

RX-M

Puppet

Day 1

1. Modern Systems Configuration Management
2. Puppet Language Overview



1: Modern Systems Configuration Management

Objectives

- Describe the pressures presented by modern systems infrastructure on systems configuration management
- Examine the shortfall of traditional systems management methods
- Explore the benefits of Puppet and declarative systems configuration approaches
- Understand the implications of Infrastructure as Code [IaC]
- List some of the key technologies in the modern configuration management market
- Gain a broad conceptual understanding of Puppet
- Understand the key Puppet installation concerns

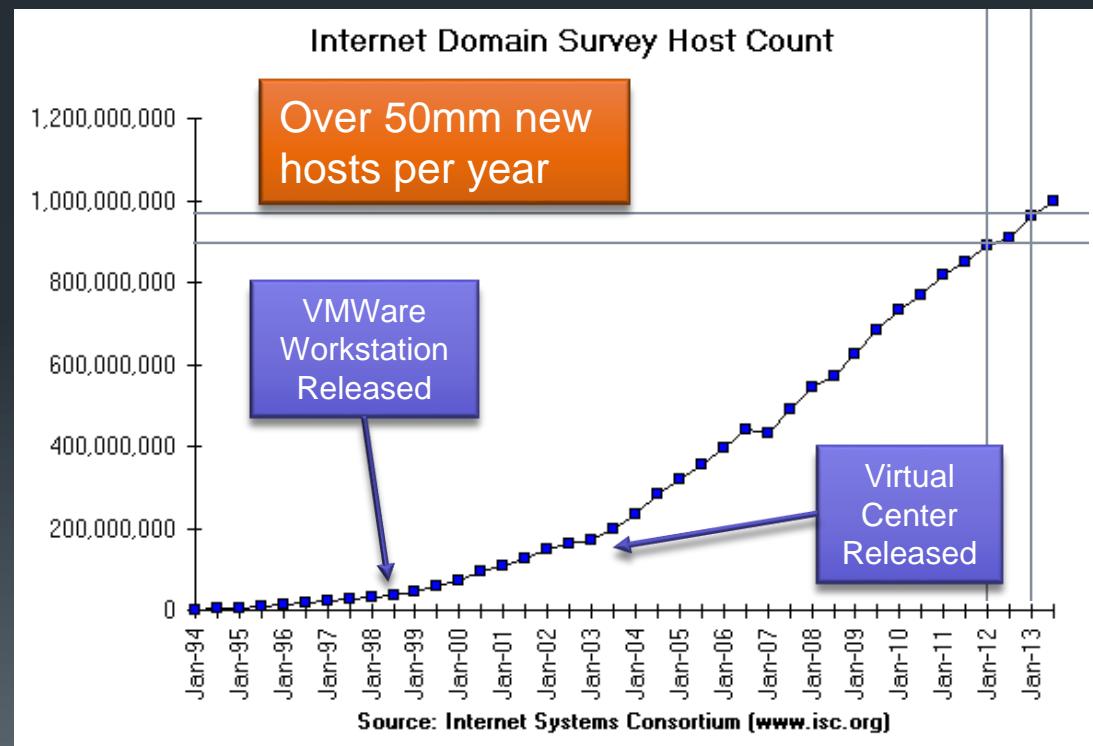
Lecture and Lab

- Our Goals in this class are two fold:
 1. Familiarize you with a large array of the most important features and issues associated with Puppet
 - This is the primary purpose of the lecture/discussion sessions
 - Focus on concepts and awareness
 2. Give you practical experience with key and complex Puppet operations
 - This is the primary purpose of the labs
 - Focus on practical Puppet usage in an enterprise setting



The Virtual Era

- Server Configuration Management faces many new challenges in the era of dynamically provisioned virtual servers
- New scaling concerns
 - Managing tens of thousands of servers is a common requirement
- Support for systems with short lifespans
 - Servers are frequently provisioned dynamically causing servers to come into existence based on load and other triggering events
- Support for a broad range of operating systems
 - Hosting facilities have little ability to standardize OS platforms and must heed market pressure to supply a range of operating systems
- IaaS, PaaS and SaaS all create new burdens on system configuration management



Custom scripts

7

Copyright 2013-2016 RX-M LLC

- The typical sys admin response to a repetitive task is to automate the task with a script
- This leads to custom-built scripts
 - Often complex, poorly documented and environment specific
 - Rarely published, documented, or reused
- Custom scripts are unlikely to scale to suit large environments
- In multi-platform environments custom scripts tend to suit only one target platform
 - A user creation script for BSD
 - A user creation script for Centos
 - A user creation script for Ubuntu
 - A user creation script for Solaris

While open source, flexible and free, custom scripts lack the scalability, community and broad contributor base of large scale open source projects

```
#!/bin/sh
#set -x

# -----
# CHECKS: is user root, did you type arguments
# -----
# Check if user is root
if [ ` /usr/bin/id | awk '{print $1}' | cut -d= -f2 | cut -d\ ( -f1 ` != 0 ]
then
    echo ; echo Sorry, you have to be root to run this script. ; echo
    exit 1
fi

# Check if 2 arguments are provided, first and last name
if [ $# != 2 ]
then
    echo ; echo Usage: `basename $0` Last_name First_name ; echo
    exit 1
fi

# --variables
SYSADM=sysadmin@domain.com
# for GCOS-FIELD
FNAME=`echo $2`
LNAME=`echo $1`

# -----
#          FUNCTIONS
# -----

# ----- Function to create 2 digits random number
rand2dig () {
    # function srand() sets new seed for random numbers
    RN=`nawk ' BEGIN { srand(); print rand() }'`

    # multiple with 100 and use int() to get 2 digit number
    IN=`(expr ${RN}*100) | bc -l | nawk '{print int($0)}'`
}
...
```

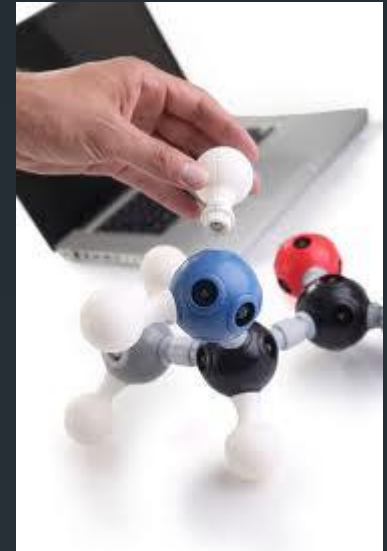
Commercial Software

- Another response to increasing configuration management burden is commercial systems management platforms
 - CA Unicenter
 - HP Opsware (etc...)
 - IBM Tivoli
 - Many others
- Commercial products are typically expensive
 - Licensing for large environments can amount to millions of dollars per year
- Commercial products have limited flexibility
 - Commercial products are not usually open source and have limited expansion options
 - Many commercial products “push” configuration changes to agents presenting challenges in dynamically provisioned virtual systems and large scale systems
- Commercial Products can be challenged at Internet scale
 - Some of the largest hosting environments have exceeded a million servers
 - -- per HP Moon Shot rollout



Modern System Management Tools

- Modern systems management platforms address challenges of scale and rapid provisioning through several key tenants
 - **Modeling** – Admins model the server configuration they want in code
 - **Pull** - Servers “pull” their configuration from centralized or distributed repositories
 - **OS Independent** - Configurations are defined generically with system specific providers implementing configuration tasks
- Separation of concerns
 - Sys Admins focus on server configurations
 - Operating systems specialists and configuration management providers focus on system specific implementations
- The Open Source Dynamic
 - Open source communities are collaborative by definition
 - This allows sys admins to readily share system configurations and collaborate
 - Also allows OS specific configuration providers to be centralized and readily shared across the broad user base of a particular systems management tool
 - Eliminates the cycle of perpetual script reproduction



Opscode Chef Jobs Demand Trend

The demand trend of job ads citing Opscode Chef as a proportion of all IT jobs with a match in the Systems Management category.

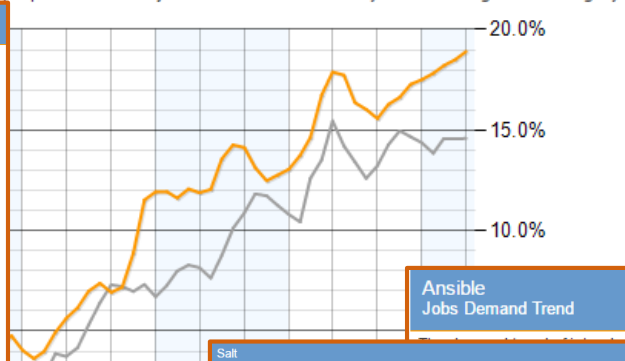
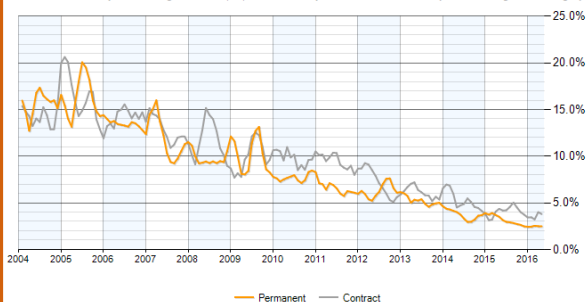
10

Copyright 2013-2016 RX-M LLC

<http://www.itjobswatch.co.uk>

Tivoli Jobs Demand Trend

The demand trend of job ads citing Tivoli as a proportion of all IT jobs with a match in the Systems Management category.



Ansible Jobs Demand Trend

Salt Jobs Demand Trend

The demand trend of job ads citing Salt as a proportion of all IT jobs with a match in the Systems Management category.



Puppet Jobs Demand Trend

The demand trend of job ads citing Puppet as a proportion of all IT jobs with a match in the Systems Management category.

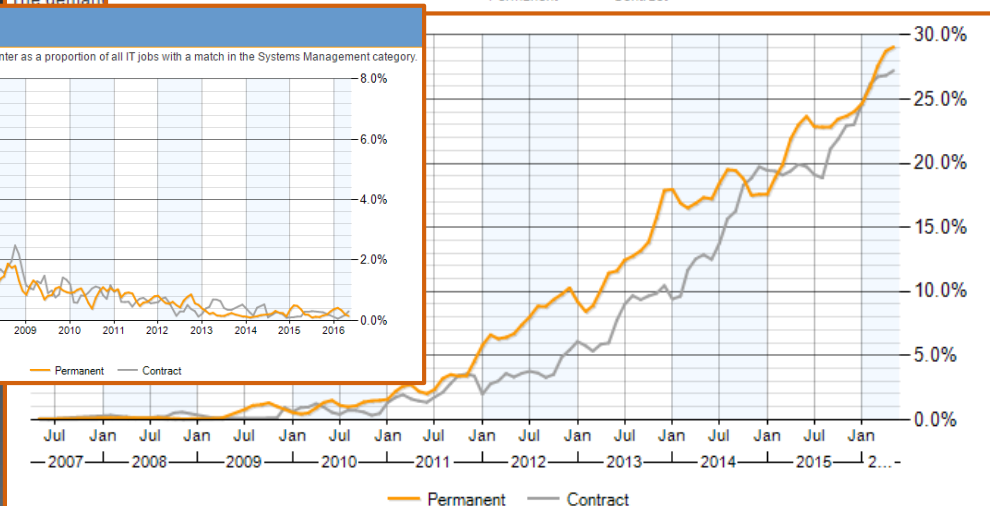
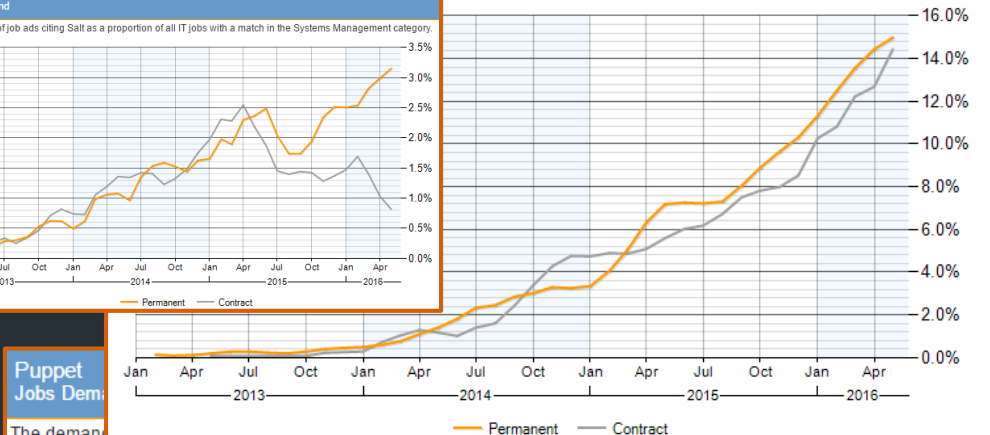
Unicenter Jobs Demand Trend

The demand trend of job ads citing Unicenter as a proportion of all IT jobs with a match in the Systems Management category.



