Getting Started with Ansible & ServerSpec

Code4Lib17, Los Angeles, CA
March 6, 2017

Alicia Cozine - Data Curation Experts
Anthony Vuong - UCLA Library
Hardy Pottinger - UCLA Library

slides and materials available here:
http://tinyurl.com/code4lib17-ansible-serverspec

Agenda

- 1/2 hour: Intro to ServerSpec
- 1 1/2 hours: Intro to Ansible
- 1/2 hour: Testing Strategies for Ansible
- 1/2 hour: questions and discussion

A word about cathedrals

- Ansible and ServerSpec are fantastic tools, and once you get into using them a bit, you will have grand plans on how to use them more
- You will have those ideas today
- We will not build any cathedrals today
- We may show you hints at the ways some cathedrals have been built, however…

We will not build cathedrals today

We are constantly learning more about our environment

- Library developers shop jobs a lot
- Have you seen the mailing list?
- We change jobs a lot
- We are always the newbie

Why Write ServerSpec Tests?

- Tests are documentation
- Your work group may or may not survive
- Provisioning tools come and go
- No matter what happens to the tools or your workforce, these tests will persist as documentation of your intentions and proof that the service is configured as you expected

ServerSpec

- Extension of RSpec
- Yep, it's a Ruby gem
- Is a great way to force yourself to think about your intentions before you provision a new service
- Is a great way to get to know your existing services

Installing ServerSpec

- It's a Ruby gem, you'll need Ruby 1.9.x+ installed
- gem install serverspec
- you'll also need SSH access to the servers you want to check
- For Ansible you'll probably need Sudo privileges on these servers

Start simple

```
require 'spec_helper'

describe package('httpd') do

  it { should be_installed }

end

describe service('httpd') do

  it { should be_enabled   }

  it { should be_running   }

end

describe port(80) do

  it { should be_listening }

end
```

1. Packages
2. Services
3. Ports

Yes, there is an init script

```
$ serverspec-init

Select OS type:

  1) UN*X

  2) Windows

Select number: 1

Select a backend type:

  1) SSH

  2) Exec (local)

Select number: 1

Vagrant instance y/n: n

Input target host name: www.example.jp

 + spec/

 + spec/www.example.jp/

 + spec/www.example.jp/sample_spec.rb

 + spec/spec_helper.rb

 + Rakefile

 + .rspec
```

Run the tests

```
$ rake spec
/usr/bin/ruby -S rspec spec/www.example.jp/sample_spec.rb
Package "httpd"
  should be installed
Service "httpd"
  should be enabled
  should be running
Port "80"
  should be listening
Finished in 0.21091 seconds (files took 6.37 seconds to load)
4 examples, 0 failures
```

What's next?

- Start simple, with the services you already know
- Consider writing tests **before** you deploy a new service
- As you use these tools, look for ways to consolidate your effort