Table of Contents

Who am I?

Good question!

One of Puppet Labs London based Professional Service Engineers, coming from a background in the London start up community with a heavy leaning towards a the devops culture.

The need to support my coffee addiction has taken me around the world working with and supporting all manner of development teams. The same addiction supported me through long nights of fire fighting and launch management for various game titles within EA, before I joined Puppet Labs.

Feel free to buy me a coffee!

Notes:

this will be additional in print

Know Before You Push 'Go'

Using the Beaker Acceptance Test Framework to Test Your Puppet Module Infrastructure

Beaker is an acceptance testing harness for Puppet PE and other Puppet Projects. A workshop to demonstrate Beaker usage and Beaker-rspec integration with module development. Demonstrating node environments, and quick start and reusability with module skeleton.

The slide deck is available at https://github.com/abuxton/kbypg, you'll need Showoff to run it.

Workshop Activities

What are we going to do?

- Demo, how hard can it be?
- Beaker, well this is what your interested in.
- Tools to make your lives easier, we hope!
- Pulling all of these together.

Our First Demo!

How hard can it be? and what is involved?

Objective of this Demo

Demonstrating reusable components of Beaker and some valuable companion tool.

- Puppetlabs/Beaker.
- Testing environments, nodes and Vagrant.
- rspec_beaker tests.
- Reusability features.
- A working demonstration, Puppetlabs-mysql.
- Using supported modules.

Installation

Bundler makes it easy

`Bundler provides a consistent environment for Ruby projects by tracking and installing the exact gems and versions that are needed.` - bundler

You don't have to use it, but it does make your life easier, repeatable and more dependable.

Notes:

this will be additional in print

Test Environment

Vagrant for when you have to fake it, but fake it well.

Beaker utilises the concept of Nodes for test environments, for the demo were using Vagrant.

Repeatable, dependable and distributable environments.

Demo

Puppetlabs-mysql

Puppetlabs-mysql is being used for the demo.

- Puppetlabs supported module.
- Built in Gemfile for Bundler.
- Default use of centos-64 for nodesets/default.yaml using Vagrant.
- Examples of classes, acceptance, and unit test.

Notes:
this will be additional in print

Beaker

Rspec testing and a whole lot more!

Notes:

this will be additional in print

Objectives

Lets discuss what makes up Beaker!

- Test environments with nodesets.
- Domain Spcecific language (DSL).
- rspec-beaker.
- Command line tool.

Test Environments

SUTs (Systems Under Test)

```
HOSTS:
  centos-64-x64:
    roles:
      - master
    platform: el-6-x86_64
    box : centos-64-x64-vbox4210-nocm
    box_url : http://puppet-vagrant-boxes.puppetlabs.com/centos-64-x64-vbox4210-noc
    hypervisor : vagrant
CONFIG:
  type: foss
```

Supports ec2, Docker, virtual machines, and physical hardware.

Yaml for configuration and reuse, with default support for install at build time, can utilise both both PE and Poss.

Notes:

Creating A Test Environment

DSL

What another language to learn?

OK, so not every one love to learn a new language, and if your new to testing were actually getting you to learn 2 languages, 3 if this is your first mis adventure in Puppet, and possibly 4 if you've never learned ruby and want to go that far.

Don't be scared, there are good reasons, honestly..

- Portability.
- Extending the functionality in a like for like manner.
- Actually breaking down the barriers between languages.

Notes:

git doco RDOC

Domain Specific Language

OK, so what is it?

```
manifest = "file { '#{target}': source => '#{source}', ensure => present }"
apply_manifest_on agent, manifest, { :catch_failures => true }
```

- Its the way weve chosen to ensure your tests are contained in a portable manner.
- Its the way we have constrained tests to allow you to extend functionality in a repeatable way.
- It allows us to provide an interface to Puppet and its tools, which is after all the ultimate goal here.