The Systems Management Market

- There are several competing open source configuration management tools
 - Puppet, Chef and Ansible are the market leaders
 - Salt, CFEngine & others comprise a smaller segment of the market
 - Commercial products are losing share



Declarative Systems Configuration

11

Copyright 2013-2016 RX-M LLC

- Puppet defines systems configurations using declarative syntax
- Configurations are declared (not implied by sets of transient scripts)
- Configuration involves declaring what the server configuration should be
 - Creates a clear picture of the ultimate server configuration
 - The underlying configuration engine handles the system specific chores associated with meeting the configuration requirements
- Dramatically reduces the imperative code required to implement systems configurations
- Allows a single server declaration to be carried out on different operating systems
 - Making configuration declarations portable
- The Puppet declarative DSL can be extended to become your DSL
 - Custom facts
 - Custom types
 - Custom classes of servers
 - Custom resource providers
 - etc.

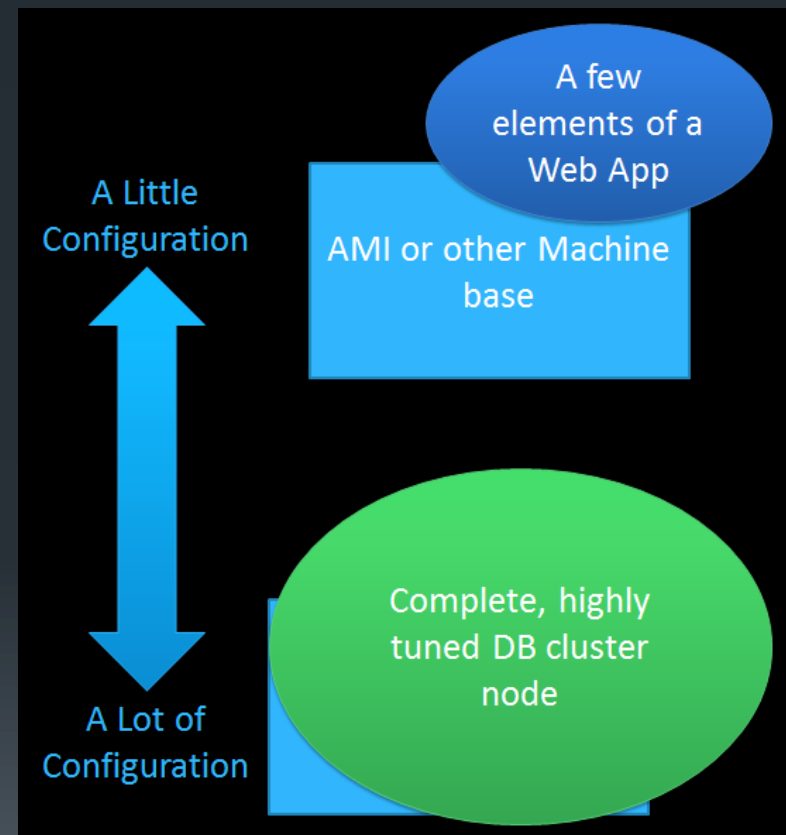
```
package { 'openssh': ensure => installed }  
service { 'ssh':  
  ensure => running,  
  enable => true,  
  require => Package['openssh'],  
}
```

Infrastructure as Code [IaC]

12

Copyright 2013-2016 RX-M LLC

- Systems can be coded to varying degrees
 - In the abstract with attention to only one or two services
 - Comprehensively with painstaking attention to detail
- Nearly everything can be modeled
 - every resource
 - every parameter
 - every relationship
 - every fact
 - for every node
- Coded Configurations Address
 - key distribution
 - monitoring
 - clustered services
 - master/slave replication
 - load balancers
 - shared filesystems
 - firewall rules
 - ...



Using the IaC approach means never manually adjusting server configuration. Just like software, the code is updated and tested on the sys admin desktop, checked in, tested and rolled out using repeatable processes and tools enabling roll-back, roll-forward, point in time recovery, etc.

What is Puppet?

- A tool for automating system administration tasks
 - A declarative language for expressing system configuration
 - An optional agent and server for distributing system configurations
 - A library for applying system configurations
 - Written in Ruby
- Open Source
 - Apache 2.0 license since v2.7
 - Surrounded by a large ecosystem of add on features and systems
 - e.g. configuration databases, management consoles, deployment tools, etc.



What does Puppet do?

- Puppet is an automated systems management tool for *nix
 - More and more support for Windows is being added with each new version, but it is not as robust as the typical *nix implementation
- Puppet performs administrative tasks based on a system specification
 - Adding users
 - Installing packages
 - Configuring services
 - Setting up network ports
 - /etc/*
 - etc.

Puppet lets you focus on how things should be configured not how to configure things

```
class ssh {  
  package { 'ssh': ensure => installed }  
  file { '/etc/ssh/sshd_config':  
    source => 'puppet:///modules/ssh/sshd_config',  
    ensure => present,  
    require => Package['ssh']  
  }  
  service { 'sshd':  
    ensure => running,  
    require => [File['/etc/ssh/sshd_config'], Package[ssh]] }  
}
```

Require SSH

What does using Puppet buy me?

15

Copyright 2013-2016 RX-M LLC

- Once services are modeled you can configure a new system by simply starting puppet
- Puppet uses configuration specifications to manufacture any number of identically configured systems
- Puppet it possible for a small team to manage thousands of systems
- Puppet promotes system consistency
- Puppet enforces good change management practices
- Puppet can manage the full life cycle
 - Configure new systems
 - Update existing systems
 - Decommission old systems
- Puppet allows you to easily
 - scale out
 - recover from hardware failures
 - Add servers, blades and VMs
- Puppet allows you to treat infrastructure as code
 - (Software Defined *, etc.)



VM Provisioning

- VM Provisioning tools build/manage machine images, spawn them and then exit the picture
- Server configuration management solutions run continuously to maintain system configuration
 - Configuration file changes are automatically applied
 - External changes countermanding the specified configuration are automatically unwound
- Virtual Machine Provisioners (e.g. VMWare vSphere, Vagrant, Kickstart, Preseed, AWS EC2) are integral to the initial setup of an actively managed node
 - Configuration management services require installation, startup, and, possibly, a network/configuration repository
 - These initialization tasks are typically handled by the VM provisioner
 - Once the virtual machine is running Puppet can bring the bulk of the configuration inline and maintain it

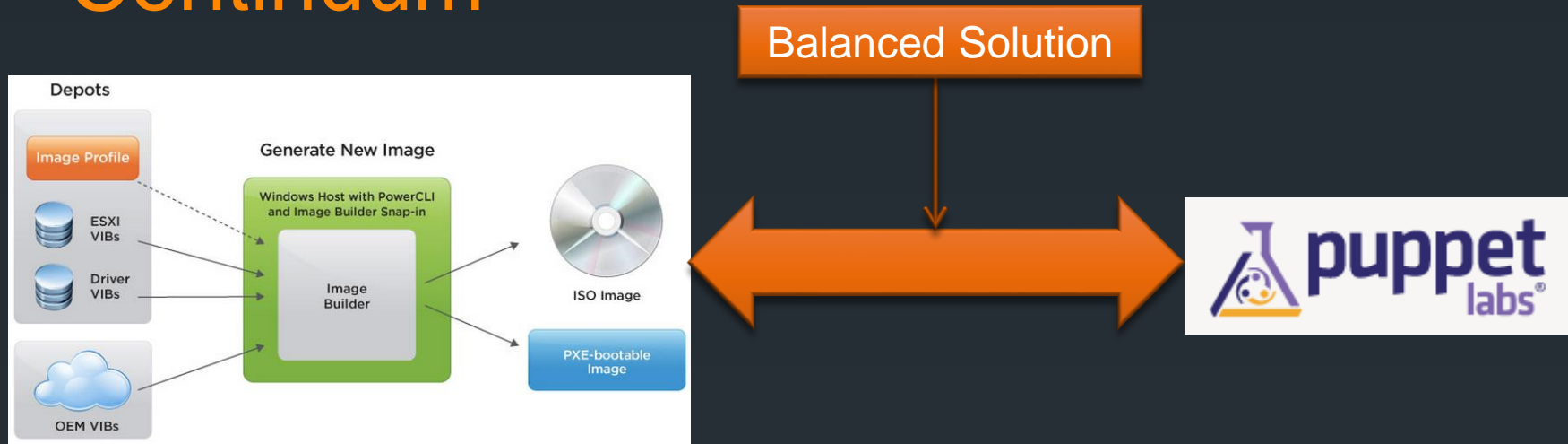
Example Vagrant Provisioning session

```
$ vagrant init precise64 http://files.vagrantup.com/precise64.box
$ vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
[default] Box 'precise64' was not found. Fetching box from specified URL for
the provider 'virtualbox'. Note that if the URL does not have
a box for this provider, you should interrupt Vagrant now and add
the box yourself. Otherwise Vagrant will attempt to download the
full box prior to discovering this error.
Downloading or copying the box...
Extracting box...ate: 679k/s, Estimated time remaining: --:--:--
Successfully added box 'precise32' with provider 'virtualbox'!
[default] Importing base box 'precise32'...
[default] Matching MAC address for NAT networking...
[default] Setting the name of the VM...
[default] Clearing any previously set forwarded ports...
[default] Creating shared folders metadata...
[default] Clearing any previously set network interfaces...
[default] Preparing network interfaces based on configuration...
[default] Forwarding ports...
[default] -- 22 => 2222 (adapter 1)
[default] Booting VM...
[default] Waiting for machine to boot. This may take a few minutes...
[default] Machine booted and ready!
[default] Mounting shared folders...
[default] -- /vagrant
<<< PUPPET Takes over HERE >>>
```


The Machine Image – CM Continuum

17

Copyright 2013-2016 RX-M LLC



- There are two extremes associated with production system deployment
 - Everything baked into the machine image (the “Golden Image”)
 - Fast to spool up
 - Many images to maintain
 - Nothing baked into the machine image
 - Slow to spool up
 - Few images to maintain
- Most solutions find a happy medium
 - Core features used across the platform baked into all machine images
 - Several base images including features which are too expensive to install on the fly
 - Everything else handled by a CM engine such as Puppet

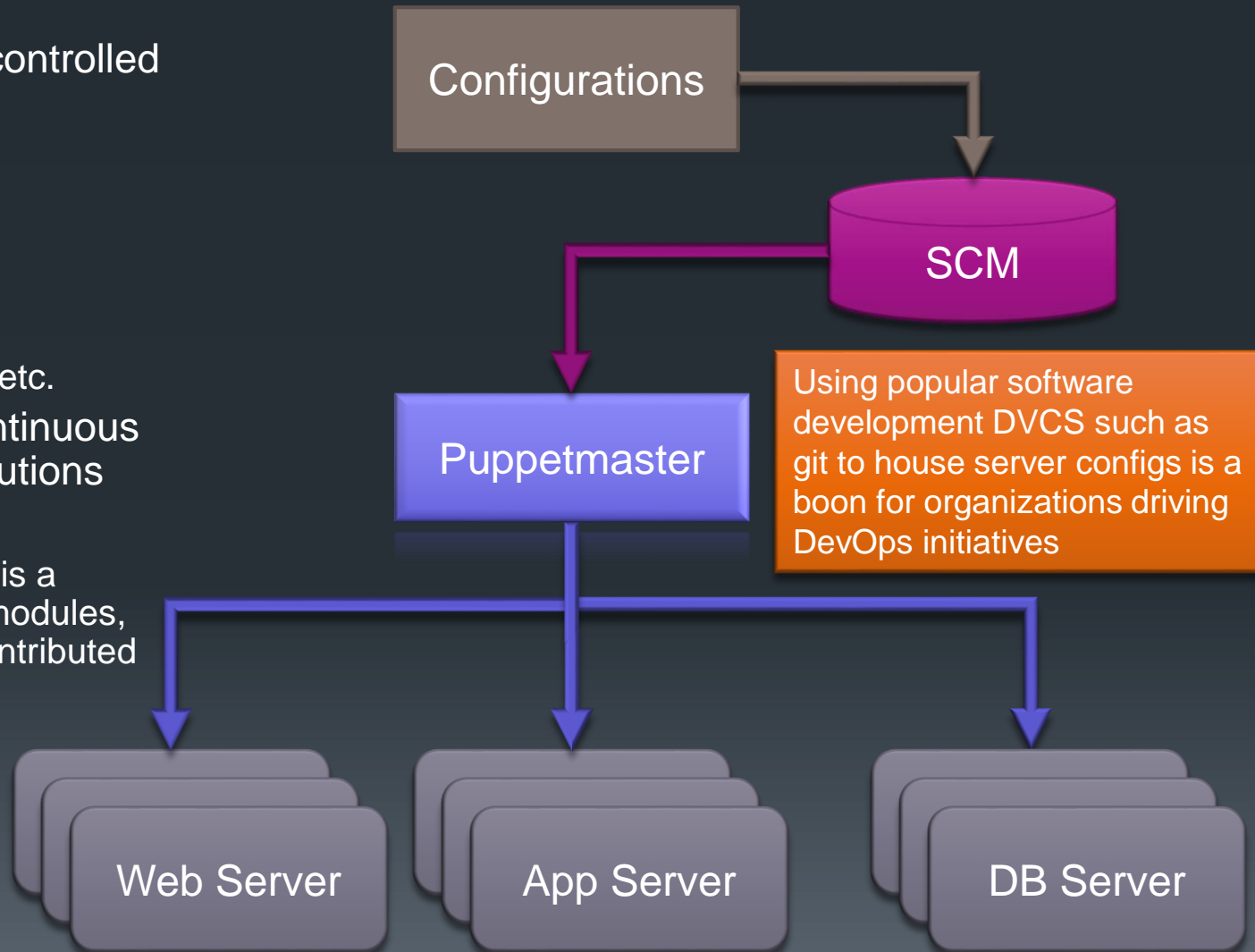
CM solutions can be used to generate the base images and apply post boot configuration on live systems

