Hands On DevOps

Mike Crute (@mcrute) and Chris Laco (@claco)

Assumptions

- Can install software on your laptop
- Basic command-line text editor skills
- Basic familiarity with a Unix shell

The Grand Plan

1. Perquisite setup
2. Standalone puppet
3. Inside puppet
4. Deploy an application
5. Using community resources
6. Convert to puppet server
7. Handling multiple environments

Command Prompts

- # — Root prompt (via sudo or su)
- $ — User prompt

Inside Puppet

Do puppets even have guts?

Puppet Standalone

Installing Puppet

```
# wget https://apt.puppetlabs.com/puppetlabs-
  release-precise.deb

# dpkg -i puppetlabs-release-precise.deb

# apt-get update

# apt-get install puppet-common
```

Test It Out

```
# puppet

# puppet help

# puppet help apply | less

# ls /etc/puppet
```

Hello World?

```
# vim hello.pp

  notify { “Hello World!”:
  }
```

Hello World?

```
# puppet apply hello.pp

  Notice: Compiled catalog for ubuntu in
environment production in 0.01 seconds
Notice: Hello world!
Notice: /Stage[main]/Main/Notify[Hello
world!]/message: defined 'message' as 'Hello
world!'
Notice: Finished catalog run in 0.02 seconds
```

Manifest

Resource

Type

Provider

Resources

by way of the Resource Abstraction Language

Canned Resources

- cron
- exec
- file
- group
- host
- interface
- mount
- nagios
- package
- service
- ssh_authorized_key
- user

Anatomy of a Resource

```
user { 'kate':
    ensure     => present,
    uid        => '507',
    gid        => 'admin',
    shell      => '/bin/zsh',
    home       => '/home/fsck8',
    managehome => true,
}
```

Uniqueness

- Everything has a title
- Context: per host and within puppet
- Title of a resource must be unique for the host and type
- Can cheat: use other attributes (e.g. file has title and path)

Inspection

```
# puppet resource user root

  user { 'root':
    ensure           => 'present',
    comment          => 'root',
    gid              => '0',
    home             => '/root',
    password         => '…',
    password_max_age => '99999',
    password_min_age => '0',
    shell            => '/bin/bash',
    uid              => '0',
  }
```

Manipulation

```
# puppet resource user kate ensure=present
  shell="/bin/bash" home="/home/fsck8"
  managehome=true
notice: /User[kate]/ensure: created

user { 'kate':
   ensure => 'present',
   home   => '/home/fsck8',
   shell  => '/bin/zsh'
}
```

Resource Defaults

```
File {
        owner => 'root',
        group => 'root',
        mode  => 0640,
}
```

Ammending Resources

```
file {'/etc/passwd':
  ensure => file,
}
```

```
... some time later ...

File['/etc/passwd'] {
  owner => 'root',
  group => 'root',
  mode  => 0640,
}
```

Condensing

```
file { ['/etc/rc.d/rc0.d',
        '/etc/rc.d/rc1.d',
        '/etc/rc.d/rc2.d',
        '/etc/rc.d/rc3.d']:
  ensure => directory,
  owner  => 'root',
  group  => 'root',
  mode   => 0755,
}
```

Condensing

```
file {
  '/etc/rc.d':
    ensure => directory,
    owner  => 'root',
    group  => 'root',
    mode   => 0755;

  '/etc/rc.d/init.d':
    ensure => directory,
    owner  => 'root',
    group  => 'root',
    mode   => 0755;
}
```

Collectors

```
class base::linux {
    file {'/etc/passwd':
        ensure => file,
    }
}
```

```
include base::linux

File <| tag == 'base::linux' |> {
    owner => 'root',
    group => 'root',
    mode  => 0640,
}
```

