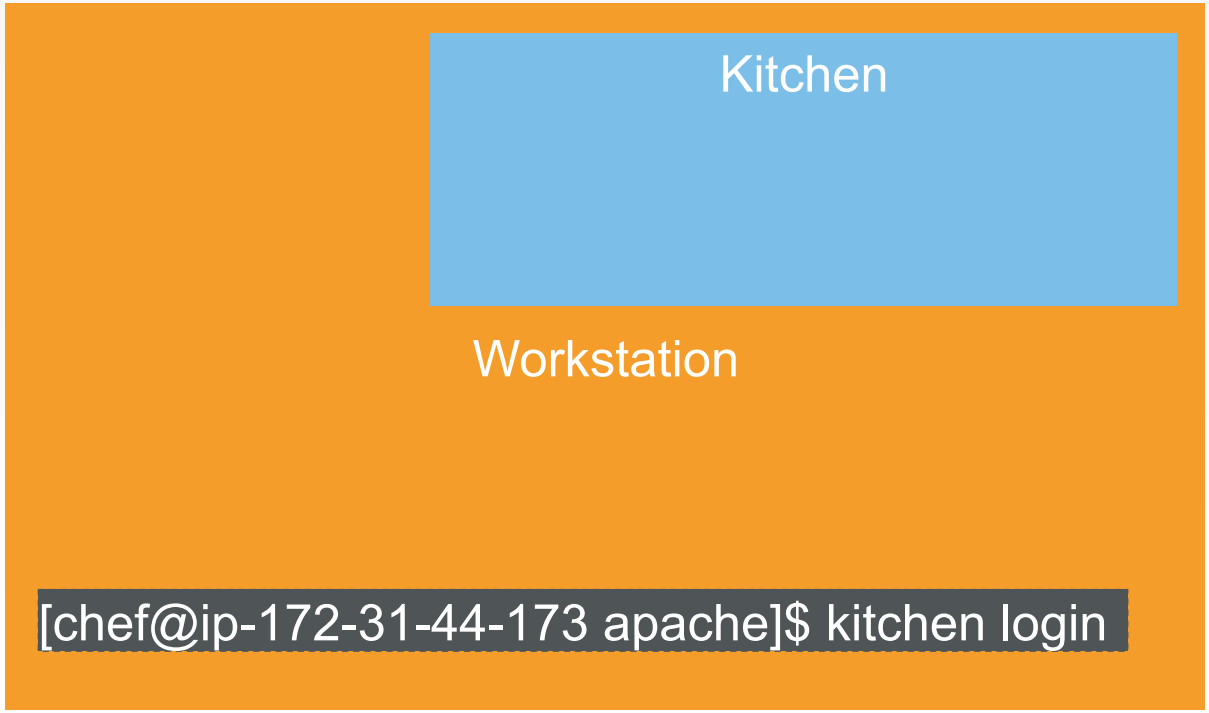
```
$ kitchen login
```

Kitchen login



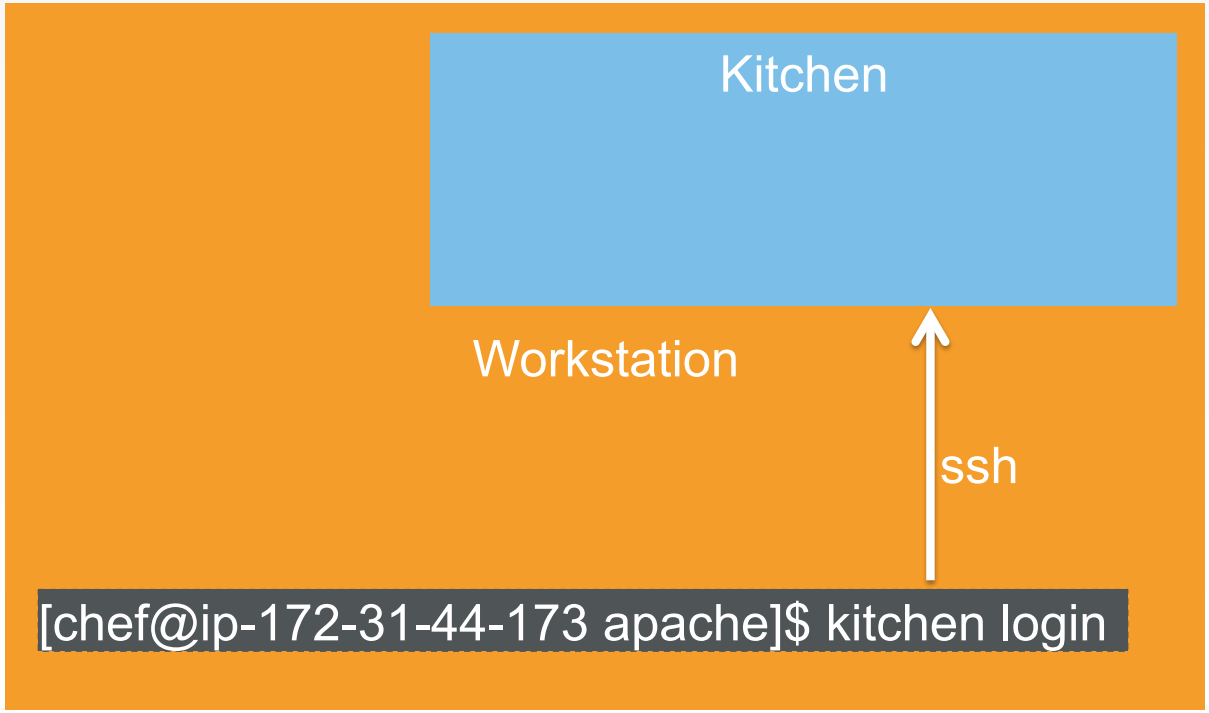
CHEF

Kitchen login



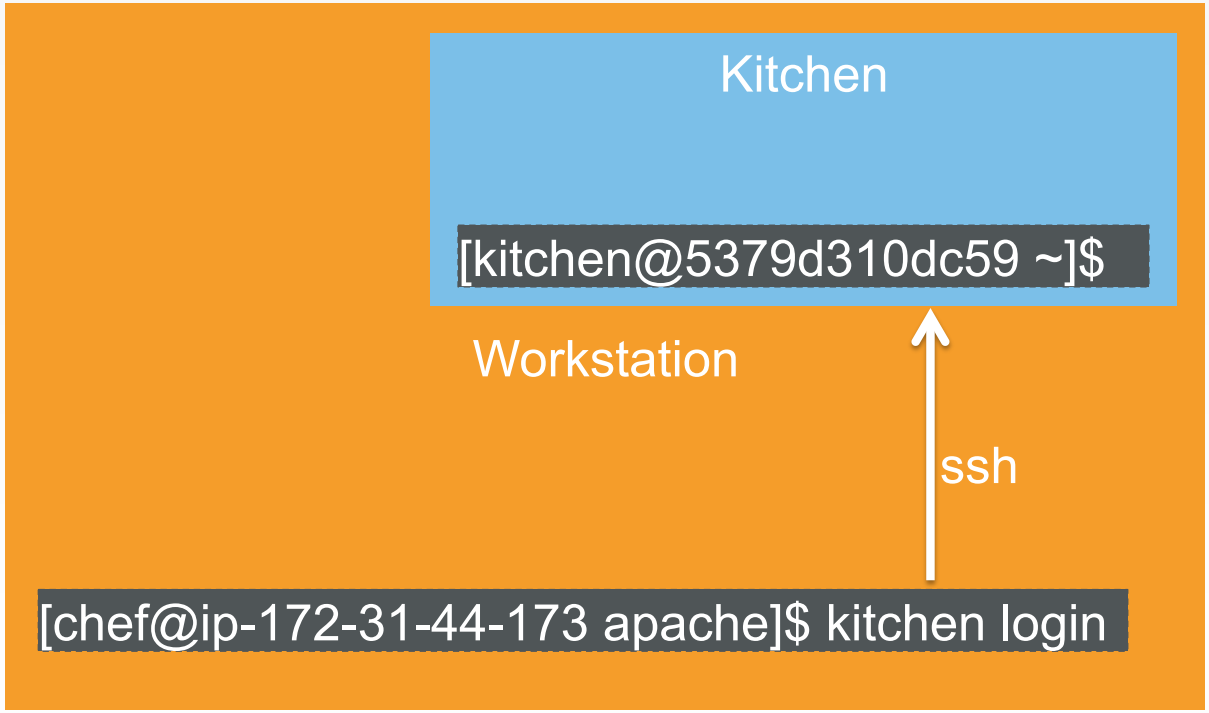
CHEF

Kitchen login



CHEF

Kitchen login



CHEF

Chef client success status

- Requirements to verify chef-client success:
  - A target server running the same OS as production
  - A chef-client with access to the cookbook

Lab – Apply our policy

- **Problem:** We have not applied our policy to the test environment.
- **Success Criteria:** The default apache recipe will be applied in the test environment