Notes:

git doco RDOC

Beaker-Rspec

Same as all the rest but different

- Exposes some new variables for Beaker, related to both Beaker and its interaction with your SUT or nodeset.
- Reflects the RSpec you are or are not familiar with, so easy to pick up and lots of sources for guidance and examples.

```
    describe 'creating a database' do
       # Using puppet_apply as a helper
       it 'should work with no errors' do
         pp = <<-EOS
           class { 'mysql::server': root_password => 'password' }
           mysql::db { 'spec1':
             user     => 'root1',
             password => 'password',
           }
         EOS
       end
     end
```

Spec Helpers

Extract the repeatable bits!

`Update spec_helper_acceptance.rb to reflect the module under test. You will need to set the correct module name and add any module dependencies. Place the file in the /spec directory.` spec helper

We use the spec helper for any set up tasks that we want to repeat that are not directly part of the SUT, sometimes this gets a little confused. But what ever works for you.

Notes:

https://github.com/puppetlabs/beaker/wiki/How-to-Write-a-Beaker-Test-for-a-Module#create-the-spec*helper*acceptancerb

Command Line Interface

Some times you just want to run the test!

I don't believe any thing ever turned up polished?

Beaker is no exception and so you can choose to run your tests individually and increment them with out the overheads of all the set up.

So the command line client is always available https://github.com/puppetlabs/beaker/wiki/The-Command-Line.

Notes:

https://github.com/puppetlabs/beaker/wiki/The-Command-Line

Running the tests

Pressing Go before you know!

So how do we run the tests? let us start with the following;

```
# beaker --help
```

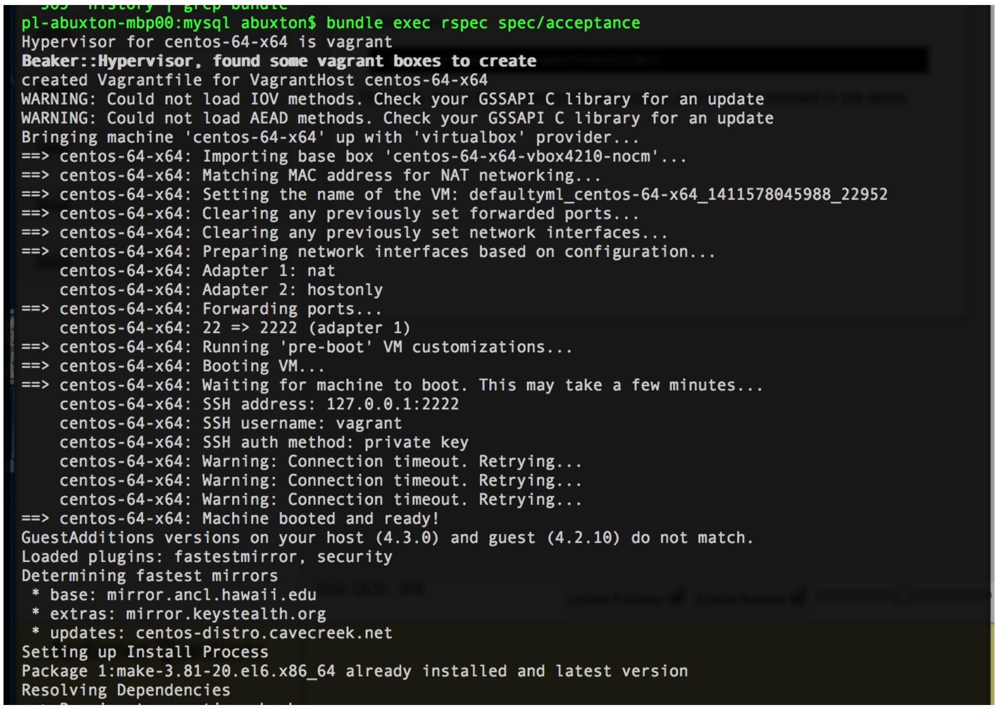
or preferably

```
# bundle exec rspec spec/<testfolder>
```

The latter line is the current preferable usage, and what we invoked in the demo.

