Chef Provisioning

Chef Users London Meetup

Nathen Harvey - @nathenharvey

https://github.com/nathenharvey/chef-london-meetup-feb-2015

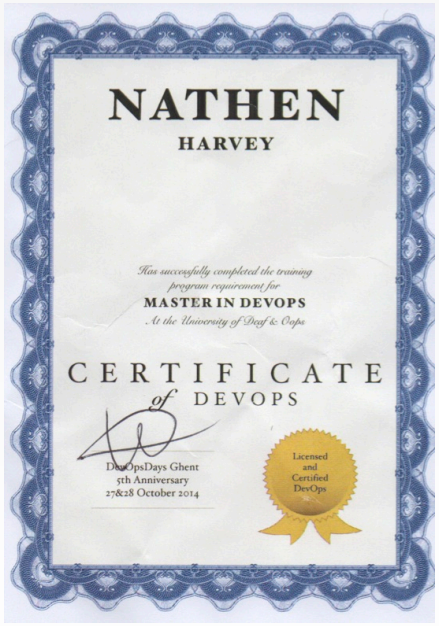
Nathen Harvey

- nharvey@chef.io
- @nathenharvey

CHEF

Nathen Harvey

- nharvey@chef.io
- @nathenharvey



CHEF

Provisioning with Chef

Chef Lifecycle

1. Provision a server, virtual machine, or cloud instance
2. Install Chef
3. Configure Chef
4. Run Chef
5. GOTO step 4

knife bootstrap

1. Provision a server, virtual machine, or cloud instance
2. Install Chef
3. Configure Chef
4. Run Chef
5. GOTO step 4

Provisioning in AWS

- Let's use AWS as our case study
- We'll look at the various ways Chef can help you provision instances there

```
$ knife ec2 server create -I ami-4ab46b3d -f t1.micro -g sg-884f2eed,sg-8b4f2eee --ssh-user ubuntu -N nathen_hw_knife -r "recipe[hello_world]"
```

```
ec2-54-77-67-44.eu-west-1.compute.amazonaws.com Chef Client finished, 85/149 resources updated in 38.998005693 seconds

Instance ID: i-80d76866
Flavor: t1.micro
Image: ami-4ab46b3d
Region: eu-west-1
Availability Zone: eu-west-1a
Security Groups: default
Security Group Ids: sg-884f2eed, sg-8b4f2eee
Tags: Name: nathen_hw_knife
SSH Key: chef-nathenharvey-eu
Root Device Type: ebs
Root Volume ID: vol-602e1f7c
Root Device Name: /dev/sda1
Root Device Delete on Terminate: true
Public DNS Name: ec2-54-77-67-44.eu-west-1.compute.amazonaws.com
Public IP Address: 54.77.67.44
Private DNS Name: ip-172-31-2-160.eu-west-1.compute.internal
Private IP Address: 172.31.2.160
Environment: _default
Run List: recipe[hello_world]
```

knife ec2 server create

CHEF

knife ec2

- “Infrastructure as Command Line” instead of “Infrastructure as Code”
- What if I want many instances?
- What if I want multiple tiers of instances?

https://flic.kr/p/eycPj7

The classic operations playbook

```
  role:monitoring*" "sudo chef-client"
```

rabbitmq (omnibus)

```
## knife Chef 0.10 syntax
knife ec2 server create -G rs-preprod,platform-server --flavor m1.small -I ami-1136fb78 -x ubuntu \
  -E rs-preprod \
  -r 'role[rabbitmq],role[rs-preprod]' -i ~/.ssh/our_ssh_key
```

Create and attach an elastic IP, or find the old rabbitmq elastic IP and attach it.
Rerun chef-client
update environment data bag item "rabbitmq_ip" with assigned elastic IP, upload data bag with knife.

run chef-client on all the servers that are rabbitmq producers or consumers:

```
sleep 90 && knife ssh "role:opscode-account OR role:opscode-chef OR role:opscode-solr OR \
  role:opscode-job-worker OR role:monitoring*" "sudo chef-client"
```

opscode-solr

```
## knife Chef 0.10 syntax
knife ec2 server create -G platform-server,rs-preprod \
  --flavor c1.medium -I ami-1136fb78  -x ubuntu \
  -E rs-preprod \
  -r 'role[opscode-solr],role[rs-preprod]' \
  -Z us-east-1c \
  -i ~/.ssh/our_ssh_key
```

Attach an elastic IP.
Rerun chef-client
update environment data bag item "solr_ip" with assigned elastic IP, upload data bag with knife.

then add it into rotation after waiting for node to be indexed

```
sleep 90 && knife ssh "role:opscode-lb-int OR role:monitoring*" "sudo chef-client"
```

opscode-authz (omnibus)

```
knife ec2 server create -G rs-preprod,platform-server --flavor c1.medium -I ami-1136fb78 -x ubuntu \
```

CHEF

Spiceweasel

- Command-line tool for batch loading Chef infrastructure
- Generate and executes knife commands

```
nodes:
- serverA:
    run_list: role[base]
    options: -i ~/.ssh/mray.pem -x user --sudo
- serverB serverC:
    run_list: role[base]
    options: -i ~/.ssh/mray.pem -x user --sudo -E production
- rackspace 3:
    run_list: recipe[mysql],role[monitoring]
    options: --image 49 --flavor 2 -N db{{n}}
- windows_winrm winboxA:
    run_list: role[base],role[iisserver]
    options: -x Administrator -P 'super_secret_password'
- windows_ssh winboxB winboxC:
    run_list: role[base],role[iisserver]
    options: -x Administrator -P 'super_secret_password'
```

https://github.com/mattray/spiceweasel

Chef Provisioning

- Allows creation of instances in Chef Recipes
- Allows for more programmatic creation
- Allows for multiple tiers to be created in one shot
- Moves more towards “Infrastructure as Code”

https://flic.kr/p/knDPjc

CHEF

Chef Provisioning

```
machine 'web1' do
  recipe 'apache'
end
```

Chef Provisioning - Drivers

- AWS
- Azure
- Fog
- Vagrant
- Docker
- LXC
- Hanlon
- …and more

```
machine 'web1' do
  recipe 'apache'
end
```

