Puppet in the Enterprise

Thomas Uphill
thomas@narrabilis.com
Latest version
http://goo.gl/GOTLff

Example Files
https://github.com/uphillian/lisa2014

If you see something, say something!
Google Comments enabled
Watch in Presentation Mode!

Seriously, Trust Me

...Animations Are Good things
time shift

"Tend the flock, not the sheep"

-- Me
The Puppet Problem

**System Administrators**
- scripts
- pipes/redirection
- lazy

**Developers**
- objects
- code reuse
- lazy
Puppet Problem

system administrators
devengers
Roles and Profiles
Roles and Profiles and Exceptions
if $::hostname == "thatweirdone" {
    include obscureclass95
} else {
    include superobscure72
}
The Puppet Problem

- Minimize exceptions
- if else if else if else if else if else if else
- case
  - case
    - case
Hierarchical

Separating **data** from **code**

Techniques:

- Parameterized classes
- hiera_include
- fact based hierarchy

But first... **custom facts**
custom facts

Why?

- facts are loaded and defined early in catalog compilation
- facts can be used in hiera hierarchy
- facts can be used as selectors in case statements
custom facts

Two methods:

- external facts
  - /etc/facter/facts.d/fact.txt
  - /etc/facter/facts.d/fact.{yaml|json}
  - /etc/facter/facts.d/filename*  (chmod +x)

- custom facts
  - module/lib/facter/fact.rb
custom facts

External facts:

- simpler to write
- unavailable for first catalog
- compile
- cannot use facts

```
#!/bin/bash
otherfact=$(facter -p otherfact
factone=$(((otherfact + 1)))
echo factone=$factone
echo facttwo=other
```
custom facts

written in ruby

can access previously defined facts

puppet 3+ $\Rightarrow$ automatically sync'ed
class Parameterized Classes

```php
class resolv ($nameserver = '8.8.8.8') {
    file { ?>/etc/resolv.conf':
        content => "nameserver $nameserver",
        owner => '0',
        group => '0',
        mode => '0644',
    }
}
```
Parameterized Classes

When?

- Include **without** modification:
  - include resolv
  - class {'resolv': }
- Include **with** modification
  - class {'resolv':
        nameserver => 'value'
    }

class selinux ($config = 'enforcing') {
    case $config {
        'enforcing': {
            exec {'selinux_enforcing':
               command => '/usr/sbin/setenforce 1',
            }
        }
        /permissive|disabled/: {
            exec {'selinux_permissive':
               command => '/usr/sbin/setenforce 0',
            }
        }
    }
    file {'/etc/selinux/config':
       content => "SELINUX=$config
SELINUXTYPE=targeted",
    }
}
Automatic Parameter lookup

---
resolv::nameserver: 8.8.4.4

include resolv
group class {'resolv':}
mode
}
}

nameserver 8.8.4.4
Hierarchy

:hs:"cunning/%{::cunning_fact}"
- global

/etc/hieradata/cunning/
  pst.yaml
  cmt.yaml
  est.yaml

nameserver 8.8.4.1
hiera_include('lookupkey','notfound')

- Lookup 'lookupkey' in hiera
- Include each class listed in 'lookupkey'
- If nothing found, include 'notfound'
- Call hiera_include from site.pp

```ruby
site.pp
node default {
  class {'base': }
  hiera_include('classes','notfound')
}
```
fact based hierarchy

hiera.yaml

---

:hierarchy:
  - "${hostname}"
  - "${operatingsystem}"
  - "is_virtual/${is_virtual}"
  - common
fact based hierarchy

hiera.yaml

/site.pp

...  
hiera_include('classes','notfound')  
...

true.yaml

...  
classes: 'virtual_machine'
...

notfound/manifests/init.pp

class notfound {  
  # nothing
}

virtual_machine/manifests/init.pp

class virtual_machine {  
  service {'tuned': ensure => running }  
  exec {'tuned-adm virtual':  
     command => 'tuned-adm profile virtual-guest',  
  }
}

common.yaml
fact based hierarchy - custom fact

hiera.yaml

---

:hierarchy:
- " %{hostname}"
- " %{operatingsystem}"
- "is_virtual/%{is_virtual}"
- "custom_fact/%{custom_fact}"
- common
Centralized or Not?
Centralized/Decentralized

- puppetdb
- master/CA
- hiera

- code

- node
- node
- node
- node

- node
- node
- node
- node
Decentralized puppet apply

role -> node
Centralized puppet agent Scaling
What is the most important thing to remember about puppet?