Leave the kitchen

```
$ exit
```

```
logout
Connection to localhost closed.
```

Go to the right place

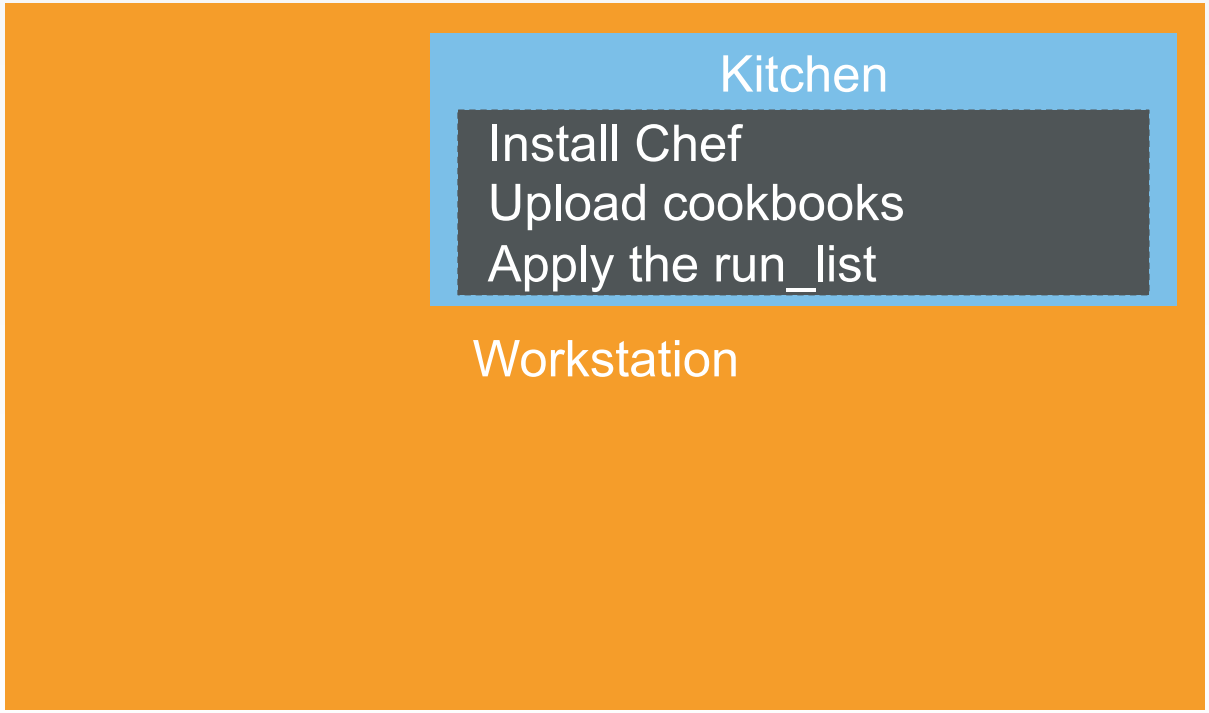
```
$ cd ~/apache
```

Apply our policy

```
$ kitchen converge
```

```
-----> Starting Kitchen (v1.4.2)
-----> Converging <default-ubuntu-1404>...
$$$$$$ Running legacy converge for 'Docker' Driver
       Preparing files for transfer
       Preparing dna.json
       Resolving cookbook dependencies with Berkshelf 3.2.4...
       Removing non-cookbook files before transfer
       Preparing validation.pem
       Preparing client.rb
-----> Installing Chef Omnibus (install only if missing)
       Downloading https://www.chef.io/chef/install.sh to file /tmp/install.sh

       Download complete.
       Downloading Chef  for ubuntu...
```

Kitchen converge



CHEF

Questions to ask when testing

- ✓ Did chef-client complete successfully?
- Did the recipe put the node in the desired state?
- Are the resources properly defined?
- Does the code following our style guide?