The classic operations playbook

```
  role:monitoring*" "sudo chef-client"
```

rabbitmq (omnibus)

```
## knife Chef 0.10 syntax
knife ec2 server create -G rs-preprod,platform-server --flavor m1.small -I ami-1136fb78 -x ubuntu \
  -E rs-preprod \
  -r 'role[rabbitmq],role[rs-preprod]' -i ~/.ssh/our_ssh_key
```

Create and attach an elastic IP, or find the old rabbitmq elastic IP and attach it.
Rerun chef-client
update environment data bag item "rabbitmq_ip" with assigned elastic IP, upload data bag with knife.

run chef-client on all the servers that are rabbitmq producers or consumers:

```
sleep 90 && knife ssh "role:opscode-account OR role:opscode-chef OR role:opscode-solr OR \
  role:opscode-job-worker OR role:monitoring*" "sudo chef-client"
```

opscode-solr

```
## knife Chef 0.10 syntax
knife ec2 server create -G platform-server,rs-preprod \
  --flavor c1.medium -I ami-1136fb78  -x ubuntu \
  -E rs-preprod \
  -r 'role[opscode-solr],role[rs-preprod]' \
  -Z us-east-1c \
  -i ~/.ssh/our_ssh_key
```

Attach an elastic IP.
Rerun chef-client
update environment data bag item "solr_ip" with assigned elastic IP, upload data bag with knife.

then add it into rotation after waiting for node to be indexed

```
sleep 90 && knife ssh "role:opscode-lb-int OR role:monitoring*" "sudo chef-client"
```

opscode-authz (omnibus)

```
knife ec2 server create -G rs-preprod,platform-server --flavor c1.medium -I ami-1136fb78 -x ubuntu \
```

https://flic.kr/p/eycPj7

CHEF

Chef Provisioning

```
require 'chef/provisioning'

machine_batch do
  machines %w(primary secondary web1 web2)
end

machine_batch do
  machine 'primary' do
    recipe 'initial_ha_setup'
  end
end

machine_batch do
  machine 'secondary' do
    recipe 'initial_ha_setup'
  end
end

machine_batch do
  %w(primary secondary).each do |name|
    machine name do
      recipe 'rest_of_my_configuration'
    end
  end
end
```

Configure Chef Provisioning

- Installed as part of Chef Development Kit
- Configure AWS
  - Create an AWS credentials file

- Gems
  - chef-provisioning
  - chef-provisioning-aws

AWS Credentials File

OPEN IN EDITOR: `~/.aws/config`

```
[default]
region=eu-west-1
aws_access_key_id = <AWS_ACCESS_KEY_ID>
aws_secret_access_key = <AWS_SECRET_ACCESS_KEY>
```

SAVE FILE!

CHEF

Simple Example

```
require 'chef/provisioning/aws_driver'
with_driver 'aws'

machine 'nathen_web1' do
  recipe 'hello_world'
end
```

Execute the Recipe

```
$ chef-client --local-mode simple.rb
```

```
[2015-02-04T12:53:22+00:00] INFO: Started chef-zero at http://localhost:8889 with repository at /Users/nathenharvey/chef_london_users/chef-repo
  One version per cookbook

[2015-02-04T12:53:22+00:00] INFO: Forking chef instance to converge...
Starting Chef Client, version 12.0.3
[2015-02-04T12:53:22+00:00] INFO: *** Chef 12.0.3 ***
[2015-02-04T12:53:22+00:00] INFO: Chef-client pid: 23233
[2015-02-04T12:53:31+00:00] INFO: Run List is []
[2015-02-04T12:53:31+00:00] INFO: Run List expands to []
[2015-02-04T12:53:31+00:00] INFO: Starting Chef Run for nharveycul215
[2015-02-04T12:53:31+00:00] INFO: Running start handlers
[2015-02-04T12:53:31+00:00] INFO: Start handlers complete.
[2015-02-04T12:53:31+00:00] INFO: HTTP Request Returned 404 Not Found : Object not found: /reports/nodes/nharveycul215/runs
resolving cookbooks for run list: []
[2015-02-04T12:53:31+00:00] INFO: Loading cookbooks []
Synchronizing Cookbooks:
Compiling Cookbooks...
[2015-02-04T12:53:35+00:00] WARN: Node nharveycul215 has an empty run list.
Converging 7 resources
Recipe: @recipe_files::/Users/nathenharvey/chef_london_users/chef-repo/web.rb
```

Test and Repair

Resources follow a test and repair model

```
machine 'nathen_web1'
```

Test and Repair

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

```
machine 'nathen_web1'
```