- Puppet is a web service.
- Puppet is a web service on port 8140.
- Puppet is an SSL web service on port 8140.
Scaling

REST API
https://puppet:8140/environment/resource/key

catalog  file_metadata
certificate  fact
file_content
Scaling

node -> 8140 -> proxy (puppet) -> puppetca

proxy (puppet) -> catalog -> worker

worker
Scaling

- apache
- nginx
- haproxy

proxy (puppet)
<Proxy balancer://puppetca>
BalancerMember http://127.0.0.1:18140
</Proxy>

<Proxy balancer://puppetworker>
BalancerMember http://192.168.100.101:18140
BalancerMember http://192.168.100.102:18140
</Proxy>

Listen 8140
<VirtualHost ::8140>
  SSLEngine on
  ...
</VirtualHost>

ProxyPassMatch ^/([^/]+)/certificate.*$ balancer://puppetca/$1
ProxyPass / balancer://puppetworker/
</VirtualHost>
Scaling: does it actually work?

Demo 1: VM
  - proxy
  - passenger
  - puppetdb / postgresql
Infrastructure as Code
Software as a Service
Platform as a Service
buzzword something

development
continuous integration
refactoring

workflow
Workflow

Decentralized:
- create machine
- install puppet
- apply role
- download code
- puppet apply

Centralized:
- create machine
- install puppet
- apply role
- puppet agent
Hardware

physical

virtual
Bootstrapping

Decentralized:
- create machine
- **install puppet (bootstrap)**
- apply role
- download code
- puppet apply

Centralized:
- create machine
- **install puppet (bootstrap)**
- apply role
- puppet agent
Bootstrapping

Install Puppet

- `gem install puppet`
- `apt-get install puppet`
  - install puppetlabs apt source
- `yum install puppet`
  - install puppetlabs yum repo
- `tar file`
- `port/brew install puppet`
Bootstrapping

Apply role

- hiera
  \$role = hiera('role','undefined')
- ENC
  CMDB lookup
  LDAP lookup

- node definition \textless{} \textit{doesn't scale well} \textgreater{}

\texttt{site.pp}
node 'nodename' { include 'webserver' }
Bootstrapping

ensure puppet running
  - agent: service
  - apply: cron task
install puppet

package {'puppet': ensure => installed}
Workflow - creation

- Provision (VM/Physical)
- Bootstrap puppet
  - Assign role to node
- Apply puppet (agent or apply)
  - ensure puppet installed properly
  - ensure puppet running (service or cron task)
- Register node
  - monitoring / nagios
Workflow - deletion

- Decommission (VM/Physical)
- Remove role assignment
  - hiera/enc/ldap
- Delete from Reports (foreman/console)
- De-register node
  - monitoring/nagios
Maximize return on investment:
1. install puppet early
2. apply bootstrap.pp manifest
3. ***
4. profit
Scaling

node → 8140 → proxy (puppet) → puppetca

proxy (puppet) → catalog → worker → worker → worker
automated workflow
Workflow (code)

- Push code to masters
- Branches
  - Code promotion
  - Environments
  - Purge old
- Hieradata
- defacto source code control for puppet
- integrates into workflow
- cheap branches
- hooks
Git Hooks
http://goo.gl/dg5TVw
- Branch is a reference
- References are hashes
- Branches are cheap
- branch == environment
- directory environments (3.6+)

```plaintext
environmentpath = /etc/puppet/environments
environment.conf
  modulepath = relative:path:/absolute/path
  manifest = relative/path/site.pp
- directory with environmentpath is the environment
```
directory environments