Infrastructure as Code

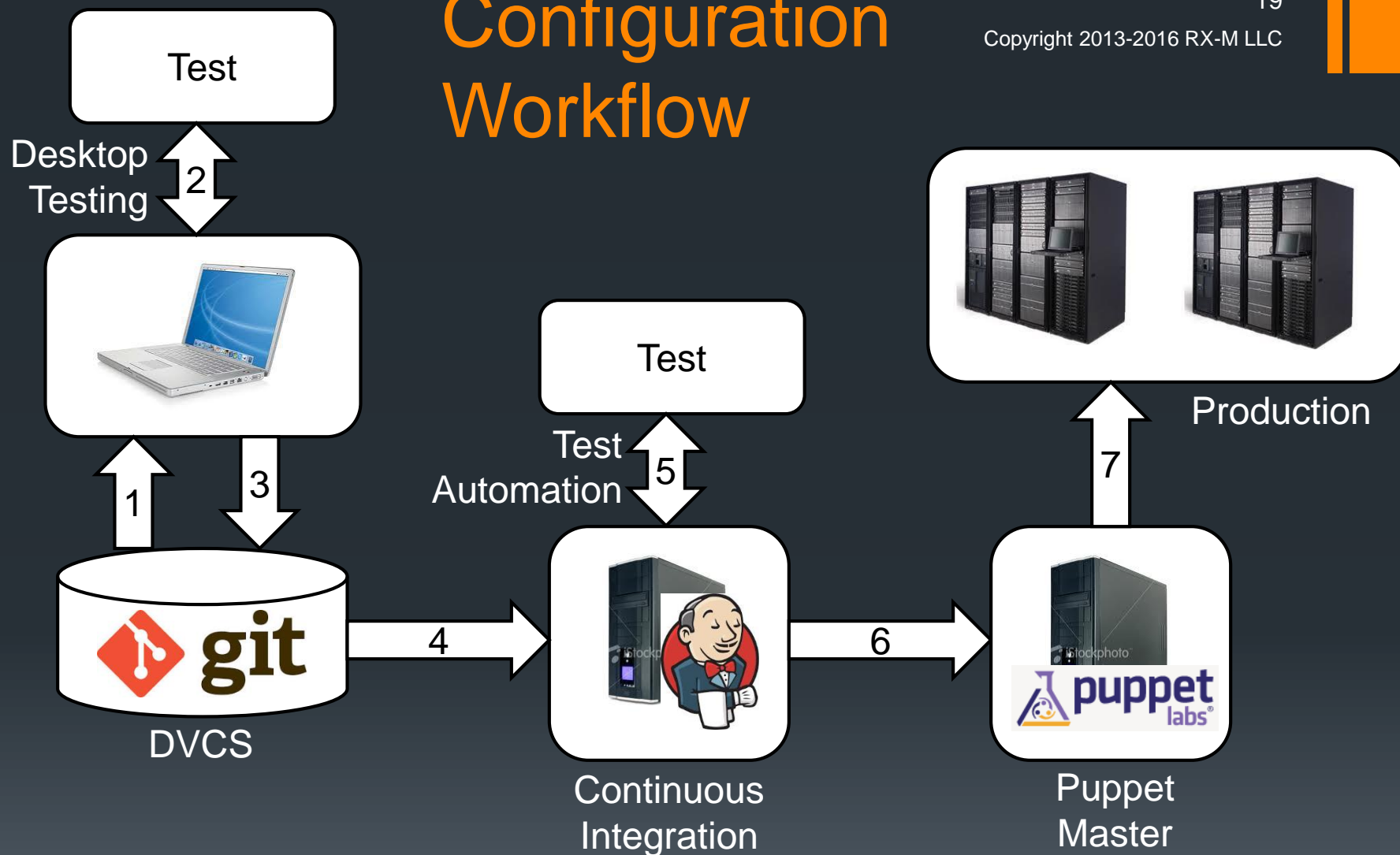
18

Copyright 2013-2016 RX-M LLC

- Code can be
 - SCM version controlled
 - Tested
 - Rolled back
 - Regressed
 - Refactored
 - Processed
 - sed/awk/grep/etc.
 - Wired into Continuous Integration solutions
 - Shared
 - Puppet Forge is a repository of modules, written and contributed by users



Configuration Workflow



Immutable Servers

20

Copyright 2013-2016 RX-M LLC

- Immutability is a property which greatly simplifies many aspects of software development and, with Puppet, server administration
 - Immutable system configurations make it much easier to test systems scenarios and reason about how a server will behave under different circumstances and in different environments
- Puppet allows you to specify how servers should be configured
 - New and existing machines can easily be brought into compliance
 - Changes to the system are automatically reverted by Puppet making the system configuration immutable
- This helps to avoid the problem of fragile Snowflake Servers
 - Servers with a one of a kind configuration
- Puppet allows the creation of Phoenix Servers
 - Servers that can be setup and torn down at will
- Immutable Servers are never modified once deployed, merely replaced with a new updated instance

Many workflows outlaw direct system configuration changes, all changes are made to the appropriate manifest and then tested, like code, prior to roll out

<http://martinfowler.com/bliki/SnowflakeServer.html>
<http://martinfowler.com/bliki/PhoenixServer.html>



Versions

21

Copyright 2013-2016 RX-M LLC

- Puppet Open Source
 - Current Release
 - Puppet 4.5.0 2016-05
 - Puppet 3.8.7 2016-04
 - V4 releases
 - 4.4.2 2016-04
 - 4.3.0 2015-11
 - 4.2.0 2015-06
 - 4.1.0 2015-05-19
 - 4.0.0 2015-04-15
 - v3 releases
 - v3.8.3 2015-09-21
 - v3.7.5 2015-03-15
 - v3.6 2014-05-15
 - v3.5 was recalled (use 3.5.1 2014-04-06)
 - v3.4 2013-12-19
 - v3.3 2013-09-12
 - v3.2 Q1 2013
 - v3.1 2012
 - V3.0 2012
 - Maintenance and Security Branches
 - Puppet 2.7 (update June 2014)
 - Puppet 2.6 (security fixes only)
 - Unsupported Releases
 - Puppet 0.25
 - Puppet 0.24
 - Puppet 0.23
- Puppet Enterprise
 - Current Release
 - PE 2016.1 [v4.4.2]
 - Maintenance and Security Branches
 - Puppet Enterprise 3.8
 - Puppet Enterprise 3.x
 - Puppet Enterprise 2.8
 - Puppet Enterprise 2.7
 - Puppet Enterprise 2.6
 - Puppet Enterprise 2.5
 - Puppet Enterprise 2.0
 - Unsupported Releases
 - Puppet Enterprise 1.2

Free	Standard	Premium support
Try out Puppet Enterprise on 10 nodes for free. Includes access to all product updates and thousands of prebuilt Puppet modules to get you started.	Get all the power of Puppet Enterprise plus our standard support package that includes bug fixes, a private knowledge base, access to all software updates and more.	Our premium support package — with extras like 24/7 and phone support for priority issues and free training — can put your mind at ease and get you up and running quickly.
\$0	Starts at \$120 per node	Contact us for pricing
Get it now	Contact us	Contact us

Semantic versioning [3.2.1]

- 3 Major, breaking changes
- 2 Minor, new features, extensions
- 1 Min, bug fixes

Puppet has employed semantic versioning since version 3.0

Puppet 2.x versus Puppet 3.x

22

Copyright 2013-2016 RX-M LLC

- Puppet is semantically versioned starting with v3.0.0 (x.y.z)
 - X must increase for major backwards-incompatible changes
 - Y may increase for backwards-compatible new functionality
 - Z may increase for bug fixes
- Puppet 3.0+ contains breaking changes that make it incompatible with Puppet 2.7.x
 - Puppet 3 adds support for Ruby 1.9.3, and drops support for Ruby 1.8.5
 - Dynamic scoping of variables, deprecated in Puppet 2.7, has been removed
 - Parameter lists in class and defined type definitions must include a dollar sign (\$) prefix for each parameter
 - `define vhost ($port = 80, $vhostdir) { ... }` not `define vhost (port = 80, vhostdir) { ... }`
 - Legacy standalone executables, which were replaced by subcommands in Puppet 2.6, have been removed (e.g. puppet is now “puppet apply”, puppetd is now “puppet agent” and puppetmasterd is now “puppet master”)
 - Various settings have been removed or harmonized
 - The API for querying resource types has changed to more closely match standard Puppet terminology
 - Most deprecated features from 2.6 have been removed
 - Puppet 3 is faster than Puppet 2.6 and significantly faster than Puppet 2.7
 - Many Solaris improvements
 - Approximately 220 bug fixes

Puppet 3 versus 4

- Improved Puppet Language
 - Puppet 3.8 future parser equivalent
- AIO (all in one) installer
 - Puppet 4
 - Facter 2.4 / CFacter 0.4
 - Hiera
 - Mcollective
 - Ruby 2.1.5
 - OpenSSL 1.0.0r
 - Gem dependencies
- Disk File structure changes
 - Executables: /opt/puppetlabs/bin
 - Code: /etc/puppetlabs/code/environments/production/manifests
 - Config: /etc/puppetlabs/puppet
 - SSL: /etc/puppetlabs/puppet/ssl
- Directory environments replace config file environments
- Drops support for Ruby 1.8.7
- REST API changes
- Puppet Server is new go forward Puppet Master platform
- Tagmail and Puppet Doc now separate modules
- Many other small changes

OSS & Enterprise Puppet

24

Copyright 2013-2016 RX-M LLC

- Puppet Open Source
 - Must setup PuppetDB, MCollective, Dashboard, puppet master, etc. yourself
 - Same code as Puppet Enterprise
- Puppet Enterprise
 - Installers/RPMs and upgrade tools to simplify and automate installation
 - Compliance module in Puppet Enterprise Console (monitor the date and time at which the system config varies from baseline)
 - Live Management (MCollective GUI for querying nodes and starting/stopping puppet on nodes)
 - node_vmware VMWare vSphere cloud provisioner (create new VM nodes from templates, list, start, stop and destroy VMs)

The screenshot shows the Puppet Enterprise Live Management console in a web browser. The URL is <https://master.dc1.puppetlabs.net/live-management/>. The interface has a top navigation bar with links for Nodes, Groups, Classes, Reports, File Search, Inventory Search, Compliance, Live Management, admin, and Help. Below this is a sub-navigation bar with 'Live Management' selected, and tabs for 'Manage Resources', 'Control Puppet', and 'Advanced Tasks'. The main content area is divided into two panels. The left panel, titled 'Live Management', contains a 'Node filter' section with a text input field (containing 'mysql'), a 'Class' dropdown menu (showing 'mysql'), a 'Fact' section with 'Fact Name' and 'Fact Value' input fields, and a 'Filter' button. Below this, it shows '1 of 12 nodes selected (8.3%)' and a 'Select all' button. The right panel, titled 'Summary', shows a 'Summary of all resource types' section with a list of resource types: group, host, package, and user. Each type has a status indicator (e.g., 'Not inspected') and a 'Inspect All' button. Below this, there is a section titled 'What is live management?' with a paragraph explaining the feature.

Live Management console

Nodes Groups Classes Reports File Search Inventory Search Compliance Live Management admin Help

Live Management Manage Resources Control Puppet Advanced Tasks

Node filter Wildcards allowed

Advanced search

Class

mysql

Fact Common fact names

Fact Name

Fact Value

Filter Reset filter

1 of 12 nodes selected (8.3%)

Select all Select none

fozzle.dc1.puppetlabs.net

Summary

Summary of all resource types

group resources

host Not inspected

package resources

host resources

Not inspected

package resources

Not inspected

user resources

Not inspected

Inspect All

What is live management?

We've harnessed the power of Puppet Enterprise and MCollective to command your nodes in real time.

The information you see on this page isn't cached from reports — it's retrieved from your infrastructure on demand, and is always up-to-date. You may notice occasional delays as we query your nodes.

The Puppet Eco System is Substantial

25

Copyright 2013-2016 RX-M LLC

- Puppet is one of the most widely adopted configuration management platforms in current use
- It is sophisticated and wired into a vast network of open source projects

The image shows two overlapping screenshots. The background screenshot is a GitHub search results page for the query 'puppet'. The search bar at the top contains 'puppet' and a 'Search' button. Below the search bar, a message states 'We've found 26,808 repository results'. On the left, there are filters for 'Repositories' (26,808), 'Code' (3,100,042), 'Issues' (64,617), and 'Users' (241). Below these are 'Languages' listed with their counts: Puppet (12,529), Ruby (6,376), Shell (1,632), HTML (721), Python (439), JavaScript (257), Perl (174), PHP (140), VimL (91), and Java (49). The top result is 'puppetlabs/puppet', described as a 'Server automation framework and application', with 3,988 stars and 1,653 forks. The foreground screenshot is the Puppet Forge homepage, with the URL 'https://forge.puppet.com'. It features a navigation bar with links to Home, Forge, Docs, Learn, Support & Services, and Contact Us. A search bar in the center says 'Search from 4,223 modules' with a 'Find' button. Below the search bar, it says 'Welcome to the Puppet Forge' and describes it as 'A repository of modules written by our community for Open Source Puppet and Puppet Enterprise IT automation software'.