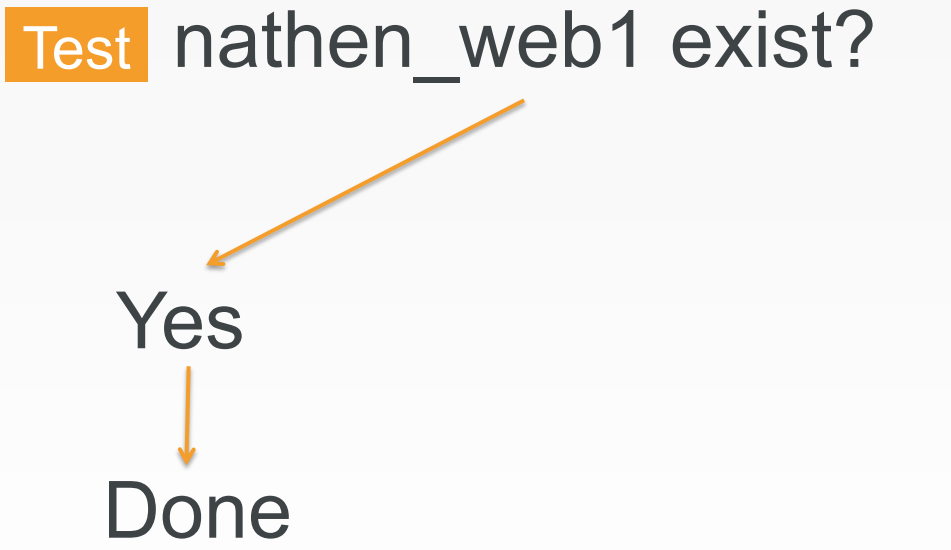
Test nathen_web1 exist?

Yes

Test and Repair

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

```
machine 'nathen_web1'
```

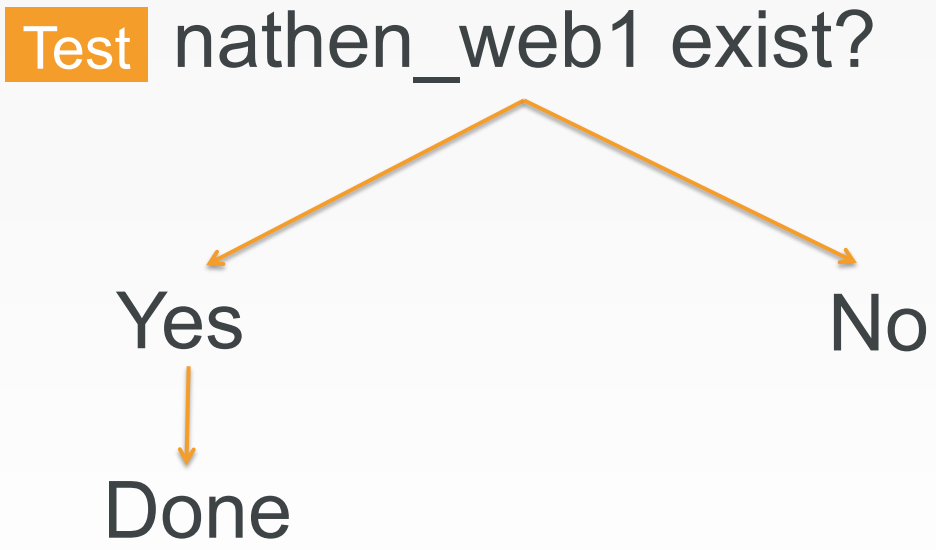


CHEF

Test and Repair

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

```
machine 'nathen_web1'
```