Manifests

Make a File

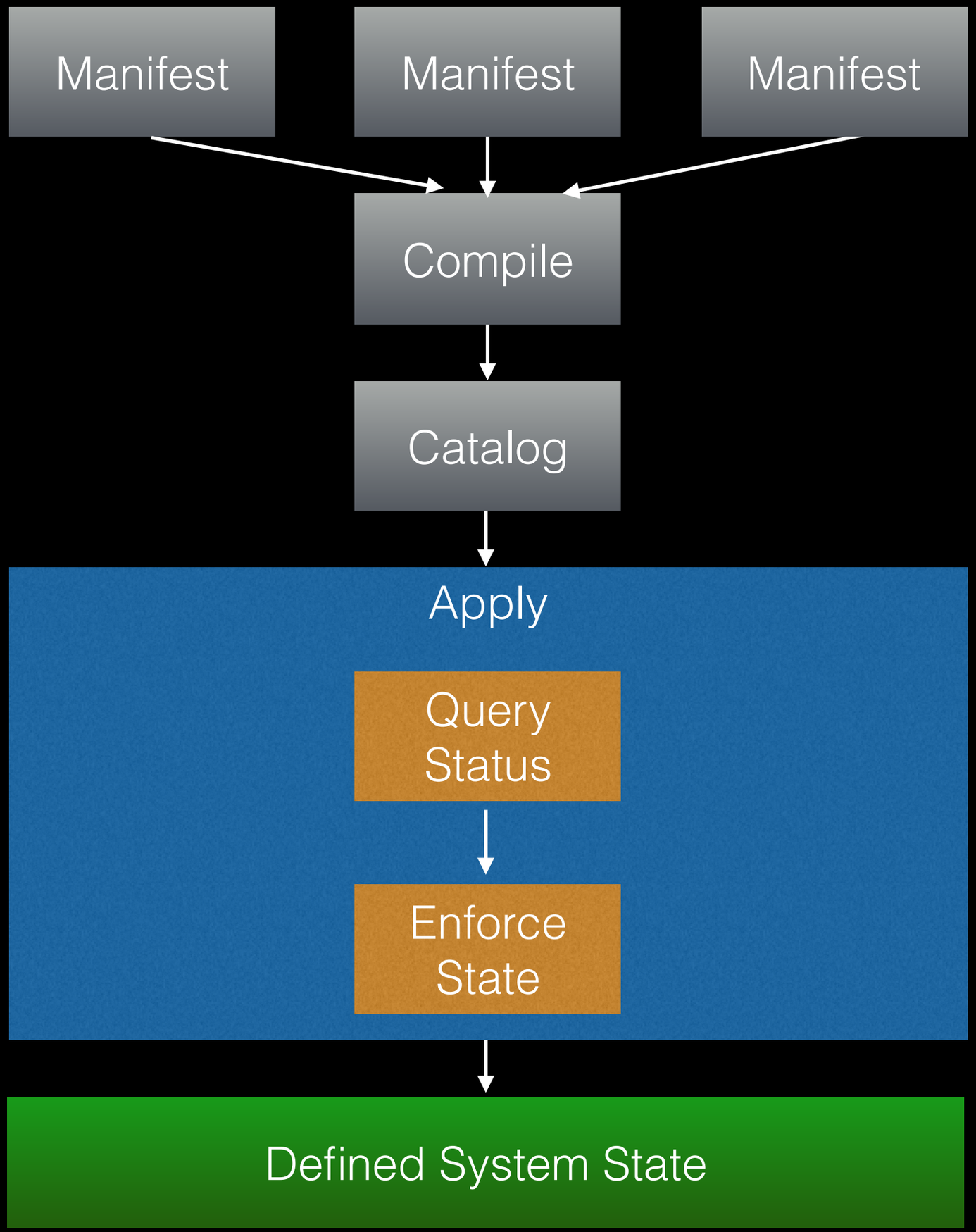
```
# vim /etc/puppet/sample.pp
file {'testfile':
    path    => '/tmp/testfile',
    ensure  => present,
    mode    => 0640,
    content => “I'm a test file.”,
}
```