GitHub, Inc. [US] <https://github.com/search?utf8=✓&q=puppet>

Pull requests Issues Gist

Search

puppet Search

We've found 26,808 repository results

Sort: Best match

Repositories 26,808

Code 3,100,042

Issues 64,617

Users 241

Languages

Puppet 12,529

Ruby 6,376

Shell 1,632

HTML 721

Python 439

JavaScript 257

Perl 174

PHP 140

VimL 91

Java 49

puppetlabs/puppet

Server automation framework and application

Ruby ★ 3,988 🍴 1,653

Updated 3 hours ago

Puppet Forge

<https://forge.puppet.com>

Home Forge Docs Learn Support & Services Contact Us

Search from 4,223 modules Find

Welcome to the Puppet Forge

A repository of modules written by our community for Open Source Puppet and Puppet Enterprise IT automation software

New CM Trends

- Containers are one of the fastest growing areas in the next generation tool space
- Containers provide “operating-system virtualization” on Linux (and windows soon)
 - OS Virtualization → PaaS
- VMs provide “hardware virtualization” on (principally) x86_
 - Machine Virtualization → IaaS
- Docker uses resource isolation features of the Linux kernel such as cgroups and kernel namespaces to allow independent containers to run within a single Linux instance
 - Not as robust as the isolation provided by VMs
 - Multitenant environments are not suited to containers for security, monitoring, billing and other reasons
- Containers enable reliable deployments like golden image VMs but launch at the speed of a process (milliseconds)



```
$ time docker run ubuntu echo hello world
hello world
real 0m0.258s
```

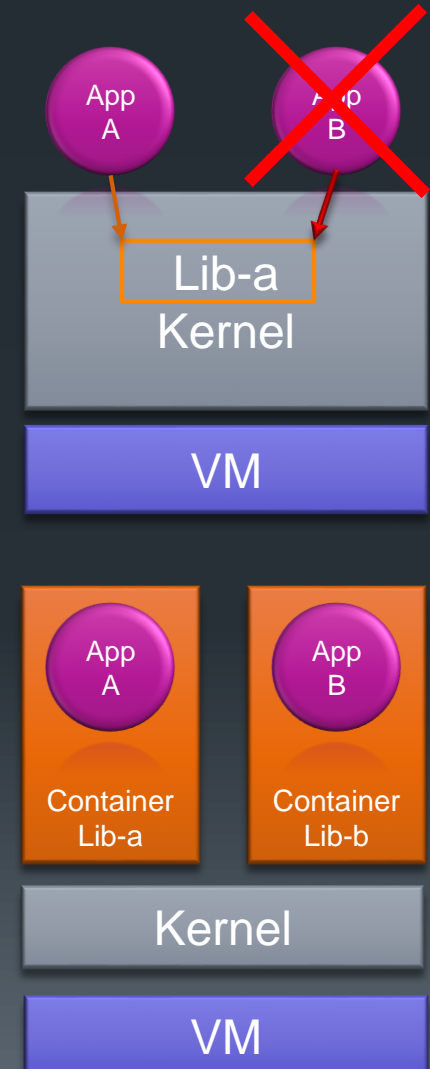
Disk usage: less than 100 kB
Memory usage: less than 1.5 MB

Containers, why do I care?

27

Copyright 2013-2016 RX-M LLC

- Containers provide a static application runtime environment creating reliable deployments
 - Fundamentally changes the approach to operations
 - Removes an entire class or extremely complex operational problems
- VMs are typically used to host infrastructure roles (e.g. Web Server, Application Server, Database Server, ...)
 - Applications running on such systems can have complex interactions with system services and other applications
 - This complexity makes it difficult to guarantee identical dev/test/prod environments, making application deployment complex and error prone
- Containers create a new layer of abstraction at the operating system level for application roles (web server, logging system, security tools, monitoring software, etc.)
 - Isolation
 - Allows multiple containerized applications to run on the same VM
 - Encapsulation
 - Each container has its own unique dependencies directly supported
 - Portability
 - Repeatable deployment across dev/test/prod environments and clouds



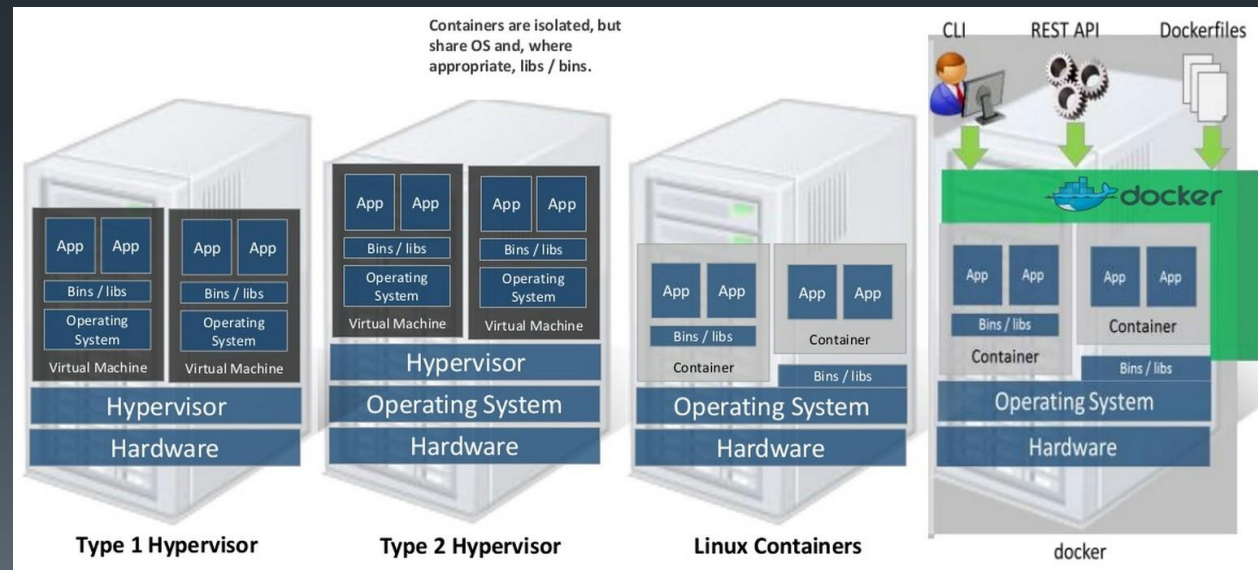
PaaS

How does Puppet work with Containers?

28

Copyright 2013-2016 RX-M LLC

- The answer is evolving
- Pioneers have different views
- Docker first appeared in Spring 2013
 - v1.0 was released in Summer 2014
- Docker hosts must be configured and this is certainly a place where Puppet can be used to good affect
- Some use Puppet to configure container internals
 - This goes against the grain of the immutable image and fast startup tenants of Docker



Puppet Dependencies

29

Copyright 2013-2016 RX-M LLC

- Puppet is a Ruby based tool
- To install Puppet a supported operating system and an appropriate Ruby version must be present
- Puppet Open Source 3.3 supported systems
 - Red Hat Enterprise Linux 5, 6 & 7 (and derivatives, CentOS, Oracle Linux, etc.)
 - Debian 6 & 7 and Ubuntu 10.04, 12.04, 12.10, 13.04
 - Fedora 18 & 19
 - SUSE Linux Enterprise Server, version 11
 - FreeBSD 4.7+ and OpenBSD 4.1+
 - Mac OS X 10.5+
 - Oracle Solaris 10+
 - AIX 5.3+
 - HP-UX
 - Windows Server 2003+
 - Client only
- Puppet 4 installs its own private ruby version [2.1.x]

Windows is different...

```
[root@CentosVM ~]# ruby --version
ruby 1.8.7 (2011-06-30 patchlevel 352) [x86_64-linux]
```

Ruby version	Puppet 2.6	Puppet 2.7	Puppet 3.x
1.8.5*	Supported	Supported	No
1.8.7	Supported	Supported	Supported
2.0.0**	No	No	Supported (3.2 and higher)
1.9.3**	No	No	Supported
1.9.2	No	No	No
1.9.1	No	No	No
1.9.0	No	No	No

* Ruby 1.8.5 is slow, 1.8.7 is preferred

** These versions have known puppet problems

Ruby

- A dynamic scripting language
- Perl influenced syntax
- Smalltalk-like object features
- Puppet is written in Ruby
- Ruby is an interpreted language and not particularly fast
 - Can only use one core
 - Uses garbage collection
 - Has many active versions
- Ruby Gems
 - Third party libraries for Ruby
- Ruby on Rails
 - Open source web framework built in Ruby
 - Restful Web Services
 - Convention over Configuration
 - The Puppet Dashboard is a Rails application



Ruby
A Programmer's Best Friend

```
# The Greeter class
class Greeter
  def initialize(name)
    @name = name.capitalize
  end

  def salute
    puts "Hello #{@name}!"
  end
end

# Create a new object
g = Greeter.new("world")

# Output "Hello World!"
g.salute
```

There are many Ruby based systems management tools, e.g. Puppet, The Foreman, Chef, Vagrant, ...

Puppet Install on Centos 6

Demo – Install Ruby

31

Copyright 2013-2016 RX-M LLC

- In this walk through we'll install a stand alone Puppet system
- Default Centos 6 package sources provide the most desirable version of Ruby

```
# yum install ruby
Loaded plugins: fastestmirror, refresh-packagekit, security
...
Installing:
  ruby                x86_64                1.8.7.352-12.el6_4                updates                534 k
Updating for dependencies:
  ruby-devel          x86_64                1.8.7.352-12.el6_4                updates                314 k
  ruby-libs           x86_64                1.8.7.352-12.el6_4                updates                1.6 M
...
Total download size: 2.5 M
Is this ok [y/N]: y
Downloading Packages:
(1/3): ruby-1.8.7.352-12.el6_4.x86_64.rpm
| 534 kB    00:00
(2/3): ruby-devel-1.8.7.352-12.el6_4.x86_64.rpm
| 314 kB    00:00
(3/3): ruby-libs-1.8.7.352-12.el6_4.x86_64.rpm
| 1.6 MB    00:02
Installed:
  ruby.x86_64 0:1.8.7.352-12.el6_4
Dependency Updated:
  ruby-devel.x86_64 0:1.8.7.352-12.el6_4
  ruby-libs.x86_64 0:1.8.7.352-12.el6_4
Complete!
[root@mysql01 mysql01]# ruby --version
ruby 1.8.7 (2011-06-30 patchlevel 352) [x86_64-linux]
```

Version specific install:

```
$ yum install ruby-1.8.7 rubygems-1.8.24
```

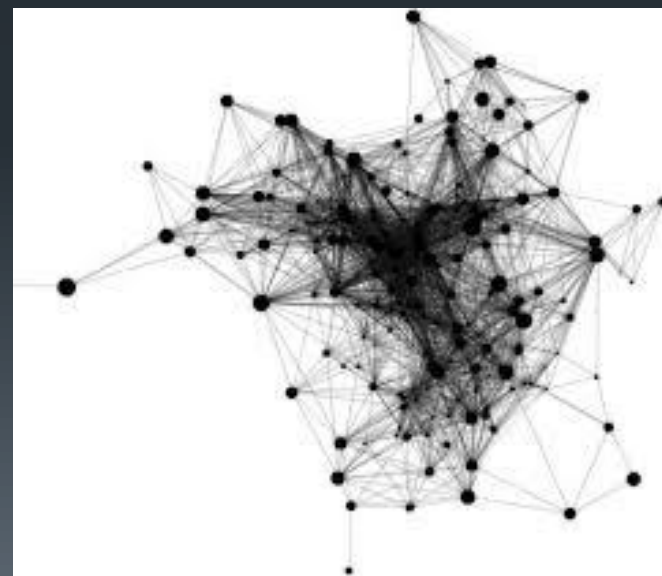
Puppet Install on Centos

Demo – Install Puppet

```
login as: root
root@192.168.1.115's password:
Last login: Thu Aug 22 19:05:11 2013
[root@CentosVM ~]# rpm -ivh http://yum.puppetlabs.com/el/6/products/i386/puppetlabs-release-6-7.noarch.rpm
Retrieving http://yum.puppetlabs.com/el/6/products/i386/puppetlabs-release-6-7.noarch.rpm
warning: /var/tmp/rpm-tmp.SFrvYu: Header V4 RSA/SHA1 Signature, key ID 4bd6ec30: NOKEY
Preparing... ##### [100%]
 1:puppetlabs-release ##### [100%]
[root@CentosVM ~]# yum install puppet
...
puppetlabs-deps | 1.9 kB 00:00
puppetlabs-deps/primary_db | 20 kB 00:00
puppetlabs-products | 1.9 kB 00:00
puppetlabs-products/primary_db | 95 kB 00:00
Setting up Install Process
...
Total download size: 3.3 M
Installed size: 8.9 M
Is this ok [y/N]: y
Downloading Packages:
(1/12): augeas-libs-0.9.0-4.el6.x86_64.rpm | 317 kB 00:01
(2/12): facter-1.7.2-1.el6.x86_64.rpm | 83 kB 00:00
(3/12): hiera-1.2.1-1.el6.noarch.rpm | 21 kB 00:00
(4/12): libselinux-ruby-2.0.94-5.3.el6_4.1.x86_64.rpm | 99 kB 00:00
(5/12): puppet-3.2.4-1.el6.noarch.rpm | 1.0 MB 00:03
(6/12): ruby-augeas-0.4.1-1.el6.x86_64.rpm | 21 kB 00:00
(7/12): ruby-irb-1.8.7.352-12.el6_4.x86_64.rpm | 313 kB 00:01
(8/12): ruby-rdoc-1.8.7.352-12.el6_4.x86_64.rpm | 376 kB 00:01
(9/12): ruby-rgen-0.6.5-1.el6.noarch.rpm | 87 kB 00:00
(10/12): ruby-shadow-1.4.1-13.el6.x86_64.rpm | 11 kB 00:00
(11/12): rubygem-json-1.5.5-1.el6.x86_64.rpm | 763 kB 00:01
(12/12): rubygems-1.3.7-1.el6.noarch.rpm | 206 kB 00:00
-----
Total 247 kB/s | 3.3 MB 00:13
...
Installed:
 puppet.noarch 0:3.2.4-1.el6
Dependency Installed:
 augeas-libs.x86_64 0:0.9.0-4.el6 facter.x86_64 1:1.7.2-1.el6
 hiera.noarch 0:1.2.1-1.el6 ruby-irb.x86_64 0:1.8.7.352-12.el6_4
 libselinux-ruby.x86_64 0:2.0.94-5.3.el6_4.1 ruby-augeas.x86_64 0:0.4.1-1.el6
 ruby-rdoc.x86_64 0:1.8.7.352-12.el6_4 ruby-rgen.noarch 0:0.6.5-1.el6
 rubygems.noarch 0:1.3.7-1.el6 ruby-shadow.x86_64 0:1.4.1-13.el6
 rubygem-json.x86_64 0:1.5.5-1.el6
Complete!
[root@CentosVM ~]# ls -l /etc/init.d/puppet
-rwxr-xr-x. 1 root root 2649 Aug 14 15:00 /etc/init.d/puppet
[root@CentosVM ~]#
```