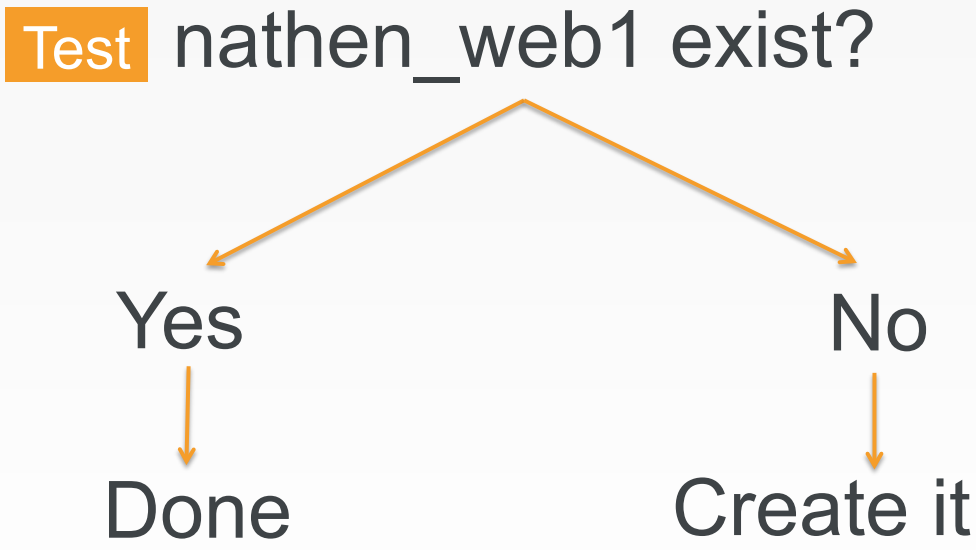


CHEF

Test and Repair

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

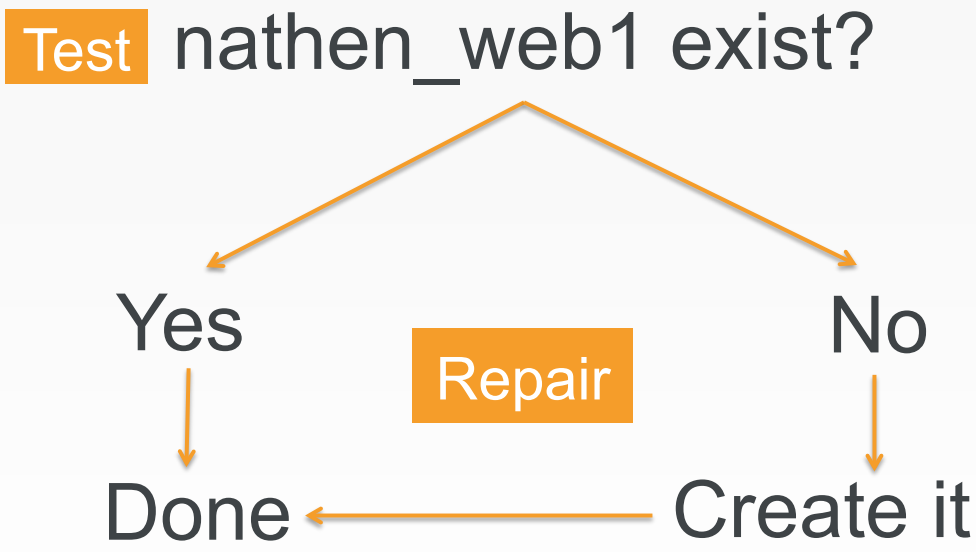
```
machine 'nathen_web1'
```



Test and Repair

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

```
machine 'nathen_web1'
```



CHEF

Behold! The power of a loop

- Need multiple instances?
- That's easy!

```
num_webservers = 3
(0...num_webservers).each do |i|
  machine "nathen_web_0#{i}" do
    recipe "hello_world"
  end
end
```

More with AWS

- Security Groups
- Elastic Load Balancers
- VPCs
- Auto Scaling Groups
- SQS Queues
- …and more
- https://github.com/chef/chef-provisioning-aws

Security Group

```
with_data_center 'eu-west-1' do
  aws_security_group "nathen-provisioning-security-group" do
    inbound_rules [
      {:ports => 22, :protocol => :tcp, :sources => ["0.0.0.0/0"] },
      {:ports => 80, :protocol => :tcp, :sources => ["0.0.0.0/0"] }
    ]
  end
end
```

Security Groups

```
{
  "id": "nathen-provisioning-security-group",
  "security_group_id": "sg-1d452878"
}
```

- Information is stored in data bag items

Load Security Group ID

```
security_group = data_bag_item('aws_security_groups',
                               'nathen-provisioning-security-group')

security_group_id = security_group['security_group_id']
```

Use the Security Group

```
with_machine_options :bootstrap_options => {
  :instance_type => 't1.micro',
  :security_group_ids => [security_group_id]
}
```

Elastic Load Balancer

```
load_balancer "nathen-elb" do
  load_balancer_options :availability_zones => ['eu-
west-1a', 'eu-west-1b', 'eu-west-1c'],
                        :listeners => [{
                             :port => 80,
                             :protocol => :http,
                             :instance_port => 80,
                             :instance_protocol => :http,
                         }]
  machines machine_names
end
```

Provisioning Production Environments

Chef Provisioning – Provisioning Node

Chef Server

AWS

Chef Provisioning – Provisioning Node

Chef Server

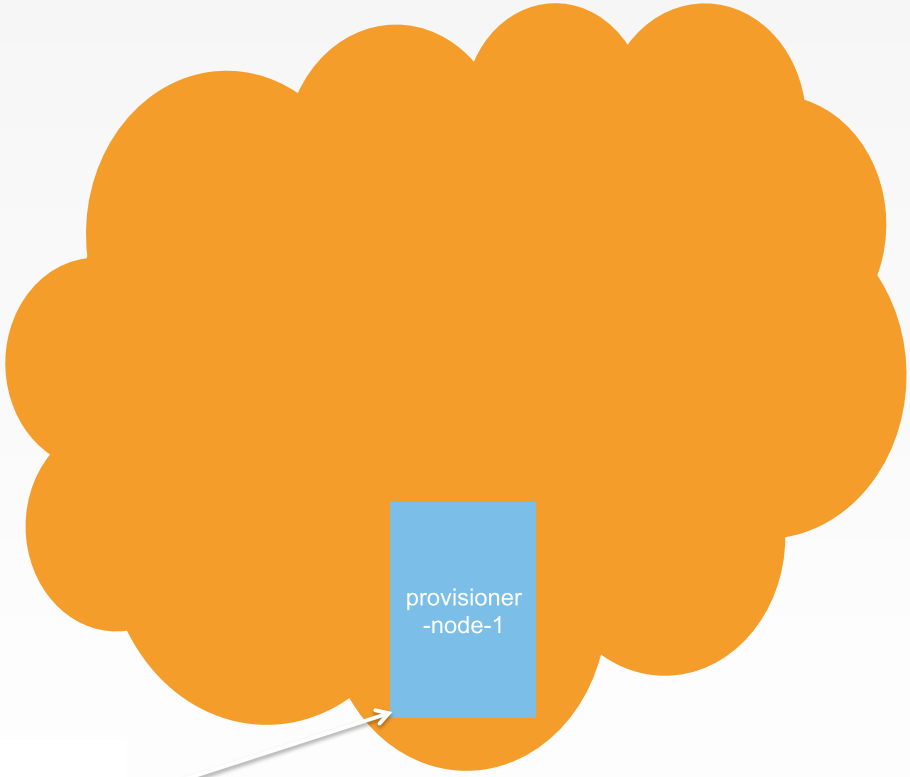
AWS

```
$ knife ec2 server create \
--node-name provisioner-node-1 \
--run-list “recipe[provision]”, “recipe[provision::orchestration]”
```