Make a File

```
# puppet apply sample.pp

  Notice: Compiled catalog for ubuntu in environment
  production in 0.03 seconds
  Notice: /Stage[main]/Main/File[testfile]/ensure: created
  Notice: Finished catalog run in 0.02 seconds

# ls /tmp

  testfile  VMwareDnD  vmware-root

# cat /tmp/testfile

  I'm a test file.
```

What If...

```
# chmod 000 /tmp/testfile

# puppet apply sample.pp

  Notice: Compiled catalog for ubuntu in
  environment production in 0.03 seconds
  Notice: /Stage[main]/Main/File[testfile]/mode:
  mode changed '0000' to '0640'
  Notice: Finished catalog run in 0.02 seconds
```


Manifest
Manifest
Manifest
Compile
Catalog
Apply
Query Status
Enforce State
Defined System State

site.pp

- /etc/puppet/manifests/site.pp
- Entry point for puppet apply
- Generally nodes only or default setup

```
File { backup => "main" }
Exec { path => "/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin" }

node default_system {
  include puppet
  include sshd
  include network

  include base
  include base::sudo
  include base::shell
}

node app_server inherits default_system {
  include nginx
  include base::python

  class { "firewall":
    profile => "web",
  }
}

import "nodes/*.pp"
```

Nodes

- A host or role that can have resources
- Only one per catalog
- KISS: includes only (generally)

Name Matching

1. Exact name of host
2. Regular expression match on hostname
3. Fully qualified domain name via stemming
4. default node

Nodes Define a Scope

```
node 'common' {
    $ntpserver = ‘time.example.com’
    include common
    include ntp
}

node 'www01.example.com' inherits ‘common' {
    $ntpserver = ‘0.pool.ntp.org’
}
```

Resource Ordering

Metaparameters

- before
- require
- notify
- subscribe

Manage a User

```
user { 'kate':
    ensure => present,
    home   => '/home/fsck8',
    shell  => '/bin/zsh',
}

ssh_authorized_key { 'kate':
    ensure => present,
    type   => 'ssh-dss',
    key    => '...',
}
```

Manage a File

```
file {'/tmp/test1':
    ensure  => present,
    content => "Hi.",
  }

  user {'kate':

  }

File['/tmp/test1'] -> User['kate']
```

Manage a User

```
user { ‘developer’:
    ensure => present,
    home   => ‘/home/developer’,
    shell  => ‘/bin/bash’,
}

ssh_authorized_key { ‘developer’:
    ensure  => present,
    type    => ‘ssh-dss’,
    key     => ‘...',
    require => User[‘developer’],
}
```

Manage a User

```
user { ‘developer’:
    ensure => present,
    home   => ‘/home/developer’,
    shell  => ‘/bin/bash’,
}

ssh_authorized_key { ‘developer’:
    ensure  => present,
    type    => ‘ssh-dss’,
    key     => ‘...',
    user => User[‘developer’],
}
```

Variables, Conditions, Facts

Types

- string: ‘singly quoted’ or “doubly quoted”
- number
- boolean: true or false
- [array]
- { hash => value }
- undef

Rules of the Road

- Always start with a $
- Default of unassigned variables is undef
- Support “${interpolation}”
- Assign once only

Scoping

- Variables have short and long names
  - $short_name
  - $::short_name in the top scope
  - $module::short_name in a module scope

Facts

- Generated by facter
- Can be extended by plugins
- Exist in the top scope

Facts

```
# facter

  architecture => i386
  bios_release_date => 07/31/2013
  bios_vendor => Phoenix Technologies LTD
  bios_version => 6.00
  domain => localdomain
  fqdn => ubuntu.localdomain
  hardwareisa => i686
  hardwaremodel => i686
  hostname => ubuntu
  id => root
  interfaces => eth0,lo
  ipaddress => 172.16.67.137
  ipaddress_eth0 => 172.16.67.137
  ipaddress_lo => 127.0.0.1
  is_virtual => true
  kernel => Linux
  kernelmajversion => 3.5
  kernelversion => 3.5.0
```