Puppet Installation

- Puppet managed systems are called Nodes
- Puppet Configurations
 - Puppet Nodes can be managed standalone with no external dependencies
 - Puppet nodes can also use an agent daemon to pull configuration data down from a Puppet Master service running on another system
 - Puppet Standalone, Agent and Master installations are similar
- Most distribution packaging systems divide Master and Agent into separate packages
 - Standalone installations only require the Agent Package
 - Some distributions provide a puppet-common package which includes the core Puppet features shared by Agent and Master (this is also suitable for standalone systems)
- Some distributions may require Ruby installation
- Occasionally other packages are required



Puppet on OpenSUSE

34

Copyright 2013-2016 RX-M LLC

- Installing the Puppet on SLES 13.1
 - Download the OpenSUSE ISO (example uses 13.1)
 - <http://software.opensuse.org/131/en>
 - Create a new VM and configure the CDROM device to use the ISO
 - Boot the VM
 - Follow the installation steps accepting the defaults
 - Update the system using YaST or zypper
 - `$ sudo zypper up`
 - Install Puppet
 - `$ sudo zypper install puppet` (installs 3.2.4)
- Installing the latest version of Puppet on SLES 13.1
 - Complete the above steps (you can skip the final puppet install step)
 - Add the latest SLES Puppet repo to zipper
 - `$ sudo zypper addrepo -f http://download.opensuse.org/repositories/systemsmanagement:puppet/openSUSE_13.1/systemsmanagement:puppet.repo`
 - Install puppet 3.5.1
 - `$ sudo zypper install puppet-3.5.1-2.1.i586` (installs puppet 3.5.1 and Ruby 2.0)
- Installing the latest version of Puppet on SLES 11SP3 (note: openSUSE 11 is EOL)
 - Download the OpenSUSE ISO (example uses 11.3)
 - <http://ftp5.gwdg.de/pub/opensuse/discontinued/distribution/11.3/iso/>
 - Create a new VM and configure the CDROM device to use the ISO
 - Boot the VM and follow the installation steps accepting the defaults
 - Add the puppet labs repo using YaST or zypper
 - `$ sudo zypper addrepo -f http://download.opensuse.org/repositories/systemsmanagement:puppet/SLE_11_SP3/systemsmanagement:puppet.repo`
 - Install Puppet
 - `$ sudo zypper install puppet`
 - (or for a specific version: `$ sudo zypper install puppet-3.5.1-2.1.i586`)



■ Package Sources:

■ Oracle Packages

- <http://pkg.oracle.com/solaris/release/en/search.shtml>

■ OpenCSW

- <http://www.opencsw.org/packages/CSWpuppet3>

- `pkgadd -d http://get.opencsw.org/now`

- `/opt/csw/bin/pkgutil -U`

- `/opt/csw/bin/pkgutil -y -i puppet3`

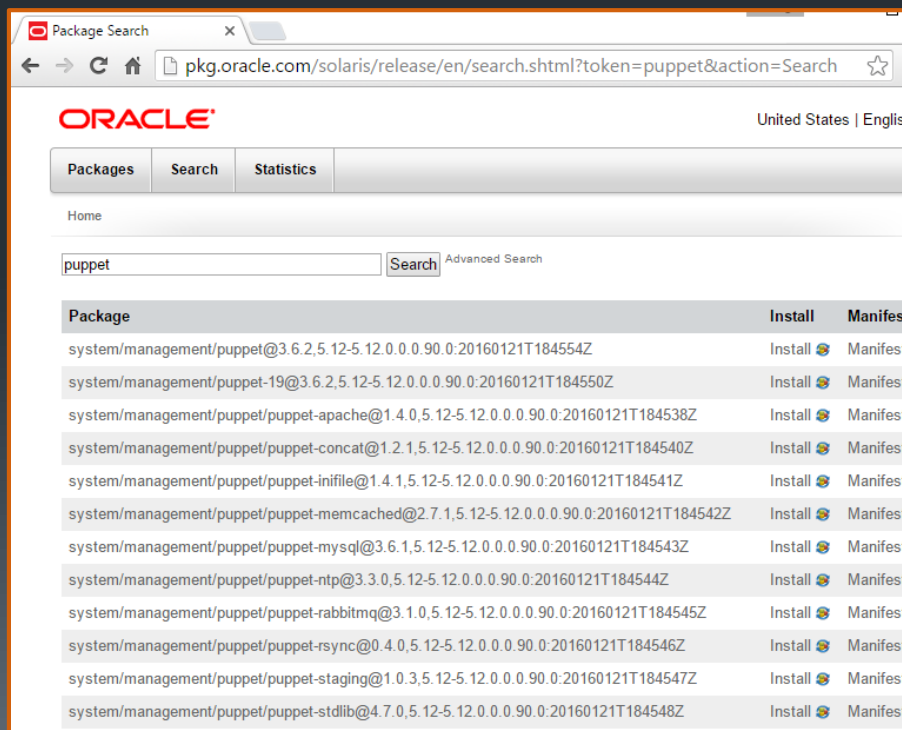
- `/usr/sbin/pkgchk -L CSWpuppet3`

- Puppet 3.7.4 [2015-02]

- Solaris 10/11

Puppet on Solaris

35



The screenshot shows the Oracle Package Search interface. The search term 'puppet' has been entered, and a list of packages is displayed. Each row includes the package name, version, architecture, and a link to the manifest file.

Package	Install	Manifests
system/management/puppet@3.6.2.5.12-5.12.0.0.0.90.0:20160121T184554Z	Install	Manifests
system/management/puppet-19@3.6.2.5.12-5.12.0.0.0.90.0:20160121T184550Z	Install	Manifests
system/management/puppet/puppet-apache@1.4.0.5.12-5.12.0.0.0.90.0:20160121T184538Z	Install	Manifests
system/management/puppet/puppet-concat@1.2.1.5.12-5.12.0.0.0.90.0:20160121T184540Z	Install	Manifests
system/management/puppet/puppet-inifile@1.4.1.5.12-5.12.0.0.0.90.0:20160121T184541Z	Install	Manifests
system/management/puppet/puppet-memcached@2.7.1.5.12-5.12.0.0.0.90.0:20160121T184542Z	Install	Manifests
system/management/puppet/puppet-mysql@3.6.1.5.12-5.12.0.0.0.90.0:20160121T184543Z	Install	Manifests
system/management/puppet/puppet-ntp@3.3.0.5.12-5.12.0.0.0.90.0:20160121T184544Z	Install	Manifests
system/management/puppet/puppet-rabbitmq@3.1.0.5.12-5.12.0.0.0.90.0:20160121T184545Z	Install	Manifests
system/management/puppet/puppet-rsync@0.4.0.5.12-5.12.0.0.0.90.0:20160121T184546Z	Install	Manifests
system/management/puppet/puppet-staging@1.0.3.5.12-5.12.0.0.0.90.0:20160121T184547Z	Install	Manifests
system/management/puppet/puppet-stdlib@4.7.0.5.12-5.12.0.0.0.90.0:20160121T184548Z	Install	Manifests

```

root@solaris:~# /opt/csw/bin/puppet --version
3.7.4
root@solaris:~# /opt/csw/bin/puppet config print confdir
/etc/puppet
root@solaris:~# puppet resource user
-bash: puppet: command not found
root@solaris:~# clear
root@solaris:~# /opt/csw/bin/puppet --version
3.7.4
root@solaris:~# /opt/csw/bin/puppet config print confdir
/etc/puppet
root@solaris:~# /opt/csw/bin/puppet resource user
user { 'adm':
  ensure           => 'present',
  comment          => 'Admin',
  gid              => '4',
  groups           => ['sys', 'tty', 'lp'],
  home             => '/var/adm',
  password         => 'NP',
  password_max_age => '-1',
  password_min_age => '-1',
  uid              => '4',
}
user { 'aiuser':
  ensure           => 'present',
  comment          => 'AI User',
  gid              => '61',
  home             => '/',
  password         => '*LK*',
  password_max_age => '-1',
  password_min_age => '-1',
  uid              => '61',
}

```

Summary

- Modern systems management solutions have challenging requirements
 - Support for tens of thousands of servers
 - Support for dynamically provisioned virtual servers
 - Support for modern provisioning and testing sensibilities
 - BDD/TDD, CI, CD
- Modern systems management solutions treat infrastructure as Code
 - Generic declarative server configuration
 - Platform specific implementations
 - Pull based server configuration
 - Support for distributed multi-master configuration version control
 - Well aligned with DevOps initiatives

Note that Virtual Box network adapters on the host may interfere with IPv4 DHCP acquisition on the VMWare VM. If you have this problem (VM with IP6 but no IP4 address) remove virtual box.

37

Copyright 2013-2016 RX-M LLC

VMs

- There are two preconfigured VMs provided with the course
- All passwords are “puppet”
 - Exception: LDAP passwords are “puppetmaster”
- All non root user accounts are “puppet”
- Node A
 - Centos 7 Minimal
- Node B
 - Ubuntu 14.04 Minimal

These virtual machines have a virtual network interface with a generated MAC address. The MAC address must be unique on a given network if the NIC is bridged. To ensure that your lab VMs do not run into network conflicts with other machines in the lab you will need to follow the steps below to create a unique MAC address for each.

- Load the VM into the VMWare library (use “Open a Virtual Machine” from the library window)
- Make sure the machine is not running and open the settings dialog (right or control click the machine and choose settings)
- In the network adapters => advanced dialog choose “generate” to create a new MAC address and choose ok to close all of the dialogs

This completes the Ubuntu setup, for Centos you must also complete the following steps

- Boot the Centos VM and login as root
- Remove the persistent network information file:
`rm -f /etc/udev/rules.d/70-persistent-net.rules`
- Edit the eth0 configuration and change the HWADDR to match the newly generated MAC address for your VM:
`vi /etc/sysconfig/network-scripts/ifcfg-eth0`
- Reboot

This completes the Centos setup

Lab 1

- Installing Puppet on multiple operating systems
- [optional] Test puppet with a VM provisioning system



2: Puppet Language Overview

Objectives

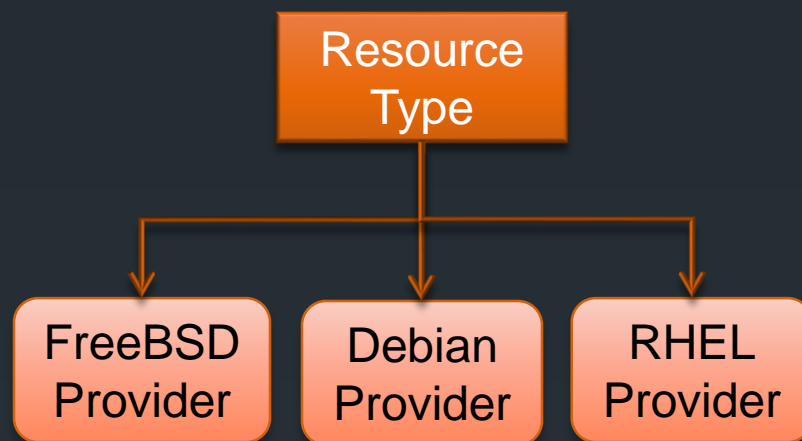
- Describe the basic features of Puppet
- Understand Manifests and the Puppet CM DSL
- Explain the benefits of declarative configuration
- Gain experience creating manifests and working with resources
- Explore the various Puppet resource types and the nature of resource providers