CHEF

Chef Provisioning – Provisioning Node

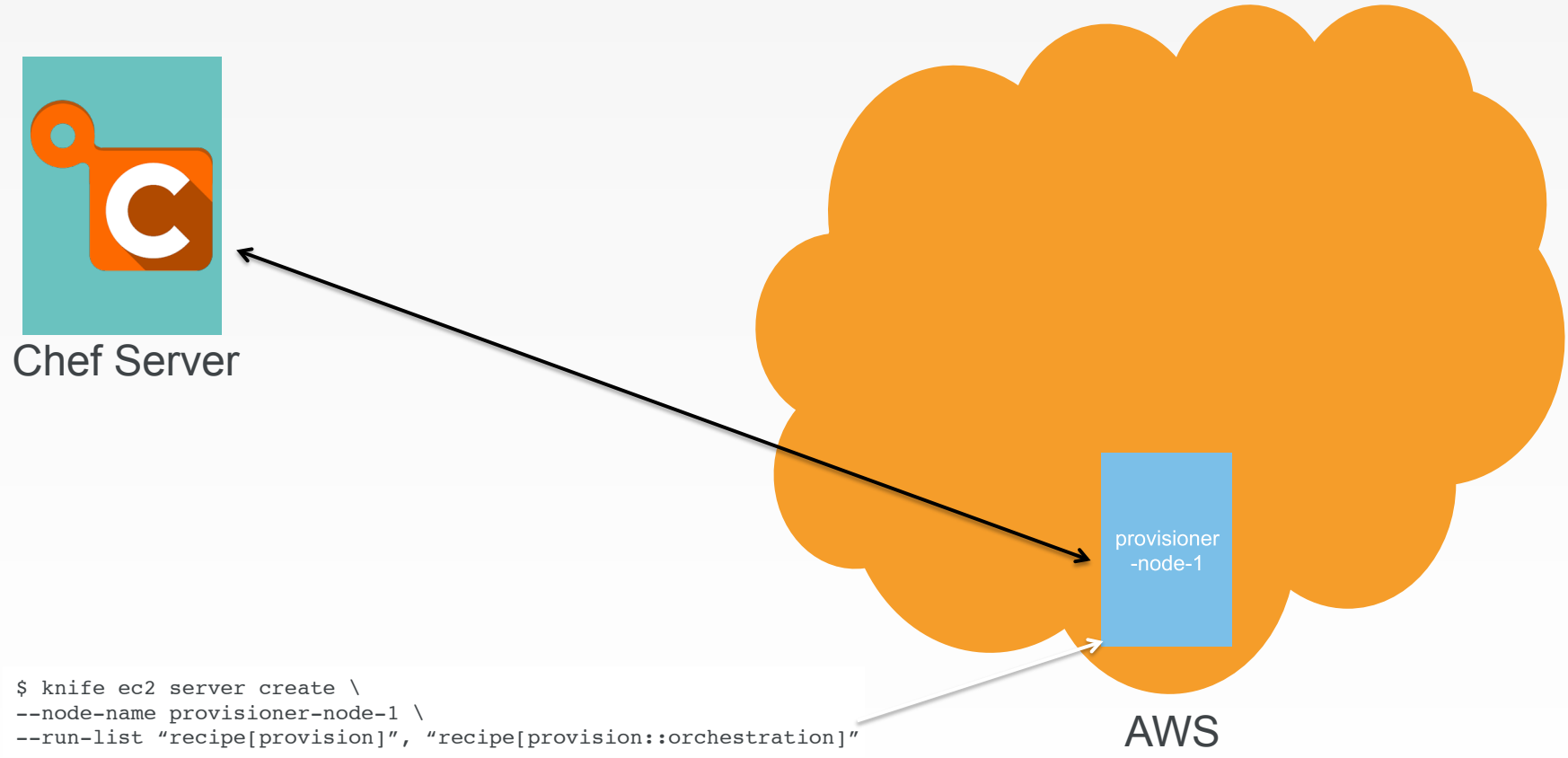
Chef Server




```
$ knife ec2 server create \
--node-name provisioner-node-1 \
--run-list “recipe[provision]”, “recipe[provision::orchestration]”
```

CHEF

Chef Provisioning – Provisioning Node

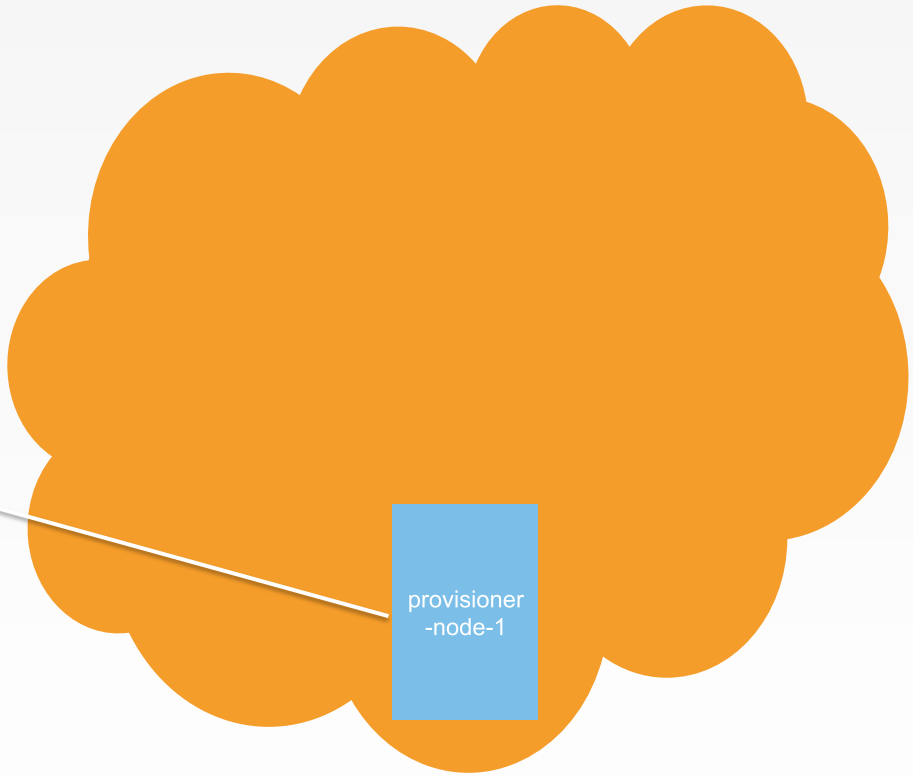



```
$ knife ec2 server create \
--node-name provisioner-node-1 \
--run-list “recipe[provision]”, “recipe[provision::orchestration]”
```