$environmentpath/
  production/
    /modules
    /manifests

devel/
  branch
    /modules
    /manifests
- repository 101
  - remote / origin
  - bare repos

- git server
- puppet worker
- worker
- worker
hooks

many hooks, two useful here:

- pre-receive
  separate who can do what
- post-receive
  push code
puppet-sync

- puppet-sync
  https://github.com/pdxcat/puppet-sync
- pull down a single
  git
  server
  puppet
  worker
  puppet
  worker
  worker
#!/bin/bash

DEPLOY='/etc/puppet/environments'
REPO='hostname:/srv/git/repos/puppet.git'

read oldrev newrev ref
BRANCH=${ref/*/\/*/}

sudo -u puppet ssh worker "puppet-sync --branch $BRANCH --repository $REPO --deploy $DEPLOY"

exit=$?
exit $exit
#!/bin/bash
DEPLOY='/etc/puppet/environments'
REPO='hostname:/srv/git/repos/puppet.git'
SSH_KEY='/var/lib/puppet/.ssh/puppet-sync.key'

read oldrev newrev ref
BRANCH=${ref/*\/*/}

sudo -Hu puppet ansible workers \
   -a "puppet-sync --branch $BRANCH --repository $REPO --deploy $DEPLOY" \
   -o --private-key=$SSH_KEY

exit=$?
exit $exit
Up to here

- single git repository
- clone to each master (worker/CA)
- automated
Workflow

- multiple git repositories
  - librarian-puppet
  - r10k
forge "https://forgeapi.puppetlabs.com"

mod 'puppetlabs/stdlib', '4.1.0'

mod 'puppetlabs/apache', '1.1.1'
  :git => "git://github.com/puppetlabs/puppetlabs-apache.git"

mod 'puppetlabs/apt', '1.1.1'
  :git => "git://github.com/puppetlabs/puppetlabs-apt.git"
  :ref => 'any/valid/gitref'
r10k

- Uses Puppetfile
- local cache
- Configuration file: r10k.yaml

r10k.yaml

```yaml
:cachedir: '/var/cache/r10k'
:sources:
:plops:
  remote: '/var/lib/git/puppet.git'
  basedir: '/etc/puppet/environments'
```

/etc/puppet/environments/$branch

/Puppetfile
/company/modulename
```
```
```
Puppetfile
mod "puppetlabs/stdlib",
mod "os",
  :git => '/var/lib/git/puppet/stdlib
mod "middleware",
  :git => '/var/lib/git/puppet/middleware
```
```
```
```
```
```
```
```
```
```
deploy using r10k

$ r10k deploy environment -p [environment]

even better

$ sudo -u puppet !!
read oldrev newrev refname
branch=${refname#*/}/*}/
sudo -u puppet \
r10k deploy environment $branch -p
r10k Workflow

master repository

Puppetfile

Modules

forge

git

github

Modules
"He who controls the spice controls the universe"
Baron Vladimir Harkonnen
r10k Workflow

One repository per module
Modules included by Puppetfile
r10k repo controls everything
hieradata is in git also
githook pushes hiera code
hiera = exceptions

*add modules/profiles to a node*
hieradata - multiple teams

multiple backends
OS Team ⇒ JSON
App Team ⇒ YAML
WebGui ⇒ Database

You can still use an ENC too.
node thx1138 {
    class { 'role::drupal7': }  
}
"If you are editing code in /etc/puppet, you are doing it wrong." - Me
Bootable ISO

tutorial.html
Demo 2, 3 and 4
Troubleshooting

http://goo.gl/b2NlSc
Summary

Create a workflow/lifecycle for nodes
Create a workflow for code
Separate data from code: hieradata
create a class hierarchy: roles/profiles
centralize or decentralize: scale
KISS
Questions?

Comments?