So Another Demo

I don't think so lets just look at what we did again.



```
pl-abuxton-mbp00:mysql abuxton$ bundle exec rspec spec/acceptance
Hypervisor for centos-64-x64 is vagrant
Beaker::Hypervisor, found some vagrant boxes to create
created Vagrantfile for VagrantHost centos-64-x64
WARNING: Could not load IOV methods. Check your GSSAPI C library for an update
WARNING: Could not load AEAD methods. Check your GSSAPI C library for an update
Bringing machine 'centos-64-x64' up with 'virtualbox' provider...
==> centos-64-x64: Importing base box 'centos-64-x64-vbox4210-nocm'...
==> centos-64-x64: Matching MAC address for NAT networking...
==> centos-64-x64: Setting the name of the VM: defaultyml_centos-64-x64_1411578045988_22952
==> centos-64-x64: Clearing any previously set forwarded ports...
==> centos-64-x64: Clearing any previously set network interfaces...
==> centos-64-x64: Preparing network interfaces based on configuration...
    centos-64-x64: Adapter 1: nat
    centos-64-x64: Adapter 2: hostonly
==> centos-64-x64: Forwarding ports...
    centos-64-x64: 22 => 2222 (adapter 1)
==> centos-64-x64: Running 'pre-boot' VM customizations...
==> centos-64-x64: Booting VM...
==> centos-64-x64: Waiting for machine to boot. This may take a few minutes...
    centos-64-x64: SSH address: 127.0.0.1:2222
    centos-64-x64: SSH username: vagrant
    centos-64-x64: SSH auth method: private key
    centos-64-x64: Warning: Connection timeout. Retrying...
    centos-64-x64: Warning: Connection timeout. Retrying...
    centos-64-x64: Warning: Connection timeout. Retrying...
==> centos-64-x64: Machine booted and ready!
GuestAdditions versions on your host (4.3.0) and guest (4.2.10) do not match.
Loaded plugins: fastestmirror, security
Determining fastest mirrors
 * base: mirror.ancl.hawaii.edu
 * extras: mirror.keystealth.org
 * updates: centos-distro.cavecreek.net
Setting up Install Process
Package 1:make-3.81-20.el6.x86_64 already installed and latest version
Resolving Dependencies
```

Notes:
this will be additional in print

Tools & Reusability

Lets just do it once ok

Notes:

this will be additional in print

Objectives

Lets introduce some useful things!

- Bundler.
- Rake.
- Puppets Module Skeletons.

Notes:
this will be additional in print

Bundler

Managing Ruby application dependancys

Things you want to do;

- Test the thing.
- Repeat the test.
- Save your self reinstalling every time and documenting all the things!

Bundler is one part of how we do the above, `Bundler provides a consistent environment for Ruby projects by tracking and installing the exact gems and versions that are needed`. The Gemspec becomes part of your module files and all you need to do is document installing Bundler, or maybe you just do that with Puppet too!

Notes:

http://bundler.io/ Install Bundler, we will use it later

Rake

Lets build some things to help

`Rake is a Make-like program implemented in Ruby`!

What does that mean for us, well in this case it just means were going to use it to provide some Ruby based extensions and functionality, and also to make use of functionality in dependencies of Beaker.

https://rubygems.org/gems/rake

```
rake beaker            # Run beaker acceptance tests
rake beaker_nodes      # List available beaker nodesets
rake build             # Build puppet module package
rake clean             # Clean a built module package
rake coverage          # Generate code coverage information
rake help              # Display the list of available rake tasks
rake lint              # Check puppet manifests with puppet-lint
rake spec              # Run spec tests in a clean fixtures directory
rake spec_clean        # Clean up the fixtures directory
rake spec_prep         # Create the fixtures directory
rake spec_standalone   # Run spec tests on an existing fixtures directory
rake syntax            # Check puppet manifest syntax
rake validate          # Validate manifests, templates, and ruby files in lib
```

Notes:

https://rubygems.org/gems/rake

Puppets Module Skeleton

Your out the box starter for 10!

When you run `puppet module generate name-module` you get given the following.

```
a-module/
├── README.md
├── Rakefile
├── manifests
│   └── init.pp
├── metadata.json
├── spec
│   ├── classes
│   │   └── init_spec.rb
│   └── spec_helper.rb
└── tests
    └── init.pp
```

We are not sure this is enough! But its more than you used to get.