Chef Provisioning – Provisioning Node

Chef Server

```
require 'chef/provisioning'

machine_batch do
  machines %w(primary secondary
web1 web2)
end

machine_batch do
  machine 'primary' do
    recipe 'initial_ha_setup'
  end
end

machine_batch do
  machine 'secondary' do
    recipe 'initial_ha_setup'
  end
end

machine_batch do
  %w(primary secondary).each do |
name|
    machine name do
      recipe 'rest_of_my_shit'
    end
  end
end
```



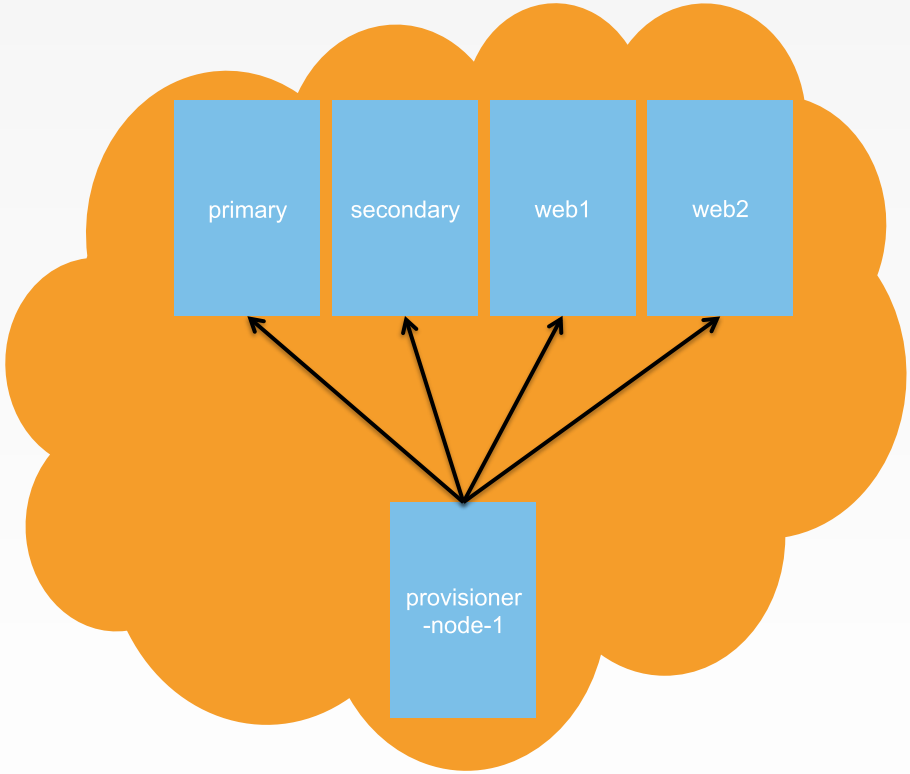
AWS

CHEF

Chef Provisioning – Provisioning Node

Chef Server

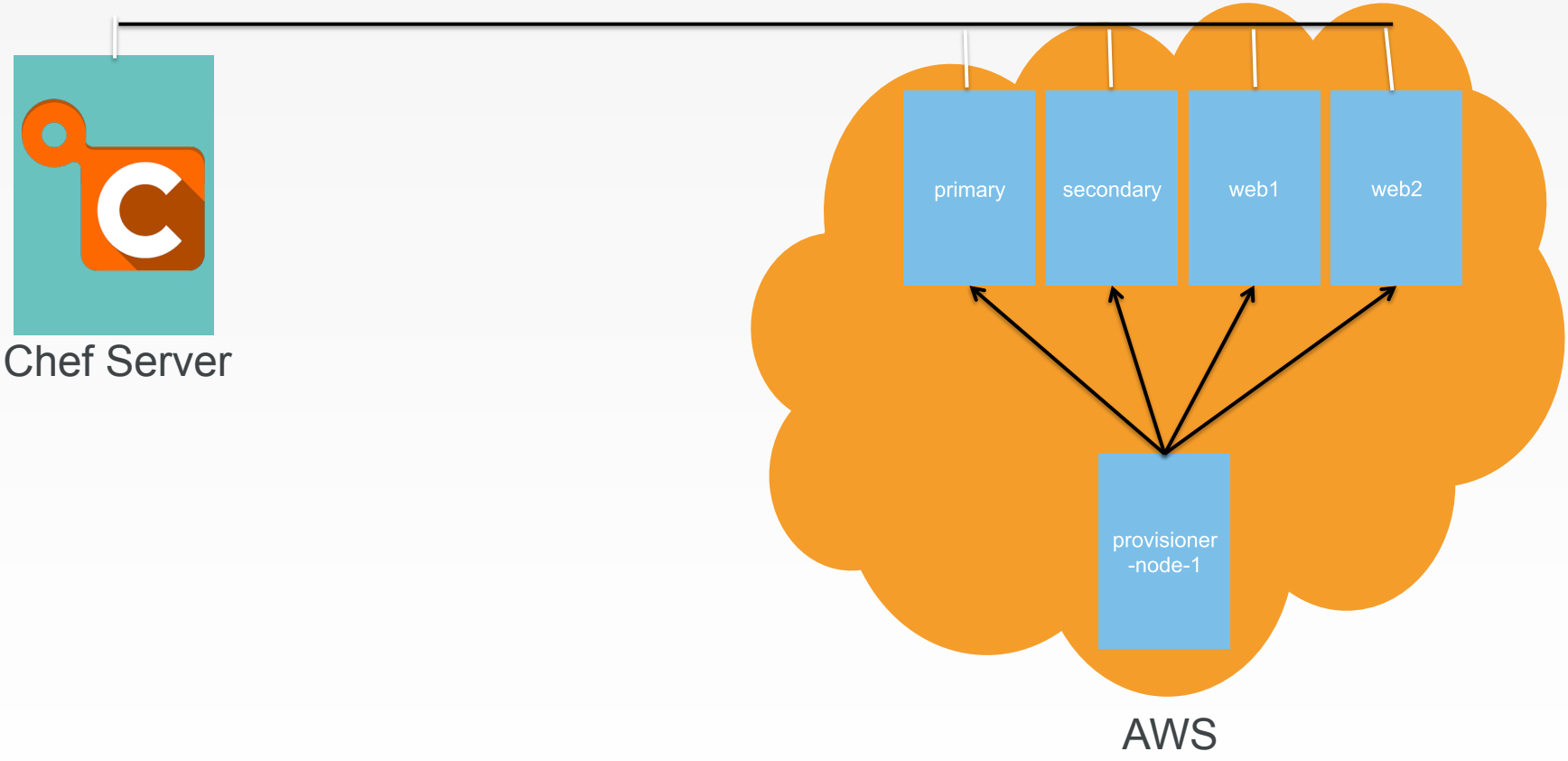
```
require 'chef/provisioning'