Verifying node state

Serverspec

CHEF

Manually inspect the test node

```
$ kitchen login
```

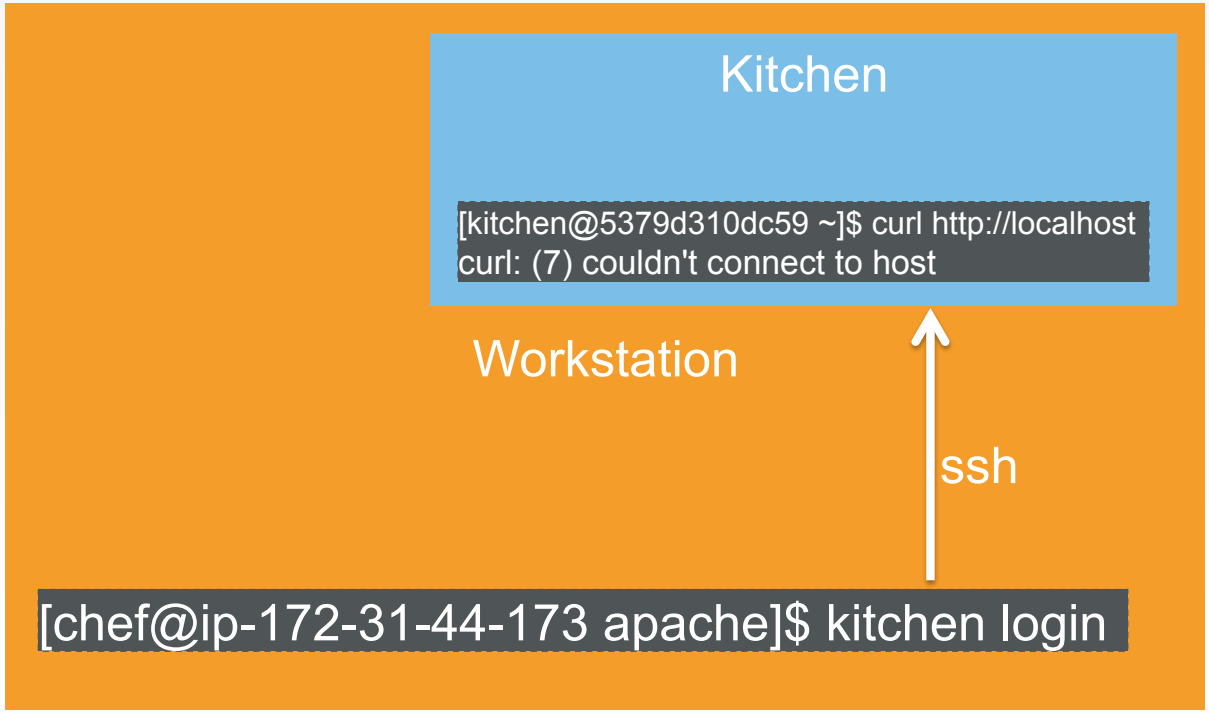
CHEF

Manually inspect the test node

```
$ curl http://localhost
```

```
curl: (7) couldn't connect to host
```

Kitchen login



CHEF

Lab – Verify node state

- **Problem**: Manually verifying the state of the test node is tedious and error-prone.
- **Success Criteria**: The end state of the node is automatically tested.

Serverspec

- Write tests to verify your servers
- Not dependent on Chef
- Defines many resource types
  - package, service, user, etc.
- Works well with Test Kitchen
- http://serverspec.org/

CHEF

Leave the Kitchen

```
$ exit
```

```
logout
Connection to localhost closed.
```

Move to the proper directory

```
$ cd ~/apache
```

Write a Serverspec test

OPEN IN EDITOR: test/integration/default/serverspec/default_spec.rb

```
require 'spec_helper'

describe 'apache::default' do

  # Serverspec examples can be found at
  # http://serverspec.org/resource_types.html

  it 'does something' do
    skip 'Replace this with meaningful tests'
  end

end
```

SAVE FILE!

CHEF

Generic Expectation Form

```
describe "<subject>" do
  it "<description>" do
    expect(thing).to eq result
  end
end
```

Awesome Expectations

OPEN IN EDITOR: test/integration/default/serverspec/default_spec.rb

```
require 'spec_helper'

describe "apache::default" do
  it "is awesome" do
    expect(true).to eq true
  end
end
```

SAVE FILE!

CHEF

Run the serverspec test

```
$ kitchen verify
```

```
-----> Running serverspec test suite
       /opt/chef/embedded/bin/ruby -I/tmp/busser/suites/serverspec -I/tmp/
busser/gems/gems/rspec-support-3.1.2/lib:/tmp/busser/gems/gems/rspec-
core-3.1.7/lib /opt/chef/embedded/bin/rspec --pattern /tmp/busser/suites/
serverspec/\*\*/\*_spec.rb --color --format documentation --default-path /
tmp/busser/suites/serverspec

       apache::default
         is awesome

       Finished in 0.02823 seconds (files took 0.99875 seconds to load)
       1 example, 0 failures
       Finished verifying <default-centos-64> (0m5.03s).
```

How would you test our criteria?

- What would you test to make sure apache is running?