Puppet Module Skeletons

So how do we share all this love and reusability?

Well lets take a little from here, and a little from there and add a little extra.

```
As a feature, puppet module tool will use ~/.puppet/var/
puppet-module/skeleton as template for its generate command.
The files provided here are meant to be better templates for
use with the puppet module tool.
```
So lets take a few things and add them to our module skeleton then!, but what?

- Gemfile
- Rakefile

Well this gives us a base functionality tested and working we can build on. but what else?

Notes:

puppet moduel skeleton code morethan seven excellent blog

Module Skeleton Spec folder

Lets have some more reusability goodness!

So what else? How about all some templated tests, spec helpers and nodesets! More Than Seven Garetsh excellent blog post on teh subject.

```
#tree .puppet/var/puppet-module/skeleton/spec
    ├── acceptance
    │   ├── class_spec.rb.erb
    │   └── nodesets
    │       ├── centos-64-x64.yml
    │       ├── default.yml
    │       └── ubuntu-server-12042-x64.yml
    ├── classes
    │   ├── coverage_spec.rb
    │   └── example_spec.rb.erb
    ├── spec_helper.rb
    └── spec_helper_acceptance.rb.erb
```

Notes:

credits: https://github.com/garethr/puppet-module-skeleton https://forge.puppetlabs.com/supported https://github.com/abuxton/module_skeleton

Demo Module Skeleton

So how do we use it!

- Lets put the skeleton in place. https://github.com/abuxton/module_skeleton
- Lets generate a module. `puppet module generate a-test`
- Lets see some the unit tests.

Notes:

All the command you need are in the slides or these notes.

Putting it all together.

How do we use this in anger?

Notes:

this will be additional in print

Final Objectives

What do we do with it all!

Lets use what we've seen, to go from nothing to test.

- Install the module skeleton.
- Generate a module.
- Run the initial tests

Notes:
this will be additional in print

Installation

Putting the pieces in place.

So what do we need for this?

- Puppet.
- Vagrant.
- Bundler.
- Module Skeleton.

Nodes or SUTs

Lets virtualise some hardware

We need nodes to test against, or servers under test (SUT). The module skeleton now provides those too.

```
tree ~/.puppet/var/puppet-module/skeleton/spec/acceptance/nodesets/
/Users/abuxton/.puppet/var/puppet-module/skeleton/spec/acceptance/nodesets/
├── centos-64-x64.yml
├── default.yml
└── ubuntu-server-12042-x64.yml
```

Notes:

this will be additional in print

Module to test.

What could go wrong!

We all know live demos are so successful. So lets keep this simple.

- Were going to generate a module (NTP).
- Were going to take a look at the out put.
- Were going to run the tests.

Whats the worst that can go wrong!

Notes:
this will be additional in print

Can I get a real Drink now

Yes I mean Coffee.

Notes:

this will be additional in print

So what did we do?

What comes next?

In summary, We actually managed 3 workshops in one.

- How to test Puppets supported modules.
- How to use and make use of the Puppet Module tool & Puppet module skeleton.
- How to generate a module and test that code.

What do we do with all this.

- Read some more Beaker, https://github.com/puppetlabs/beaker/wiki
- Run some tests https://forge.puppetlabs.com/modules?endorsements=supported to forge supported list.
- Run this at home and try extend it into your working day.
- https://github.com/abuxton/kbypg my presentation on git

Thank you

I hope you learned a little something to take away.

Credits

Heres all the links.

Beaker and useful links.

- https://github.com/puppetlabs/beaker link to Beaker
- http://www.morethanseven.net/2014/02/05/a-template-for-puppet-modules/link to Gareths blog
- https://forge.puppetlabs.com/modules?endorsements=supported link to Forge.
- https://github.com/abuxton/kbypg this talk.

Additional credits:

- http://www.charlottemoss.co.uk/ profile image.
- http://2014.puppetconf.com/ There is a threat you may be eventiually able to find audio of this session.