machine_batch do
  machines %w(primary secondary
web1 web2)
end

machine_batch do
  machine 'primary' do
    recipe 'initial_ha_setup'
  end
end

machine_batch do
  machine 'secondary' do
    recipe 'initial_ha_setup'
  end
end

machine_batch do
  %w(primary secondary).each do |
name|
    machine name do
      recipe 'rest_of_my_shit'
    end
  end
end
```



AWS

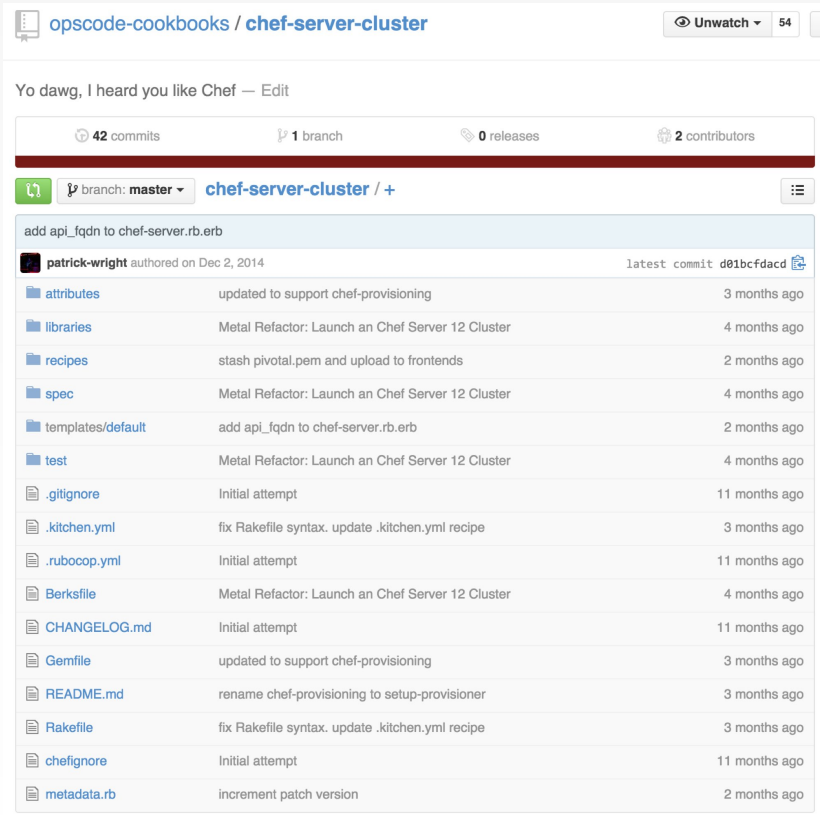
CHEF

Chef Provisioning – Provisioning Node



chef-server-cluster

- chef-server-cluster Cookbook
- Built On chef-provisioning
- Stand up Chef Server 12 AWS in a tiered configuration
- Stand up Analytics