Verify package is installed

OPEN IN EDITOR: test/integration/default/serverspec/default_spec.rb

```
require 'spec_helper'

describe "apache" do
  it "is awesome" do
    expect(true).to eq true
  end

  it "is installed" do
    expect(package("apache2")).to be_installed
  end
end
```

SAVE FILE!

CHEF

Exercise the test

```
$ kitchen verify
```

```
 apache::default
         is awesome
         is installed (FAILED - 1)

       Failures:

         1) apache::default is installed
            Failure/Error: expect(package 'apache2').to be_installed
              expected Package "apache2" to be installed
              /bin/sh -c dpkg-query\ -f\ \'\$\{Status\}\'\ -W\ apache2\ \|\
grep\ -E\ \'\^\(install\|hold\)\ ok\ installed\$\'
              No packages found matching apache2.
```

Test is failing, make it pass

- Test-driven development involves
  - Write a test to verify something is working
  - Watch the test fail
  - Write just enough code to make the test pass
  - Repeat

Update our cookbook

OPEN IN EDITOR: ~/apache/recipes/default.rb

```
package "apache2"
```

SAVE FILE!

CHEF

Converge the node again

```
$ kitchen converge
```

```
-----> Starting Kitchen (v1.4.2)
-----> Converging <default-ubuntu-1404>...
$$$$$$ Running legacy converge for 'Docker' Driver
       Preparing files for transfer
       Preparing dna.json
       Resolving cookbook dependencies with Berkshelf 3.2.4...
       Removing non-cookbook files before transfer
       Preparing validation.pem
       Preparing client.rb
-----> Chef Omnibus installation detected (install only if missing)
       Transferring files to <default-ubuntu-1404>
       Starting Chef Client, version 12.4.1
       [2015-09-11T04:40:21+00:00] WARN: Child with name 'dna.json' found in multiple directories: /tmp/kitchen/dna.json and /tmp/kitchen/
dna.json
       resolving cookbooks for run list: ["apache::default"]
       Synchronizing Cookbooks:
         - apache
       Compiling Cookbooks...
       Converging 1 resources
```

Exercise the test

```
$ kitchen verify
```

```
 apache
         is awesome
         is installed

       Finished in 0.48165 seconds (files took 1.05
seconds to load)
       2 examples, 0 failures
       Finished verifying <default-centos-64>
(0m5.64s).
-----> Kitchen is finished. (0m11.84s)
```

What else will you test?

- Is the service running?
- Is the port accessible?
- Is the expected content being served?

- Make sure everything works from a fresh kitchen, too!

Extend the Serverspec test

OPEN IN EDITOR: test/integration/default/serverspec/default_spec.rb

```
describe 'apache' do
  it "is installed" do
    expect(package 'apache2').to be_installed
  end

  it "is running" do
    expect(service 'apache2').to be_running
  end

  it "responds to http requests" do
    expect(command("curl http://localhost").exit_status).to eq 0
  end
end
```

SAVE FILE!

CHEF

Verify the kitchen

```
$ kitchen verify
```

```
Failures:

  1) apache::default is running
     Failure/Error: expect(service 'apache2').to be_running
     expected Service "apache2" to be running
    /bin/sh -c service\ apache2\ status\ \&\&\ service\ apache2\ status\ \|\ grep\
\'running\'
               Apache2 is NOT running.

  2) apache::default responds to http requests
     Failure/Error: expect(command("curl localhost").exit_status).to eq 0

     expected: 0
     got: 7
```

Update our cookbook

OPEN IN EDITOR: ~/chef-repo/cookbooks/apache/recipes/default.rb

```
package 'apache2'

service 'apache2' do
  action :start
end
```

SAVE FILE!

CHEF

Converge the node again

```
$ kitchen converge
```

```
-----> Starting Kitchen (v1.4.2)
-----> Converging <default-ubuntu-1404>...
$$$$$$ Running legacy converge for 'Docker' Driver
       Preparing files for transfer
       Preparing dna.json
       Resolving cookbook dependencies with Berkshelf 3.2.4...
       Removing non-cookbook files before transfer
       Preparing validation.pem
       Preparing client.rb
-----> Chef Omnibus installation detected (install only if missing)
       Transferring files to <default-ubuntu-1404>
       Starting Chef Client, version 12.4.1
       [2015-09-11T04:42:17+00:00] WARN: Child with name 'dna.json' found in multiple directories: /tmp/kitchen/dna.json and /tmp/kitchen/
dna.json
       resolving cookbooks for run list: ["apache::default"]
       Synchronizing Cookbooks:
         - apache
       Compiling Cookbooks...
       Converging 2 resources
       Recipe: apache::default
```