Conditional Statements

```
if condition {
    block of code
} elsif condition {
    block of code
} else {
    block of code
}
```

Truthiness

- true
- 1
- non-empty strings

Case

```
case $operatingsystem {
    centos: { $apache = "httpd" }
    redhat: { $apache = "httpd" }
    /(?i)(ubuntu|debian)/: { $apache =
'apache2' },
    default: { fail("Unrecognized OS") }
}

package {'apache':
    name   => $apache,
    ensure => latest,
}
```

Selectors

```
$apache = $operatingsystem ? {
    centos  => 'httpd',
    redhat  => 'httpd',
    default => undef,
}
```

Installing nginx

```
# vim /etc/puppet/manifests/site.pp

  node default {
      $nginx_package = $operatingsystem ? {
          redhat => 'nginx',
          Ubuntu => 'nginx-full',
          default => 'nginx'
      }

      package { $nginx_package:
          ensure => latest,
      }
  }
```

Installing nginx

```
# puppet apply manifests/site.pp
Notice: Compiled catalog for ubuntu in
environment production in 0.07 seconds
Notice: /Stage[main]/Main/Node[default]/
Package[nginx-full]/ensure: ensure changed
'purged' to 'present'
Notice: Finished catalog run in 1.78 seconds
```

Classes

Classes are Containers

- Define a group of resources once deploy many times
- Don’t get used until included
- Not really classes in a software development sense

A Class

```
class my_class {
    notify { "This actually did something":
    }
}

include my_class
```

Refactor

```
class nginx {
    $nginx_package = $operatingsystem ? {
        redhat => 'nginx',
        Ubuntu => 'nginx-full',
        default => 'nginx'
    }

    package { $nginx_package:
        ensure => installed,
    }

    service { 'nginx':
        ensure  => running,
        require => Package[$nginx_package],
    }
}

node default {
    include nginx
}
```

Parameters

```
class echo ($to_echo = "default value") {
    notify {"What? ${to_echo}.":
    }
}

class {'echo_class':
    to_echo => 'Custom value',
}
```

Refactor

```
class nginx($ensure = present) {
    $nginx_package = $operatingsystem ? {
        redhat => 'nginx',
        Ubuntu => 'nginx-full',
        default => 'nginx'
    }

    package { $nginx_package:
        ensure => $ensure ? { present => installed, default => absent },
    }

    service { 'nginx':
        ensure  => $ensure ? { present => running, default => stopped },
        require => Package[$nginx_package],
    }
}

node default {
    class { 'nginx':
        ensure => absent,
    }
}
```

Caveat

- Classes are singletons

Modules

Anatomy of a Module

- manifests — store all of your manifests
- files — any extra config files
- templates — ruby templates for the module
- lib — ruby extensions to puppet

Make a Module

```
# tree .

.
├── manifests
|   └── site.pp
├── modules
|   └── nginx
|       ├── files
|       |   └── init.pp
|       ├── manifests
|       └── templates
├── puppet.conf
└── templates
```

Make a Module

```
# cd /etc/puppet

# mkdir -p modules/nginx/
  {manifests,files,templates}
```

Make a Module

```
# vim /etc/puppet/modules/nginx/manifests/init.pp

  class nginx($ensure = present) {
      $nginx_package = $operatingsystem ? {
          redhat => 'nginx',
          Ubuntu => 'nginx-full',
          default => 'nginx'
      }

      package { $nginx_package:
          ensure => $ensure ? { present => installed, default => absent },
      }

      service { 'nginx':
          ensure  => $ensure ? { present => running, default => stopped },
          require => Package[$nginx_package],
      }
  }
```

Use The Module

```
node default {
    class { 'nginx':
        ensure => absent,
    }
}
```

Managing Files

```
# /etc/puppet/modules/nginx/files/nginx.conf

class nginx {
file { '/etc/nginx/nginx.conf':
    ensure => $ensure ? {
        present => present,
        default => absent
    },
    source => 'puppet:///modules/nginx/
nginx.conf',
}
}
```

