CEPH deployment using puppet

Alessandro Italiano
Manual deployment

Login ➔ Server
Manual deployment

Login

- install
- conf2
- conf3
- restart
Manual deployment
Manual deployment

- Two main issues
  - Repetitive tasks: waste time
  - Host can have inconsistent state due to
    - reinstallation
    - manual change
Manual deployment, the common habits

IT team of the same computing center
Automate deployment

- Define config host1
- Define config host2
- Define config host3
- Define config host4
- Define config host5
- Define config host6
Automate deployment

Central server
- Define config host1
- Define config host2
- Define config host3
- Define config host4
- Define config host5
- Define config host6

host1

host2
Automate deployment

Central server
- Define config host1
- Define config host2
- Define config host3
- Define config host4
- Define config host5
- Define config host6

host1

host2
Automate deployment

- Define config host1
- Define config host2
- Define config host3
- Define config host4
- Define config host5
- Define config host6
Automate deployment

Central server
- Define config host1
- Define config host2
- Define config host3
- Define config host4
- Define config host5
- Define config host6

host1

host1
Automate, why it is really important

- Save time avoiding repetitive actions
- Synchronisation
- Replication
- Optimisation
- IT infrastructure under control
- Contextualisation
- Recovery
- High level definition
- Configuration language
Automate, why it is really important
Puppet, one solution!

Available free of charge
Server configuration defined at high level
Resources, classes and modules
Master/Agent, Masterless deployment
Node definition, manifest/site.pp
puppetAgent as daemon, cronJob or onDemand
Catalog compilation
Reports
Puppet basic

- Define host configuration using the puppet language syntax
- Puppet will try to apply the node configuration on the host once it has a valid catalog as result of the node definition compilation
- Puppet compile the catalog only when the agent request it
Puppet basics: resource

Define the single action to take on the target host. There are a list of builtin resource as well as user defined resource

```perl
file { '/etc/passwd':
  owner => root,
  group => root,
  mode => 644
}
```
Puppet basics: default resources

Type Reference

- augeas
- computer
- cron
- exec
- file
- filebucket
- group
- host
- interface
- k5login
- macauthorization
- mailalias
- maillist
- mcx
- mount
- nagios_command

» nagios_contact
» nagios_contactgroup
» nagios_host
» nagios_hostdependency
» nagios_hostescalation
» nagios_hostextinfo
» nagios_hostgroup
» nagios_service
» nagios_servicedependency
» nagios_serviceescalation
» nagios_serviceextinfo
» nagios_servicegroup
» nagios_timeperiod
» notify
» package
» resources

» router
» schedule
» scheduled_task
» selboolean
» selmodule
» service
» ssh_authorized_key
» sshkey
» stage
» tidy
» user
» vlan
» yumrepo
» zfs
» zone
» zpool
Puppet basics: provider

Providers implement the same resource type on different kinds of systems. They usually do this by calling out to external commands. For instance, package resources on Red Hat systems use `yum` as provider.
Classes are named blocks of Puppet code, which are not applied until they are invoked by name. They can be added to a node’s catalog by either declaring them in your manifests.

```puppet
# A class with parameters
class apache ($version = 'latest') {
  package {'httpd':
    ensure => $version, # Using the class parameter from above
    before => File['/etc/httpd.conf'],
  }

  file '/etc/httpd.conf':
    ensure => file,
    owner  => 'httpd',
    content => template('apache/httpd.conf.erb'), # Template from a

  service {'httpd':
    ensure => running,
    enable => true,
    subscribe => File['/etc/httpd.conf'],
  }
}
```
Puppet basics: module

Modules are self-contained bundles of code and data. You can write your own modules or you can download pre-built modules from the Puppet Forge.
Puppet basics: site.pp

The place [a file] where we declared all the classes or resources we wanted to apply

```ruby
# Append this at the bottom of /etc/puppetlabs/puppet/manifests/site.pp
node 'agent1.localdomain' {
  # Note the quotes around the name! Node names can have characters that aren't legal for class names, so you can't always use bare, unquoted # strings like we do with classes.
  # Any resource or class declaration can go inside here. For now:
  include apache

  class {'ntp':
    servers => [
      "ntp1.example.com dynamic",
      "ntp2.example.com dynamic",
    ],
  }
}
```
Puppet Agent/Master workflow
Organise configuration using “hiera”

• Hiera is a key/value lookup tool
• Hierarchical config lookup process
• Split configuration from puppet logic
• Fine or macro
Hierarchy definition

bash: warning: setcutter cc_error: cannot change locale (0: 0)

[[centos@puppetmaster-1 ~]$ cat /etc/puppet/hiera.yaml
# managed by puppet
---
:backends:
  - eyaml
  - yaml
:logger: console
:hierarchy:
  - secure
  - "%{::fqdn}"%{::environment}"
  - "%{::osfamily}"%{::osfamily}"
  - common
:yaml:
  :datadir: /etc/puppet/environments/%{::environment}/hieradata
:eyaml:
  :datadir: /etc/puppet/environments/%{::environment}/hieradata
  :pkcs7_private_key: /etc/puppet/keys/private_key.pkcs7.pem
  :pkcs7_public_key: /etc/puppet/keys/public_key.pkcs7.pem

:merge_behavior: deeper

[[centos@puppetmaster-1 ~]$
Puppet example

```puppet
class {
  puppet::agent:
    puppet_server  => puppet.ba.infn.it,
    environment    => production,
    splay          => true,
    puppet_run_interval => 15,
}
```
Puppet example

#cat site.pp

class { 'puppet::agent':
    puppet_server => puppet.ba.infn.it,
    environment   => production,
    splay         => true,
    puppet_run_interval => 15,
}
	node 'myserver.ba.infn.it' {

class { 'puppet::agent':
    puppet_server => puppet.ba.infn.it,
    environment   => production,
    splay         => true,
    puppet_run_interval => 30,
    version       => '3.8.1-1puppetlabs1',
}
}

It doesn’t work, duplication declaration
#cat site.pp

node 'myserver.ba.infn.it' {

class { 'puppet::agent':
    puppet_server => puppet.ba.infn.it,
    environment   => production,
    splay         => true,
    puppet_run_interval => 30,
    version       => '3.8.1-1puppetlabs1',
}
}

node default {

class { 'puppet::agent':
    puppet_server => puppet.ba.infn.it,
    environment   => production,
    splay         => true,
    puppet_run_interval => 15,
}
}
Puppet example

#cat site.pp

hiera_include('default')

#cat nodes/myserver.ba.infn.it

puppet::agent::puppet_run_interval: 30

#cat debian.yaml

puppet::agent::version: '3.8.1-1puppetlabs1'

#cat common.yaml

---
default:
  - puppet::agent

puppet::agent::puppet_server: puppet.ba.infn.it
puppet::agent::environment: production
puppet::agent::splay: true
puppet::agent::puppet_run_interval: 15
### Syntax Comparison

#### site.pp

```ruby
hiera_include('default')
```

```ruby
node 'myserver.ba.infn.it' {
  class {
    puppet::agent::puppet_run_interval: 30
    puppet::agent::puppet_server: puppet.ba.infn.it
    puppet::agent::environment: production
    puppet::agent::splay: true
  }
}
```

```ruby
node default {
  class {
    puppet::agent::puppet_run_interval: 15
  }
}
```

#### nodes/myserver.ba.infn.it

```ruby
puppet::agent::puppet_run_interval: 30
```

#### debian.yaml

```ruby
puppet::agent::version: '3.8.1-1puppetlabs1'
```

#### common.yaml

```ruby
default:  
  - puppet::agent
```
https://github.com/openstack/puppet-ceph