Verify the kitchen

```
$ kitchen verify
```

```
 apache::default
          is awesome
          is installed
          is running
          responds to http requests

        Finished in 0.9764 seconds (files took 1.22 seconds to
load)
        4 examples, 0 failures

        Finished verifying <default-ubuntu-1204> (0m9.78s).
-----> Kitchen is finished. (0m11.23s)
```

Kitchen Workflow

- `kitchen create`
- `kitchen converge`
- `kitchen verify`
- `kitchen destroy`

- All at once with `kitchen test`

Chef Testing

- ✓ Did chef-client complete successfully?
- ✓ Did the recipe put the node in the desired state?
- Are the resources properly defined?
- Does the code following our style guide?

Time Check! What's Next?

- ChefSpec
- Foodcritic
- Install a simple application
- Make your apache cookbook work on CentOS
- Introduction to Audit Mode
  - Test for Shell Shock
- Wrap-up

Even Faster Feedback

ChefSpec

CHEF

Chef Testing

- ✓ Did chef-client complete successfully?
- ✓ Did the recipe put the node in the desired state?
- Are the resources properly defined?
- Does the code following our style guide?

This is too slow!

- To test our code, we need to spin up a test kitchen, converge a node, execute some tests.
- Our simple test case takes about 2 minutes to fully execute.

Properly configured resources

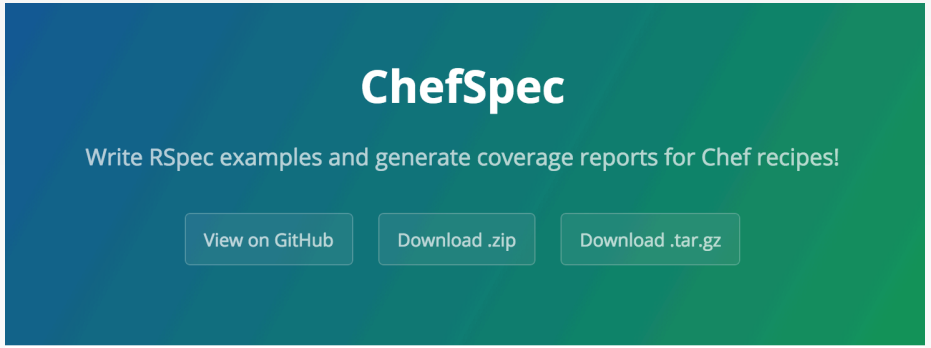
- We need a way to verify that the resources in our recipes are properly configured
- We want to get faster feedback

Lab – Verify the resources

- **Problem**: We should be able to catch errors before we need to converge a node
- **Success Criteria**: Catch a typo prior to converge

ChefSpec

- Test before you converge
- Get feedback on cookbook changes without the need for target servers



ChefSpec

gem v4.2.0 build passing

ChefSpec is a unit testing framework for testing Chef cookbooks. ChefSpec makes it easy to write examples and get fast feedback on cookbook changes without the need for virtual machines or cloud servers.

ChefSpec runs your cookbook(s) locally with Chef Solo without actually converging a node. This has two primary benefits:

- It's really fast!
- Your tests can vary node attributes, operating systems, and search results to assert behavior under varying conditions.

http://sethvargo.github.io/chefspec/

CHEF

Change to the apache cookbook directory

```
$ cd ~/apache
```

Write a ChefSpec test

OPEN IN EDITOR: spec/unit/recipes/default_spec.rb

```
require 'spec_helper'

describe 'apache::default' do

  context 'When all attributes are default, on an unspecified platform' do

    let(:chef_run) do
      runner = ChefSpec::ServerRunner.new
      runner.converge(described_recipe)
    end

    it 'converges successfully' do
      expect { chef_run }.to_not raise_error
    end

  end
end
```

SAVE FILE!

CHEF

Write a ChefSpec test

OPEN IN EDITOR: spec/unit/recipes/default_spec.rb

```
describe 'apache::default' do
  context 'When all attributes are default, on an unspecified platform' do

    let(:chef_run) do
      runner = ChefSpec::ServerRunner.new
      runner.converge(described_recipe)
    end

    it 'converges successfully' do
      chef_run # This should not raise an error
    end

    it 'installs apache' do
      expect(chef_run).to install_package 'apache2'
    end
  end
end
```

SAVE FILE!

CHEF

Run the ChefSpec tests

```
$ rspec spec -fd -c
```

```
apache::default
  When all attributes are default, on an unspecified platform
    converges successfully
    installs apache

Finished in 0.36297 seconds (files took 2.75 seconds to load)
2 examples, 0 failures
```

Break the cookbook

OPEN IN EDITOR: recipes/default.rb

```
package "apache"

service "apache2" do
  action :start
end
```

SAVE FILE!