Core Puppet Facilities

- Deployment Configuration Language
 - A declarative language used to describe server configurations
- Resource Abstraction Layer
 - A set of implementation libraries built behind a high level abstraction allowing declarative configuration elements to be implemented on a variety of platforms
- Transactional Layer
 - A logic layer which identifies configuration deltas and then invokes the appropriate configuration resource needed to bring the server inline with the declared configuration

Resource Abstraction

- Puppet treats resources as abstract system features
- All resources are instances of resource types
 - Resource types have providers which are platform specific implementations of the type
- The resource abstraction model allows type based configuration to be independent of the underlying OS
- Resource instances have
 - Type
 - Title
 - Parameters
 - Key/value pairs defining the type's attributes
- Puppet's configuration language is declarative not imperative
 - You describe the configuration you want and Puppet performs the appropriate actions



Puppet Resource Declaration

```
user { 'randy':  
  ensure => present,  
  uid => '501',  
  gid => 'admin',  
  shell => '/bin/bash',  
  home => '/home/randy',  
  managehome => true,  
}
```

Puppet is Powerful

43

Copyright 2013-2016 RX-M LLC

- Puppet can configure (or ruin) just about any aspect of your system configuration
 - “puppet” is the root command line tool
 - The “resource” sub command allows you to list and change system resources
 - “puppet resource service” displays all services on the system, running or otherwise
 - “puppet describe --list” displays all of the resource types available (additional types can be supported through plug-ins)

```
[root@CentosVM ~]# puppet describe --list
```

These are the types known to puppet:

augeas	- Apply a change or an array of changes to the ...
computer	- Computer object management using DirectorySer ...
cron	- Installs and manages cron jobs
exec	- Executes external commands
file	- Manages files, including their content, owner ...
filebucket	- A repository for storing and retrieving file ...
group	- Manage groups
host	- Installs and manages host entries
interface	- This represents a router or switch interface
k5login	- Manage the `.k5login` file for a user
...	

```
[root@CentosVM ~]# puppet resource service
```

```
...
service { 'crond':
  ensure => 'running',
  enable => 'true',
}
service { 'hadoop-datanode':
  ensure => 'stopped',
  enable => 'false',
}
service { 'httpd':
  ensure => 'stopped',
  enable => 'false',
}
service { 'ip6tables':
  ensure => 'running',
  enable => 'true',
}
service { 'iptables':
  ensure => 'running',
  enable => 'true',
}
service { 'puppet':
  ensure => 'stopped',
  enable => 'false',
}
service { 'ypbind':
  ensure => 'stopped',
  enable => 'false',
}
[root@CentosVM ~]#
```

Resource
The Puppet
term for
anything
that can be
managed

Built-in Puppet 3/4 Types

- augeas
- computer
- cron
- exec
- **file**
- filebucket
- group
- host
- interface
- k5login
- macauthorization
- mailalias
- maillist
- mcx
- mount
- notify
- **package**
- resources
- router
- schedule
- scheduled_task
- selboolean
- selmodule
- **service**
- ssh_authorized_key
- sshkey
- stage
- tidy
- user
- vlan
- yumrepo
- zfs
- zone
- zpool
- 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

- Types are the cornerstone of the Puppet declarative DSL
- Adding custom types is an important way to extend Puppet functionality

Puppet Versus Scripts

45

Copyright 2013-2016 RX-M LLC

- In the previous example we added a user to the system
- Imagine you have 1,000 systems that need to have this user added
- With Puppet you would add the user resource to a configuration manifest used by the servers
 - This would typically be handled just like a critical software development task
 - The change is tested locally then added to a manifest which is committed to a DVCS
 - The change is pulled in integration test and verified (possibly in a CI framework)
 - The change is pulled in prod (possibly in a CD framework)
 - All servers using the manifest pull it from the production repository and apply it based on a schedule or notification system
- The IaC approach treats infrastructure updates like software product updates
- Puppet agent/master services handle:
 - Connection and Security between nodes and the master manifest repository
 - Installer and other platform details: apt-get, yum, pkgadd, ports, etc.
 - Testing current state
 - Determining appropriate configuration for each node

Consider the scripting necessary to implement the below Puppet declarations

```
package {'openssh-server':  
  ensure => installed,  
}  
  
service {'ssh':  
  ensure => running,  
  enable => true,  
  require => Package['openssh-server'],  
}
```

Describing Resource Types

- The “puppet describe” command can be used to gather information on puppet types (-s for short version)
 - Available Parameters
 - Available Providers
- The online Puppet type reference is also a useful resource
 - #1 link hit by sys admins
 - <http://docs.puppetlabs.com/references/latest/type.html>

```
[root@CentosVM ~]# puppet describe user
```

```
...
```

```
- **managehome**
```

```
Whether to manage the home directory when managing the user.
```

```
This will create the home directory when `ensure => present`, and delete the home directory when `ensure => absent`. Defaults to `false`. Valid values are `true`, `false`.
```

```
[root@CentosVM ~]# puppet describe -s user
```

```
user
```

```
====
```

```
Manage users. This type is mostly built to manage system users, so it is lacking some features useful for managing normal users.
```

```
This resource type uses the prescribed native tools for creating groups and generally uses POSIX APIs for retrieving information about them. It does not directly modify `/etc/passwd` or anything.
```

```
**Autorequires:** If Puppet is managing the user's primary group (as provided in the `gid` attribute), the user resource will autorequire that group. If Puppet is managing any role accounts corresponding to the user's roles, the user resource will autorequire those role accounts.
```

```
Parameters
```

```
-----
```

```
allowdupe, attribute_membership, attributes, auth_membership, auths, comment, ensure, expiry, forcelocal, gid, groups, home, ia_load_module, iterations, key_membership, keys, managehome, membership, name, password, password_max_age, password_min_age, profile_membership, profiles, project, role_membership, roles, salt, shell, system, uid
```

```
Providers
```

```
-----
```

```
aix, directoryservice, hpuxuseradd, ldap, pw, user_role_add, useradd, windows_adsi
```

Inspecting Resources

47

Copyright 2013-2016 RX-M LLC

- The Puppet resource command allows you to list and change resources
 - `puppet resource <TYPE> [<NAME>] [ATTRIBUTE=VALUE ...]`

```
[root@CentosVM ~]# puppet resource user randy
user { 'randy':
  ensure      => 'present',
  gid         => '500',
  home        => '/home/randy',
  password    => '$6$R8L0oe2s$VdqBXKr1CxilSv8rpg69ySyQ4yC/ojbvWKYoW4Yv31iG5CRMmnxzhEFmm53F5oQe7tkK7WTjGxsP772z1',
  password_max_age => '99999',
  password_min_age => '0',
  shell       => '/bin/bash',
  uid         => '500',
}
[root@CentosVM ~]# puppet resource user randy shell='/bin/sh'
Notice: /User[randy]/shell: shell changed '/bin/bash' to '/bin/sh'
user { 'randy':
  ensure => 'present',
  shell  => '/bin/sh',
}
[root@CentosVM ~]# puppet resource user randy
user { 'randy':
  ensure      => 'present',
  gid         => '500',
  home        => '/home/randy',
  password    => '$6$R8L0oe2s$VdqBXKr1CxilSv8rpg69ySyQ4yC/ojbvWKYoW4Yv31iG5CRMmnxzhEFmm53F5oQe7tkK7WTjGxsP772z1',
  password_max_age => '99999',
  password_min_age => '0',
  shell       => '/bin/sh',
  uid         => '500',
}
```

```
[root@CentosVM ~]# su randy
sh-4.1$ ps
10499 pts/0    00:00:00 sh
sh-4.1$ exit
[root@CentosVM ~]# puppet resource user randy shell='/bin/bash'
Notice: /User[randy]/shell: shell changed '/bin/sh' to '/bin/bash'
user { 'randy':
  ensure => 'present',
  shell  => '/bin/bash',
}
[root@CentosVM ~]# su randy
[randy@CentosVM root]$ ps
10549 pts/0    00:00:00 bash
```

Key Resources Managed

- Some of the more important resources typically managed by Puppet include:
 - **Files**
 - **Services**
 - **Packages**
 - **Users**
 - **Groups**
 - **Cron jobs**

All of these are listed on the cheat sheet

When all else fails, arbitrary commands can be executed with the exec type

The exec type should not be overused as it side steps many of the benefits provided by declarative configuration

Resource Providers

49

Copyright 2013-2016 RX-M LLC

- Puppet's cross-platform support is implemented through Resource Providers
- Providers are back-ends that offer support for a specific implementation of a given resource type
- A provider's main job is to supply a generic wrapper for OS specific client-side tools
 - Packages - There are more than 20 package providers (dpkg/apt, rpm/yum, etc.)]
 - Services - Identifying which services are running
 - Files – Setting up and managing configuration files
 - Facts - Evaluating logic based on system state in manifests
- Not all resource types have or need providers, but any resource type concerned with portability will likely have providers
- Choosing the correct provider for a resource requires knowledge of the operating system type, possibly the version and often other details

- Providers manage the implementation details associated with resource declarations
 - You say what
 - Providers figure out how
- To use a resource with Puppet you must have a provider for that resource type native to each of your target systems

```
[root@nodea ~]# puppet describe user
```

user

====

Manage users. This type is mostly built to manage system users, so it is lacking some features useful for managing normal users.

This resource type uses the prescribed native tools for creating groups and generally uses POSIX APIs for retrieving information about them. It does not directly modify `/etc/passwd` or anything.

****Autorequires:**** If Puppet is managing the user's primary group (as provided in the ``gid`` attribute), the user resource will autorequire that group. If Puppet is managing any role accounts corresponding to the user's roles, the user resource will autorequire those role accounts.

Parameters

- ****allowdupe****
Whether to allow duplicate UIDs. Defaults to ``false``.

...

Providers

aix, directoryservice, hpuxuseradd, ldap, pw, user_role_add, useradd, windows_adsi

Manifest Syntax

- Puppet uses its own declarative configuration language
- Designed to be accessible to sys admins
- Inspired by the Nagios configuration file format
- The core of the Puppet language is declaring resources
- Every other part of the language exists to add flexibility and convenience to the way resources are declared
- Resources can be defined in any order and are not applied in the order declared
- Resource dependencies must be explicitly declared

```
class apache::service (
  $service_enable = true,
) {
  validate_bool($service_enable)

  service { 'httpd':
    ensure => $service_enable,
    name   => $apache::apache_name,
    enable => $service_enable,
  }
}
```

When Puppet applies a compiled manifest it will:

- Read the current state of the resource
- Compare the current state to the desired state
- If necessary, change the system to enforce the desired state

System Configuration Modeling

51

Copyright 2013-2016 RX-M LLC

- Creating a Node configuration in puppet is equivalent to modeling the desired system
- Resources are the fundamental unit for modeling system configurations
- The block of Puppet code that describes a resource is called a resource declaration
- Declaring a resource instructs Puppet to include it in the “catalog” and manage its state on the target system
- A catalog is the complete set of configuration instructions provided to puppet for a node

Resource Declaration General Form

- The resource type (lower-case)
- An opening curly brace
- The title (string)
- A colon
- Optionally, any number of attribute and value pairs
 - attribute name (a bare word)
 - => (arrow, aka fat comma, aka hash rocket)
 - value (of appropriate data type)
 - trailing comma (note that the comma is optional after the final attribute/value pair)
- Optionally, a semicolon, followed by another title, colon, and attribute block
- A closing curly brace

```
package { 'vim':  
  ensure => installed  
}
```

```
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;  
}
```

```
service { 'ssh':  
  ensure => 'running',  
  enable => true,  
  require => Package["openssh-server"],  
}
```

Manifests

52

Copyright 2013-2016 RX-M LLC

- Puppet Manifests are files containing resource declarations (usually with a .pp extension)
- Simple manifests declare the resources desired for a particular system
 - Manifests may also contain conditional logic and other more complex statements
- The “puppet apply” subcommand brings the system inline with the manifest file
 - Puppet is idempotent, you can apply the same manifest as many times as you like
 - If the system configuration matches the manifest no action is taken
- In Puppet:
 - Whitespace is fungible
 - Bare words are typically treated as strings
 - Single quotes define explicit strings
 - Double quotes define strings which allow variable substitution

```
[root@CentosVM ~]# id fred
id: fred: No such user
[root@CentosVM ~]# vi addfred.pp
[root@CentosVM ~]# cat addfred.pp
user { 'fred':
  ensure      => 'present',
  gid         => '500',
  home        => '/home/fred',
  password    => '',
  password_max_age => '99999',
  password_min_age => '0',
  shell       => '/bin/bash',
  uid         => '508',
}
[root@CentosVM ~]# puppet apply addfred.pp
Notice: /User[fred]/ensure: created
Notice: Finished catalog run in 0.10 seconds
[root@CentosVM ~]# id fred
uid=508(fred) gid=500(randy) groups=500(randy)
[root@CentosVM ~]# puppet apply addfred.pp
Notice: Finished catalog run in 0.03 seconds
[root@CentosVM ~]# id fred
uid=508(fred) gid=500(randy) groups=500(randy)
[root@CentosVM ~]# vi addfred.pp
[root@CentosVM ~]# cat addfred.pp
user { 'fred':
  ensure      => 'absent',
}
[root@CentosVM ~]# puppet apply addfred.pp
Notice: /User[fred]/ensure: removed
Notice: Finished catalog run in 0.09 seconds
[root@CentosVM ~]# id fred
id: fred: No such user
[root@CentosVM ~]#
```