LUNCH

- Puppet Forge
- Convert to client/server model
- Handling Multiple Environments

Templates

Templates

- ERB templates
- Accessed by the template function

Rendering Templates

```
file {'/etc/foo.conf':
      ensure  => file,
      require => Package['foo'],
      content => template('foo/foo.conf.erb'),
}
```

Variables

- Everything in your local scope
- Facts and globals accessible with @ prefix
- Everything else via scope.lookup
  - `scope.lookup('apache::user')`

ERB Templates

- <% document = “foo” %>

ERB Templates

- <%# This is a comment %>

ERB Templates

- <%= local_variable %>

ERB Templates

- <% if foo: %>
- <% end %>

ERB Templates

- <%- document += this_line -%>

The Resource

```
# /etc/puppet/modules/nginx/manifests/vhost.pp

class nginx::vhost(
  $ensure=present, $hostname, $www_root) {

  file { "/etc/nginx/sites-enabled/${hostname}":
      ensure => $ensure,
      content => template('nginx/vhost.erb'),
      notify => Service[‘nginx’],
  }

}
```

The Template

```
server {
    root <%= @www_root %>;
    index index.html index.htm;
    server_name <%= @hostname %>;
}
```

site.pp

```
node default {
    class { 'nginx':
        ensure => present,
    }

    class { 'nginx::vhost':
        ensure => present,
        hostname => 'xapian-docs',
        www_root => '/usr/share/doc/python-
xapian/',
    }
}
```

```
# curl -H “Host: xapian-docs” 127.0.0.1
```

Defined Types

The Fix

```
define nginx::vhost(
    $ensure=present, $hostname, $www_root) {

  file { "/etc/nginx/sites-enabled/${hostname}":
      ensure => $ensure,
      content => template('nginx/vhost.erb'),
  }

}
```

Now Use It

```
nginx::vhost {‘a fancy name':
  hostname => ‘xapian-docs',
  www_root => ‘/doc/root‘,
}
```

The Title

```
define nginx::vhost(
    $ensure=present, $host=$title, $www_root) {

  file { "/etc/nginx/sites-enabled/${host}":
      ensure => $ensure,
      content => template('nginx/vhost.erb'),
  }

}
```

Now Use It

```
apache::vhost {‘xapian-docs‘:
  www_root => ‘/doc/root‘,
}
```

Remember

- You still can’t duplicate resource names!

Puppet Forge

Everything you've done is useless.

puppet module

- Part package manager
- Part release tool

Puppet Forge

- Community modules for everything
- Several are “official” from Puppet Labs

Puppet Forge

```
# puppet module search nginx

# puppet module install puppetlabs/nginx

# puppet module list
```

Detour

```
package { "python-pip":
        ensure => present,
}
```

```
package { [“django”, “gunicorn”]:
        ensure => present,
        provider => 'pip',
        require => Package["python-pip"],
}
```

Detour

```
# django-admin.py startproject foo

# gunicorn --pythonpath=foo foo.wsgi:application
```

Convert to Puppet Server

It’s web scale!

Puppetmaster

```
# apt-get install puppetmaster

# puppet agent -t --server=ubuntu.localdomain
```

Puppetmaster

```
# vim /etc/default/puppet

  START=yes
```

Agent

```
# puppet agent -t –
  server=workstation.kalaharipub.local
```

Installing Puppet

```
# wget https://apt.puppetlabs.com/puppetlabs-
  release-precise.deb

# dpkg -i puppetlabs-release-precise.deb

# apt-get update

# apt-get install puppet
```

Certificates

```
# puppet cert list

# puppet cert sign host.example.com

# puppet agent -t
```

Finding the Server

- DNS lookup for puppet
- Config file /etc/puppet/puppet.cfg
- Command line switch

Host Bootstrap Script

- https://gist.github.com/mcrute/8324918

Mike Crute
Finite Loop Software
mike@finiteloopsoftware.com