CHEF

Run the ChefSpec tests

```
$ rspec spec -fd -c
```

```
apache::default
  When all attributes are default, on an unspecified platform
    converges successfully
    installs apache (FAILED - 1)

Failures:

  1) apache::default When all attributes are default, on an unspecified platform installs apache
     Failure/Error: expect(chef_run).to install_package 'apache2'
       expected "package[apache2]" with action :install to be in Chef run. Other package resources:

         package[apache]

     # ./spec/unit/recipes/default_spec.rb:21:in `block (3 levels) in <top (required)>'

Finished in 0.35601 seconds (files took 2.52 seconds to load)
2 examples, 1 failure
```

Fix the cookbook

OPEN IN EDITOR: recipes/default.rb

```
package "apache2"

service "apache2" do
  action :start
end
```

SAVE FILE!

CHEF

Run the ChefSpec tests

```
$ rspec spec -fd -c
```

```
apache::default
  When all attributes are default, on an unspecified platform
    converges successfully
    installs apache

Finished in 0.35496 seconds (files took 2.53 seconds to load)
2 examples, 0 failures
```

Chef Testing

- ✓ Did chef-client complete successfully?
- ✓ Did the recipe put the node in the desired state?
- ✓ Are the resources properly defined?
- Does the code following our style guide?

Time Check! What's Next?

- ChefSpec
- Foodcritic
- Install a simple application
- Make your apache cookbook work on CentOS
- Introduction to Audit Mode
  - Test for Shell Shock
- Wrap-up

Clean code

Follow best practices, avoid mistakes

CHEF

Foodcritic

- Check cookbooks for common problems
- Style, correctness, deprecations, etc.
- Included with ChefDK

http://www.foodcritic.io/

CHEF

Change our recipe

OPEN IN EDITOR: recipes/default.rb

```
package_name = "apache2"

package "#{package_name}"

service "apache2" do
  action :start
end
```

SAVE FILE!

CHEF

Run Foodcritic

```
$ foodcritic .
```

```
FC002: Avoid string interpolation
where not required: ./recipes/
default.rb:8
```

Chef Testing

- ✓ Did chef-client complete successfully?
- ✓ Did the recipe put the node in the desired state?
- ✓ Are the resources properly defined?
- ✓ Does the code following our style guide?

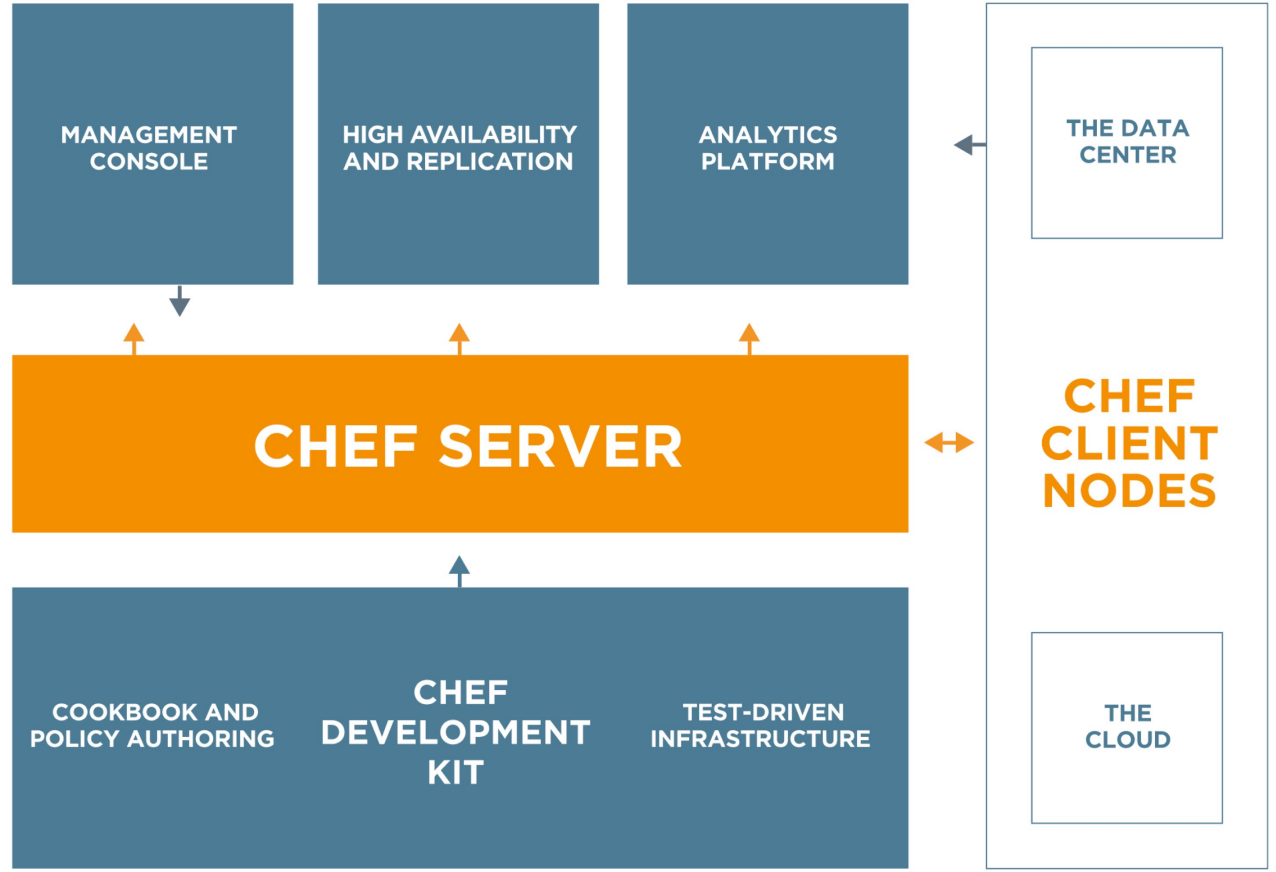
Time Check! What's Next?