Type and Title

53

Copyright 2013-2016 RX-M LLC

- Puppet refers to specific resources by type and title
 - Package["openssl"]
 - User["randy"]
- This notation is used when referencing resources in configuration files
 - Creating explicit dependencies, etc.
- Titles must be unique per resource type
 - You may have a package and a service both titled "fred" but you may only have one service titled "fred"

Attributes

- Attributes describe the desired state of the resource
- Each resource type has its own set of attributes
- Most types have a short list of key attributes and a larger number of optional attributes
- Many attributes have default values
- Every attribute you declare must have a value
- Attributes that can take multiple values generally accept them as an array

```
file { 'greeting':  
  path      => '/tmp/greeting',  
  ensure    => present,  
  mode      => 0640,  
  content   => "Welcome to Puppet.",  
}
```

Reserved Words

- Puppet has several reserved words which
 - Cannot be used as bare word strings (must be quoted to use as strings)
 - Cannot be used as names for custom functions
 - Cannot be used as names for custom resource types or defined resource types
 - Cannot be used as names for classes
 - May be used as names for attributes in custom resource types in Puppet 3+
- Reserved Words
 - Language Keywords
 - case
 - class
 - default
 - define
 - else
 - elsif
 - if
 - import
 - inherits
 - node
 - unless
 - Expression Operators
 - and
 - in
 - or
 - Boolean value
 - false
 - true
 - Special value
 - undef

In essence it is best to avoid keyword usage anywhere except as keywords, just because you can doesn't mean you should

- # Dependencies

- ```
[root@CentosVM ~]# curl http://localhost | head
curl: (7) couldn't connect to host
[root@CentosVM ~]# cat apache.pp
package { 'httpd':
 ensure => 'installed',
 before => File['/etc/httpd/conf/httpd.conf']
}

file { ['/etc/httpd/conf/httpd.conf':
 ensure => 'present',
 source => '/root/puppet/files/httpd.conf'
}

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

[root@CentosVM ~]# puppet apply apache.pp
Notice: /Stage[main]//Package[httpd]/ensure: created
Notice: /Stage[main]//Service[httpd]/ensure: ensure changed 'stopped' to
'running'
Notice: Finished catalog run in 3.52 seconds
[root@CentosVM ~]# curl http://localhost | head
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<head>
<title>Apache HTTP Server Test Page powered by CentOS</title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<style type="text/css">
 body {
 background-color: #fff;
 color: #000;
 font-size: 0.9em;
 font-family: sans-serif,helvetica;
```

# The Package Resource

56

Copyright 2013-2016 RX-M LLC

- Package resources are used to manage software packages
- There is a dichotomy in package support
  - Some package types (e.g., yum and apt) can retrieve their own package files
  - Others (e.g., rpm and sun) cannot
  - Package formats that cannot retrieve their own files can use the source parameter to point to the correct file
- Puppet will automatically guess the packaging format that you are using based on the platform you are on, but you can override it using the provider parameter
  - Each provider defines what it requires in order to function, and you must meet those requirements to use a given provider
- Attributes

```
package { 'resource title':
 name => # (namevar) The package name
 ensure => # What state the package should be in (present|absent|purged|held|latest)
 adminfile => # A file containing package defaults
 allowcdrom => # Tells apt to allow cdrom sources (true|false)
 category => # A read-only parameter set by the package
 configfiles => # Whether configfiles should be kept or replaced (keep|replace)
 description => # A read-only parameter set by the package
 flavor => # OpenBSD supports 'flavors', which are further specifications of a package type
 install_options => # An array of additional options to pass
 instance => # A read-only parameter set by the package
 platform => # A read-only parameter set by the package
 provider => # The specific backend to use for this package
 responsefile => # A file containing any necessary answers to questions
 root => # A read-only parameter set by the package
 source => # Where to find the package, must be a local file or URL
 status => # A read-only parameter set by the package
 uninstall_options => # An array of additional options to pass when installing
 vendor => # A read-only parameter set by the package
 # ...plus any applicable metaparameters.
}
```

```
$ssl = $operatingsystem ? {
 solaris => SMCssl,
 default => openssl
}
```

```
package { $ssl:
 ensure => installed,
 alias => openssl
}
```



# Metaparameters

57

Copyright 2013-2016 RX-M LLC

- Attributes that can be used with every resource type
- Don't map directly to system state
- Most commonly used metaparameters are for specifying order relationships between resources
- Metaparameters
  - **alias** - Creates an alternate name for the resource
  - **audit** - Marks a subset of this resource's unmanaged attributes for auditing. Accepts an attribute name, an array of attribute names, or all
  - **before** - References to one or more objects that depend on this object
  - **loglevel** - Sets the level that information will be logged (debug | info | notice | warning | err | alert | emerg | crit | verbose)
  - **noop** - Boolean flag indicating whether work should actually be done (true | false)
  - **notify** - References to one or more objects that depend on this object and require refresh on change
  - **require** - References to one or more objects that this object depends on
  - **schedule** - On what schedule the object should be managed (must supply a schedule object)
  - **stage** - Which run stage a given resource should reside in
  - **subscribe** - References to one or more objects that this object depends on and is refreshed by
  - **tag** - Add the specified tags to the associated resource (several resources can share the same user defined tags, useful when creating manifest logic)

```
file { '/etc/ssh/sshd_config':
 owner => root,
 group => root,
 alias => 'sshdconfig',
}

File['sshdconfig'] {
 mode => 644,
}

schedule { 'daily':
 period => daily,
 range => '2-4'
}

exec { '/usr/bin/apt-get update':
 schedule => 'daily'
}

file { '/etc/hosts':
 ensure => file,
 source => '/tmp/hosts',
 mode => 0644,
 tag => ['bootstrap', 'minrun'],
}
```

```
#apply only resources with the bootstrap tag
puppet agent --test --tags bootstrap
```

# Refreshing

58

Copyright 2013-2016 RX-M LLC

- The notify metaparameter allows a resource to “refresh” other dependent resources when it changes
- The subscribe metaparameter allows a resource to “refresh” when one of its dependencies changes
- Only three built in resource types can refresh
  - Service - refresh by restarting the service
  - Mount - refresh by unmounting and then remounting the volume
  - Exec - usually does not refresh
    - To force refresh use “refreshonly => true” which causes the exec to never fire unless it receives a refresh event

## Autorequire

- Some resource types will notice when an instance is related to other resources
- This causes the set up of automatic dependencies
- Most common is the autorequire between files and their parent directories
  - If a file and its parent directory are both being managed as resources the parent directory will be managed before the file
  - This never creates new resources; it only adds dependencies to resources that are already being managed
- Explicit dependencies override autorequires

# The File Resource

59

Copyright 2013-2016 RX-M LLC

- Manages files, including their content, ownership, and permissions
- Can manage normal files, directories, and symlinks (symlinks cannot be managed on Windows)
- File contents can be managed directly with the content attribute, or downloaded from a remote source using the source attribute
  - The source attribute can be used to recursively serve directories (when the recurse attribute is set to true or local)
- Attributes

```
file { 'resource title':
 path => # (namevar) The path to the file to manage. Must be fully...
 ensure => # Whether to create files that don't currently...
 backup => # Whether (and how) file content should be backed...
 checksum => # The checksum type to use when determining...
 content => # The desired contents of a file, as a string...
 ctime => # A read-only state to check the file ctime. On...
 force => # Perform the file operation even if it will...
 group => # Which group should own the file. Argument can...
 ignore => # A parameter which omits action on files matching
 links => # How to handle links during file actions. During
 mode => # The desired permissions mode for the file, in...
 mtime => # A read-only state to check the file mtime. On...
 owner => # The user to whom the file should belong...
 provider => # The specific backend to use for this `file...
 purge => # Whether unmanaged files should be purged. This...
 recurse => # Whether and how deeply to do recursive...
 recurselimit => # How deeply to do recursive management. Values...
 replace => # Whether to replace a file or symlink that...
 selinux_ignore_defaults => # If this is set then Puppet will not ask SELinux...
 selrange => # What the SELinux range component of the context...
 selrole => # What the SELinux role component of the context...
 seltype => # What the SELinux type component of the context...
 seluser => # What the SELinux user component of the context...
 show_diff => # Whether to display differences when the file...
 source => # A source file, which will be copied into place...
 sourceselect => # Whether to copy all valid sources, or just the...
 target => # The target for creating a link. Currently...
 type => # A read-only state to check the file...
 # ...plus any applicable metaparameters.
}
```

The recurse attribute is particularly useful allowing you to apply user/group/mode settings to an entire subdirectory tree

```
file { '/etc/ssh/sshd_config':
 ensure => file,
 mode => 600,
 source => '/root/var/sshd_config',
}
```

# The Service Resource

60

Copyright 2013-2016 RX-M LLC

- Manages running services
- Service support varies widely by platform (can require very few attributes to manage properly on some platforms and a number of attributes to manage properly on other platforms)
- Puppet 2.7 and newer expect init scripts to have a working status command
  - If this isn't the case for any of your services you must set `hasstatus` to `false` and possibly specify a custom status command in the `status` attribute
- If a service receives a refresh event from another resource, the service will get restarted
  - The command to restart the service depends on the platform
  - You can provide a command explicitly with the `restart` attribute or you can set `hasrestart` to `true` to use the init script's restart command; if you do neither, the service's stop and start commands will be used

```
service { 'resource title':
 name => # (namevar) The name of the service to run. This name is...
 ensure => # Whether a service should be running. Valid...
 binary => # The path to the daemon. This is only used for...
 control => # The control variable used to manage services...
 enable => # Whether a service should be enabled to start at...
 hasrestart => # Specify that an init script has a `restart...
 hasstatus => # Declare whether the service's init script has a...
 manifest => # Specify a command to config a service, or a path
 path => # The search path for finding init scripts....
 pattern => # The pattern to search for in the process table...
 provider => # The specific backend to use for this `service...
 restart => # Specify a *restart* command manually. If left...
 start => # Specify a *start* command manually. Most...
 status => # Specify a *status* command manually. This...
 stop => # Specify a *stop* command...
 # ...plus any applicable metaparameters.
}
```

```
if $service_ensure == 'running' {
 $ensure_real = 'running'
 $enable_real = true
} else {
 $ensure_real = 'stopped'
 $enable_real = false
}
```

```
service { rabbitmq:
 ensure => $ensure_real,
 enable => $enable_real,
 hasstatus => true,
 hasrestart => true,
}
```

# The Exec Resource

61

Copyright 2013-2016 RX-M LLC

- Executes external commands
- Commands executed must be idempotent (typically using checks to avoid running unless some condition is met)
- You can restrict an exec to only run when it receives refresh events by using the refreshonly parameter
- There is a strong tendency to use exec to do work Puppet can't already do; yet it is highly recommended that execs be considered for migration to native Puppet types to gain the benefits of
  - Declarative syntax
  - DRY (don't repeat yourself)
  - Easy reuse
- If Puppet is managing an exec's cwd or the executable file used in an exec's command, the exec resource will autorequire those files
- If Puppet is managing the user that an exec should run as, the exec resource will autorequire that user

```
exec { 'resource title':
 command => # (namevar) The actual command to execute. Must either be...
 creates => # A file to look for before running the command...
 cwd => # The directory from which to run the command. If
 environment => # Any additional environment variables you want to
 group => # The group to run the command as. This seems to...
 logoutput => # Whether to log command output in addition to...
 onlyif => # If this parameter is set, then this `exec` will...
 path => # The search path used for command execution...
 provider => # The specific backend to use for this `exec`...
 refresh => # How to refresh this command. By default, the...
 refreshonly => # The command should only be run as a refresh...
 returns => # The expected return code(s). An error will be...
 timeout => # The maximum time the command should take. If...
 tries => # The number of times execution of the command...
 try_sleep => # The time to sleep in seconds between...
 unless => # If this parameter is set, then this `exec` will...
 user => # The user to run the command as...
 # ...plus any applicable metaparameters.
}
```

```
exec {'tar -xf /Volumes/nfs02/important.tar':
 cwd => '/var/tmp',
 creates => '/var/tmp/myfile',
 path => ['/usr/bin', '/usr/sbin']
}
```

# Managed and Unmanaged configuration

62

Copyright 2013-2016 RX-M LLC

- Resources not identified in a Puppet manifest are not managed by Puppet
- Attributes not mentioned in a manifest for managed resources are also unmanaged
- If the “ensure” attribute is not present, other attributes are only applied if the resource exists

```
[root@CentosVM ~]# cat rc.pp
file {'/tmp/rxmdata.conf.d':
 mode => 0700,
}
[root@CentosVM ~]# ls -la /tmp/rxmdata.conf.d
total 8
drwx-----. 2 root root 4096 Aug 22 23:42 .
drwxrwxrwt. 6 root root 4096 Aug 22 23:42 ..
[root@CentosVM ~]# chmod 0755 /tmp/rxmdata.conf.d/
[root@CentosVM ~]# ls -la /tmp/rxmdata.conf.d
total 8
drwxr-xr-x. 2 root root 4096 Aug 22 23:42 .
drwxrwxrwt. 6 root root 4096 Aug 22 23:42 ..
[root@CentosVM ~]# puppet apply rc.pp
Notice: /Stage[main]/File[/tmp/rxmdata.conf.d]/mode: mode changed '0755' to '0700'
Notice: Finished catalog run in 0.03 seconds
[root@CentosVM ~]# ls -la /tmp/rxmdata.conf.d
total 8
drwx-----. 2 root root 4096 Aug 22 23:42 .
drwxrwxrwt. 6 root root 4096 Aug 22 23:42 ..
[root@CentosVM ~]# rm -rf /tmp/rxmdata.conf.d/
[root@CentosVM ~]# ls -la /tmp/rxmdata.conf.d
ls: cannot access /tmp/rxmdata.conf.d: No such file or directory
[root@CentosVM ~]# puppet apply rc.pp
Notice: Finished catalog run in 0.04 seconds
[root@CentosVM ~]# ls -la /tmp/rxmdata.conf.d
ls: cannot access /tmp/rxmdata.conf.d: No such file or directory
[root@CentosVM ~]#
```