opscode-cookbooks / chef-server-cluster — Unwatch 54

Yo dawg, I heard you like Chef — Edit

42 commits | 1 branch | 0 releases | 2 contributors

branch: master — chef-server-cluster / +

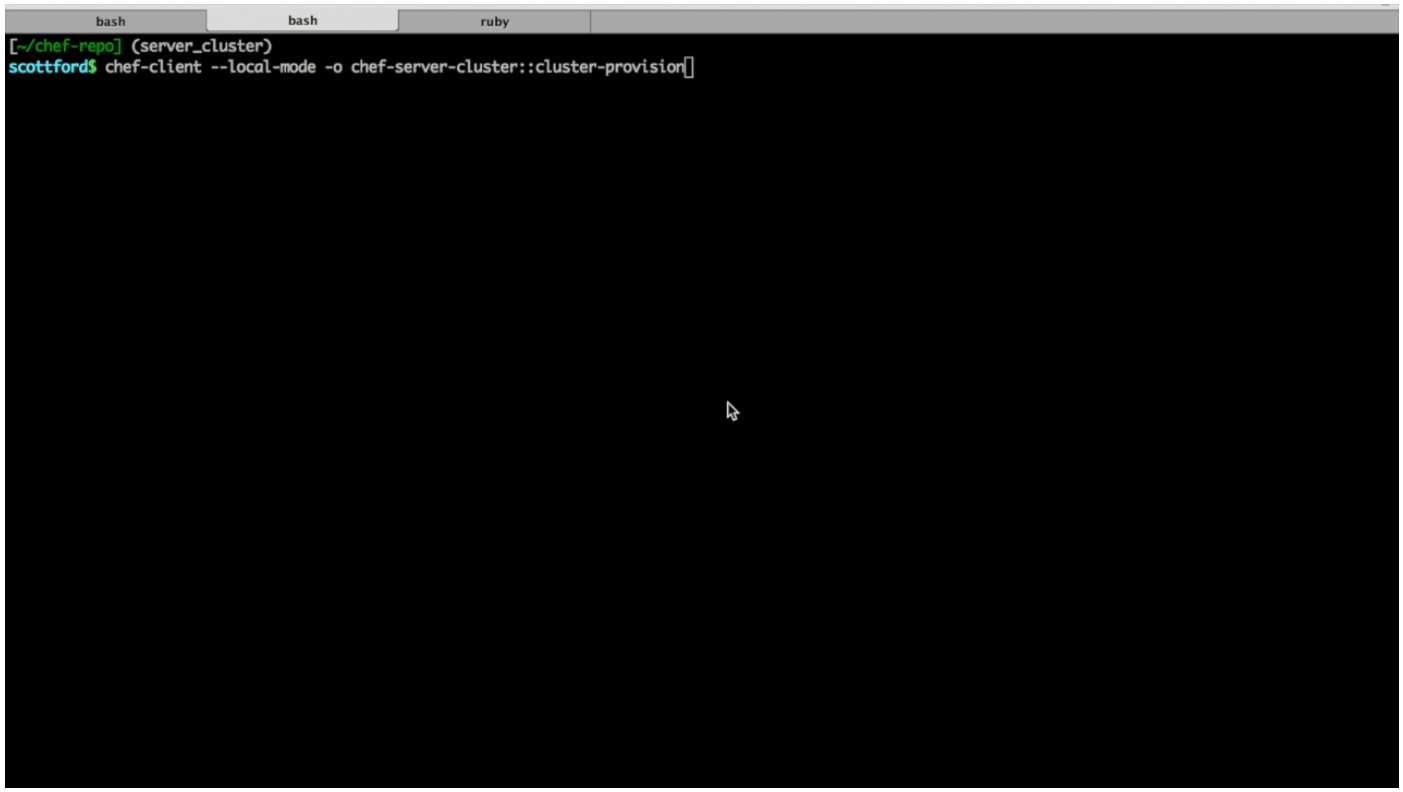
add api_fqdn to chef-server.rb.erb

patrick-wright authored on Dec 2, 2014 — latest commit d01bcfdacd

attributes	updated to support chef-provisioning	3 months ago
libraries	Metal Refactor: Launch an Chef Server 12 Cluster	4 months ago
recipes	stash pivotal.pem and upload to frontends	2 months ago
spec	Metal Refactor: Launch an Chef Server 12 Cluster	4 months ago
templates/default	add api_fqdn to chef-server.rb.erb	2 months ago
test	Metal Refactor: Launch an Chef Server 12 Cluster	4 months ago
.gitignore	Initial attempt	11 months ago
.kitchen.yml	fix Rakefile syntax. update .kitchen.yml recipe	3 months ago
.rubocop.yml	Initial attempt	11 months ago
Berksfile	Metal Refactor: Launch an Chef Server 12 Cluster	4 months ago
CHANGELOG.md	Initial attempt	11 months ago
Gemfile	updated to support chef-provisioning	3 months ago
README.md	rename chef-provisioning to setup-provisioner	3 months ago
Rakefile	fix Rakefile syntax. update .kitchen.yml recipe	3 months ago
chefignore	Initial attempt	11 months ago
metadata.rb	increment patch version	2 months ago

https://github.com/opscode-cookbooks/chef-server-cluster

Chef Provisioning – chef-server-cluster



CHEF

Chef Provisioning Recap

- Machine resource for creating instances
- Drivers for many providers
- Programmatically declare your infrastructure as code
- Testable
- Repeatable

What's on the horizon?

- Version 1.0 coming soon!
- IAM Roles support
- More resources…

How you can contribute

- Use it
- Fork the repositories
- Write new drivers
- Add resources
- Open issues

Where to go from here

- Chef-provisioning
  https://github.com/chef/chef-provisioning
- Chef-provisioning-aws
  https://github.com/chef/chef-provisioning-aws
- Gitter IM Channel
  https://gitter.im/chef/chef-provisioning

CHEF
CODE CAN

What Questions Do You Have?

https://github.com/nathenharvey/chef-london-meetup-feb-2015