- ChefSpec
- Foodcritic
- Install a simple application
- Make your apache cookbook work on CentOS
- Introduction to Audit Mode
  - Test for Shell Shock
- Wrap-up

Wrapping Up

CHEF

We've only scratched the surface



https://www.chef.io/chef/

CHEF

Build Anything

- Simple internal applications
- Complex external applications
- Workstations
- Hadoop clusters
- IaaS infrastructure
- PaaS infrastructure
- SaaS applications
- Storage systems
- You name it

http://www.flickr.com/photos/hyku/245010680/

CHEF

And Manage it Simply

http://www.flickr.com/photos/helico/404640681/

- Automatically reconfigure everything
- Linux, Windows, Unixes, BSDs
- Load balancers
- Metrics collection systems
- Monitoring systems
- Cloud migrations become trivial

CHEF

What questions do you have?

- Ask me anything!
- @nathenharvey
- Thank you!

Time Check! What's Next?

- ChefSpec
- Foodcritic
- Install a simple application
- Make your apache cookbook work on CentOS
- Introduction to Audit Mode
  - Test for Shell Shock
- Wrap-up

Install a Simple Application

- Modify `~/cookbooks/aar/recipes/default.rb`
- `cd ~/`
- `sudo chef-client -z -r 'recipe[aar]'`
- Open your workstation in a browser

CHEF

Time Check! What's Next?

- ChefSpec
- Foodcritic
- Install a simple application
- Make your apache cookbook work on CentOS
- Introduction to Audit Mode
  - Test for Shell Shock
- Wrap-up

Make your apache cookbook work on CentOS

- Add centos-7.1 to your .kitchen.yml
- Apache package name on CentOS – ‘httpd’
- Apache service name on CentOS – ‘httpd’

Time Check! What's Next?

- ChefSpec
- Foodcritic
- Install a simple application
- Make your apache cookbook work on CentOS
- Introduction to Audit Mode
  - Test for Shell Shock
- Wrap-up

Audit Shell Shock

```
$ cd ~/cookbooks/audit-shell-shock
```

CHEF

Audit Shell Shock

```
$ kitchen converge
```

```
 Shell Shock
        CVE-2014-6271
          is remediated
        CVE-2014-6277
          is remediated (FAILED - 1)
        CVE-2014-6278
          is remediated
        CVE-2014-7169
          is remediated
        CVE-2014-7186
          is remediated
        CVE-2014-7187
          is remediated


      Failures:


        1) Shell Shock CVE-2014-6277 is remediated
           Failure/Error: expect(command("bash -c \"f() { x() { _;}; x() { _;} <<a; }\" 2>/dev/null || echo vulnerable").stdout).not_to
contain(/vulnerable/)
      expected "vulnerable\n" not to contain /vulnerable/
           # /tmp/kitchen/cache/cookbooks/audit-shell-shock/recipes/default.rb:15:in `block (3 levels) in from_file'
```