# Previewing Changes

63

Copyright 2013-2016 RX-M LLC

- You can preview Puppet configuration changes before implementing them
- This is very helpful when making delicate adjustments to critical systems
- This is also a great tool to use while learning Puppet
- The `--noop` switch tells Puppet not to perform the operation
- The `--show_diff` switch tells puppet to display the changes which [are | would] be made
- Both switches can be used with inline commands and manifest apply commands

```
[root@NodeA ~]# puppet resource user ann ensure='present' --noop --show_diff
Notice: /User[ann]/ensure: current_value absent, should be present (noop)
user { 'ann':
 ensure => 'absent',
}
[root@NodeA ~]# puppet resource user ann ensure='present'
Notice: /User[ann]/ensure: created
user { 'ann':
 ensure => 'present',
}
```

# Variables

64

Copyright 2013-2016 RX-M LLC

- Variable names are prefixed with `$`
- Values are assigned with `=`
- Single quoted strings are explicit literals
- Double quoted strings allow variable replacement
  - Called interpolation in Puppet
  - Variables can optionally be surrounded by curly braces within quotes
    - `"${greeting_file}"`
    - Can eliminate ambiguity and allows variables to be placed directly next to non-whitespace characters
- The `notify{}` resource statement allows you to output messages on the agent/apply node
- The `notice()` function produces output in the server (master) log

```
$greeting_file = 'greeting'
$greeting_mode = 0640

file { $greeting_file:
 path => '/tmp/greeting',
 ensure => present,
 mode => $greeting_mode,
 content => 'Welcome to Puppet.',
}
```

```
root@NodeB:~/manifests# cat var.pp
$msg = "Welcome to Puppet\n"
```

```
notify{'The message is ${msg}': }
notify{"The message is ${msg}": }
```

```
root@NodeB:~/manifests# puppet apply var.pp
```

```
Notice: Compiled catalog for nodeb.att.net in environment production in 0.01 seconds
```

```
Notice: The message is Welcome to Puppet
```

```
Notice: /Stage[main]/::Notify[The message is Welcome to Puppet]/message: defined 'message' as 'The message is Welcome to Puppet'
```

```
Notice: The message is ${msg}
```

```
Notice: /Stage[main]/::Notify[The message is ${msg}]/message: defined 'message' as 'The message is ${msg}'
```

```
Notice: Finished catalog run in 0.02 seconds
```

```
$
```



# Puppet Data Types

- Booleans – may be true or false
  - Values are coerced to Boolean as follows:
    - Empty strings are false, all other strings are true
    - All numbers are true (!)
    - undef is false
    - Arrays and hashes are true
    - Resource references are true
- Undef – like undefined, nil or null in other languages
- Strings – text of any length
  - Bare Words are usually treated as unquoted strings
  - Escape sequences are supported: `\$, \", \', \\, \n, \r, \t, \s`
  - Quoted strings may continue over multiple lines
- Resource References – identified by type and title (e.g. `User["randy"]`)
- Numbers (Integer, Float, and Numeric) – may be written as bare words or quoted strings
- Arrays - comma-separated lists of items surrounded by square brackets
  - Zero based indexed (e.g. `$myArray[2]` #retrieve the 3<sup>rd</sup> element)
- Hashes – key => value pairs surrounded by curly braces
  - e.g. `$ports = { http => 80, https => 443 }` `$ports[https]`
- Regular Expressions
  - e.g. `if $host =~ /^www(\d+)\./ { notify { "Welcome web server #1": } }`
  - `=~` the regular expression match operator (`!~` for not match)
  - `$0` for the entire match, `$1` for the first capture, etc.

# Arrays

66

Copyright 2013-2016 RX-M LLC

- Many attributes can have an array of values
- In most cases Puppet attempts to expand arrays when they are encountered in resource definitions
  - e.g. one resource can require an array of other resources
- Resources can also be given an array of titles
  - Puppet treats this as multiple resource declarations
  - In this case all of the titled resources share the same set of attributes
- Arrays are enclosed in square brackets and their values are comma separated

```
file { ['/etc',
 '/etc/rc.d',
 '/etc/rc.d/init.d',
 '/etc/rc.d/rc0.d',
 '/etc/rc.d/rc1.d',
 '/etc/rc.d/rc2.d',
 '/etc/rc.d/rc3.d',
 '/etc/rc.d/rc4.d',
 '/etc/rc.d/rc5.d',
 '/etc/rc.d/rc6.d'],
 ensure => directory,
 owner => 'root',
 group => 'root',
 mode => 0755,
 }
```

We will look at installing and using puppetlabs-stdlib in a later chapter

## array functions from puppetlabs-stdlib

delete  
delete\_at  
flatten  
grep  
hash  
is\_array  
join  
member  
prefix  
range  
reverse  
shuffle  
size  
sort  
unique  
validate\_array  
values\_at  
zip

# Facts

67

Copyright 2013-2016 RX-M LLC

- Puppet defines many built in variables called facts
- The Puppet tool called Facter discovers system information and then loads it into a set of variables

```
facter
...
disks => {
 sda => {
 model => "VMware Virtual S",
 size => "20.00 GiB",
 size_bytes => 21474836480,
 vendor => "VMware,"
 },
 sr0 => {
 model => "VMware SATA CD01",
 size => "1.00 GiB",
 size_bytes => 1073741312,
 vendor => "NECVMWar"
 }
}
dmi => {
 bios => {
 release_date => "07/02/2015",
 vendor => "Phoenix Technologies LTD",
 version => "6.00"
 },
 board => {
 manufacturer => "Intel Corporation",
 product => "440BX Desktop Reference P1",
 serial_number => "None"
 },
 chassis => {
 asset_tag => "No Asset Tag",
 type => "Other"
 },
 manufacturer => "VMware, Inc.",
 ...
```

```
root@ubuntu:~# vim fac.pp
root@ubuntu:~# cat fac.pp
$user_name = "randy"
notify {$user_name;}

file {'user.txt':
 path => '/tmp/user.txt',
 content => "
 User ${user_name}
 IP ${ipaddress}
 Host ${fqdn}
 OS ${operatingsystem} ${operatingsystemrelease}
 Puppet ${puppetversion}"
}

root@ubuntu:~# puppet apply fac.pp
Notice: Compiled catalog for ubuntu.localdomain in environment production
in 0.25 seconds
Notice: randy
Notice: /Stage[main]/Main/Notify[randy]/message: defined 'message' as
'randy'
Notice: /Stage[main]/Main/File[user.txt]/ensure: defined content as
'{md5}90a399957d87cd4399dc0174f18464ef'
Notice: Applied catalog in 0.01 seconds
root@ubuntu:~# cat /tmp/user.txt

User randy
IP 192.168.131.145
Host ubuntu.localdomain
OS Ubuntu 14.04
Puppet 4.2.2
```

# Conditional Statements

- Puppet supports a range of conditional expressions

- if/elsif/else

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

- unless

```
unless condition { code }
```

- case

```
case condition {
 case1: { code }
 case2: { code }
 default: { code }
}
```

- Selectors

```
variable = selector {
 case1 => result,
 case2 => result,
 default => result,
```

```
if str2bool($is_virtual) {
 service {'ntpd':
 ensure => stopped,
 enable => false,
 }
} else {
 service { 'ntpd':
 name => 'ntpd',
 ensure => running,
 enable => true,
 hasrestart => true,
 require => Package['ntp'],
 }
}

unless $memorysize > 1024 {
 $maxclient = 500
}

case $operatingsystem {
 centos: { $apache = 'httpd' }
 redhat: { $apache = 'httpd' }
 debian: { $apache = 'apache2' }
 ubuntu: { $apache = 'apache2' }
 default: { fail("Unrecognized OS") }
}

$apache = $operatingsystem ? {
 centos => 'httpd',
 redhat => 'httpd',
 ubuntu => 'apache2',
 default => undef,
}
```

# Puppet Expressions

- Puppet Operator Precedence from highest to lowest:
  - ! (logical not)
  - in (true if the right operand contains the left operand)
  - \* and / (multiplication and division)
  - - and + (addition and subtraction)
  - << and >> (left shift and right shift)
  - == and != (equal and not equal)
  - >=, <=, >, and < (greater or equal, less or equal, greater than, and less than)
  - and (logical and)
  - or (logical or)

```
5 < 9
($operatingsystem != 'Solaris')
$kernel in ['linux', 'solaris']
!str2bool($is_virtual)
```

- Puppet provides support for built in and custom functions
  - e.g. str2bool()
- Puppet functions are coded in Ruby
- More on functions in later chapters

- Arithmetic Operators
  - + (addition)
  - - (subtraction)
  - / (division)
  - \* (multiplication)
  - % (modulo)
  - << (left shift)
  - >> (right shift)

# Immutable system configuration

- Puppet will repair systems if their configuration goes awry
- This gives system configuration the appearance of immutability

```
[root@CentosVM ~]# adduser fred
Creating mailbox file: File exists
[root@CentosVM ~]# id fred
uid=501(fred) gid=501(fred) groups=501(fred)
[root@CentosVM ~]# puppet apply addfred.pp
Notice: /User[fred]/ensure: removed
Notice: Finished catalog run in 0.08 seconds
[root@CentosVM ~]# id fred
id: fred: No such user
[root@CentosVM ~]#
```

- Multiple resources can be declared in a manifest
  - Each must have a unique title
- Most resources are: ensure =>
  - present or absent
- Files can also be: ensure =>
  - File, directory or link

```
[root@CentosVM ~]# vi rxmconf.pp
[root@CentosVM ~]# cat rxmconf.pp
#Puppet line comments start with #
/*Puppet block comments are enclosed by slash star and */
file {'/tmp/rxmdata.conf.d':
 ensure => directory,
 mode => 0644,
}

file {'/tmp/rxmdata.conf.d/rxm.thrift':
 ensure => file,
 content => "namespace * rxm
 service status { string get_status(); }",
}

user { 'fred':
 ensure => 'present',
 gid => '500',
 home => '/home/fred',
 password => '',
 password_max_age => '99999',
 password_min_age => '0',
 shell => '/bin/bash',
 uid => '508',
}

notify {"Super Custom Config-o-rama.";}
[root@CentosVM ~]# puppet apply rxmconf.pp
Notice: /Stage[main]//File[/tmp/rxmdata.conf.d]/ensure:
created
Notice:
/Stage[main]//File[/tmp/rxmdata.conf.d/rxm.thrift]/ensure:
defined content as '{md5}7a05e79a512c648ef4bafd2c618faa0b'
Notice: /User[fred]/ensure: created
Notice: Super Custom Config-o-rama.
Notice: /Stage[main]//Notify[Super Custom Config-o-
rama.]/message: defined 'message' as 'Super Custom Config-
o-rama.'
Notice: Finished catalog run in 0.15 seconds
[root@CentosVM ~]#
```

# Puppet 4 Language Specification

71

- The Puppet 4 language is the first puppet language dialect to have a formal specification
- While specifically for Puppet 4, Puppet 3 users will find this helpful as well
  - Preparing Puppet 3 code for use with 4
  - Writing better Puppet 3 code
  - Relatively small number of inconsistencies with later version of Puppet 3

The screenshot shows the GitHub repository page for `puppetlabs/puppet-specifications`. The repository is described as "Specification of the Puppet Language, Catalog, Extension points". It has 187 commits, 1 branch, 1 release, and 18 contributors. The current branch is `master`. A merge pull request #49 from `kylog/maint-drop-cfactor-refs` is highlighted, authored by `joshcooper` 4 days ago. The commit hash is `eFa20b0ef3`. The commit message is "Merge pull request #49 from kylog/maint-drop-cfactor-refs". The commit includes changes to `language`, `models`, `.gitignore`, `README.md`, and `file_paths.md`. The `README.md` file is shown below the commit list, with the title "Puppet Specifications" and the text "This repository contains specifications for the project Puppet, and related technologies." The `Puppet Language Specification` section is also visible, stating "The Puppet Programming Language Specification is a specification of the Puppet Programming Language. The first published release of this specification specifies the language version 4."

# Summary

- Puppet is a systems configuration tool which uses a declarative approach to system specification
- Puppet provides a layer of abstraction over the underlying system by using system specific providers
- Puppet is idempotent, manifests can be applied repeatedly, only changing system features which are incongruent with the manifest
- Puppet manages resources, the most important of which are
  - Packages, Files & Services
- The Puppet language supports variables, conditional statements and many other features allowing sophisticated configuration tasks to be accomplished
- Puppet uses Facter to create a comprehensive set of system variables available within Puppet manifests



# Lab 2

- Working with Puppet Manifests