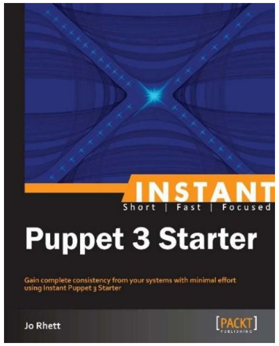
Puppet 101

Marc Skinner

Q2:2014

What is Puppet

Configuration Management at Scale
Managing Infrastructure with Puppet
O'REILLY
James Loope


INSTANT
Short | Fast | Focused
Puppet 3 Starter
Gain complete consistency from your systems with minimal effort using Instant Puppet 3 Starter
Jo Rhett
[PACKT] PUBLISHING

- Wikipedia
  - *Puppet is an open source **configuration management** tool from Puppet Labs, founded by Luke Kanies in 2005. It is **written in Ruby** and released as free software under the GPL until version 2.7.0 and the **Apache 2.0** license after that*
- *Managing Infrastructure With Puppet*
  - *Puppet is a **configuration management** framework with an object-oriented twist. If provides a **declarative language syntax** and an abstraction layer that allow you to write **heavily reusable** and **understandable configuration definitions**.*
- *Puppet 3 Starter*
  - ***Do more with less, ensure each thing is exactly the same, butler & a maid, deterministic NOT procedural***

Puppet Vocabulary

- **Resources**
  - A user account
  - A specific file
  - A directory of files
  - A software package
  - A running service
  - A scheduled cron job
  - An invocation of a shell command, when certain conditions are met

Resource Types

- File : manipulate a file, change uid/gid, permissions
- Package : install/uninstall
- Service : start/stop/enable/disable
- Notify : send agent run-time log
- Exec : run arbitrary command
- Cron : manage cron and cron jobs
- User : manage user accounts
- Group : manage groups

Manifest

- Action to run on a resource, using Puppet DSL (Domain Specific Language) stored in pp file

```
cat /root/mskinner-remove.pp

User { 'mskinner' :

   Ensure => absent,

}
```

Variables, Conditionals and Facts

- Variables
  - $my_variable = “This RHUG is sweet!”
  - Notify { $my_variable: }
- Conditionals
  - Logic
    - If, elseif, else
    - Case statement
- Facts
  - Built in, pre-assigned variables
    - $fqdn, $hostname, $kernel

Modules and Classes

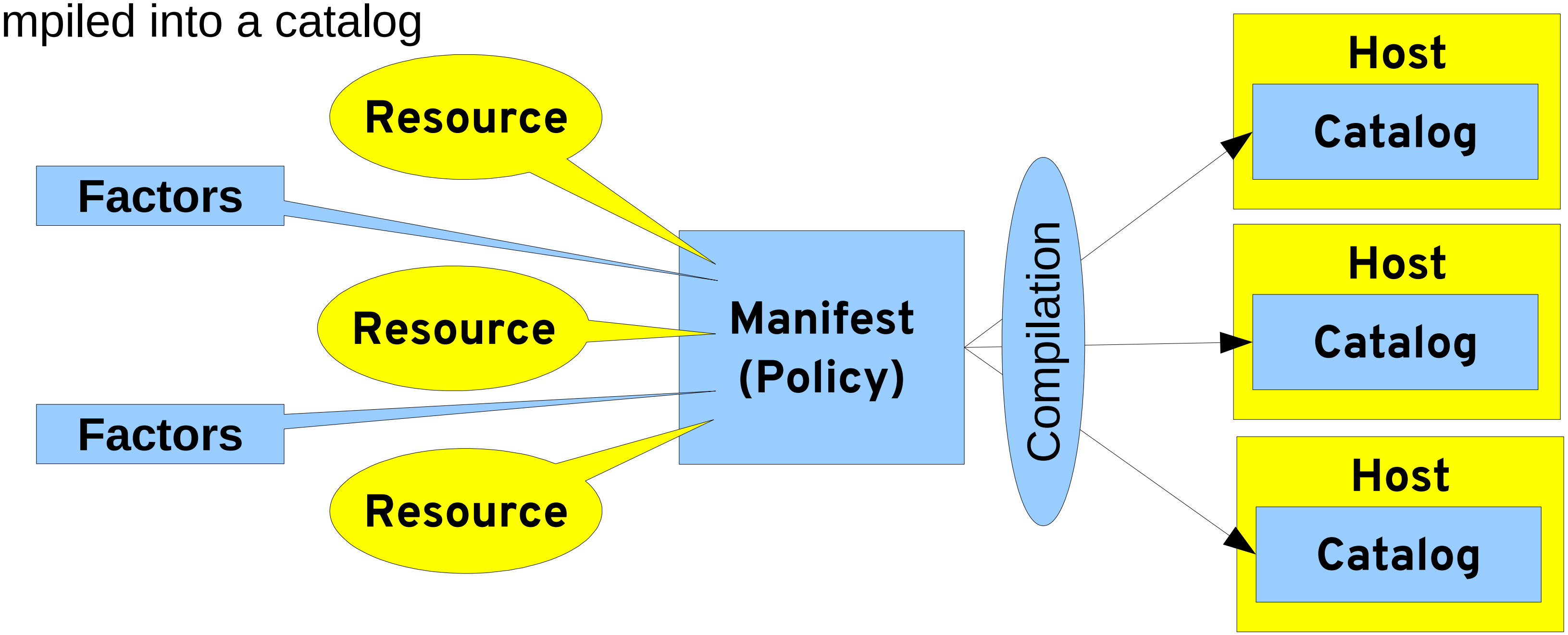
- Classes : like functions, named blocks of code

```
class my_class {
  … puppet stuff ...
}
include my_class
```

- Modules : are classes, logically save in individual pp files.
  - ntp.pp, mskinner-remove.pp

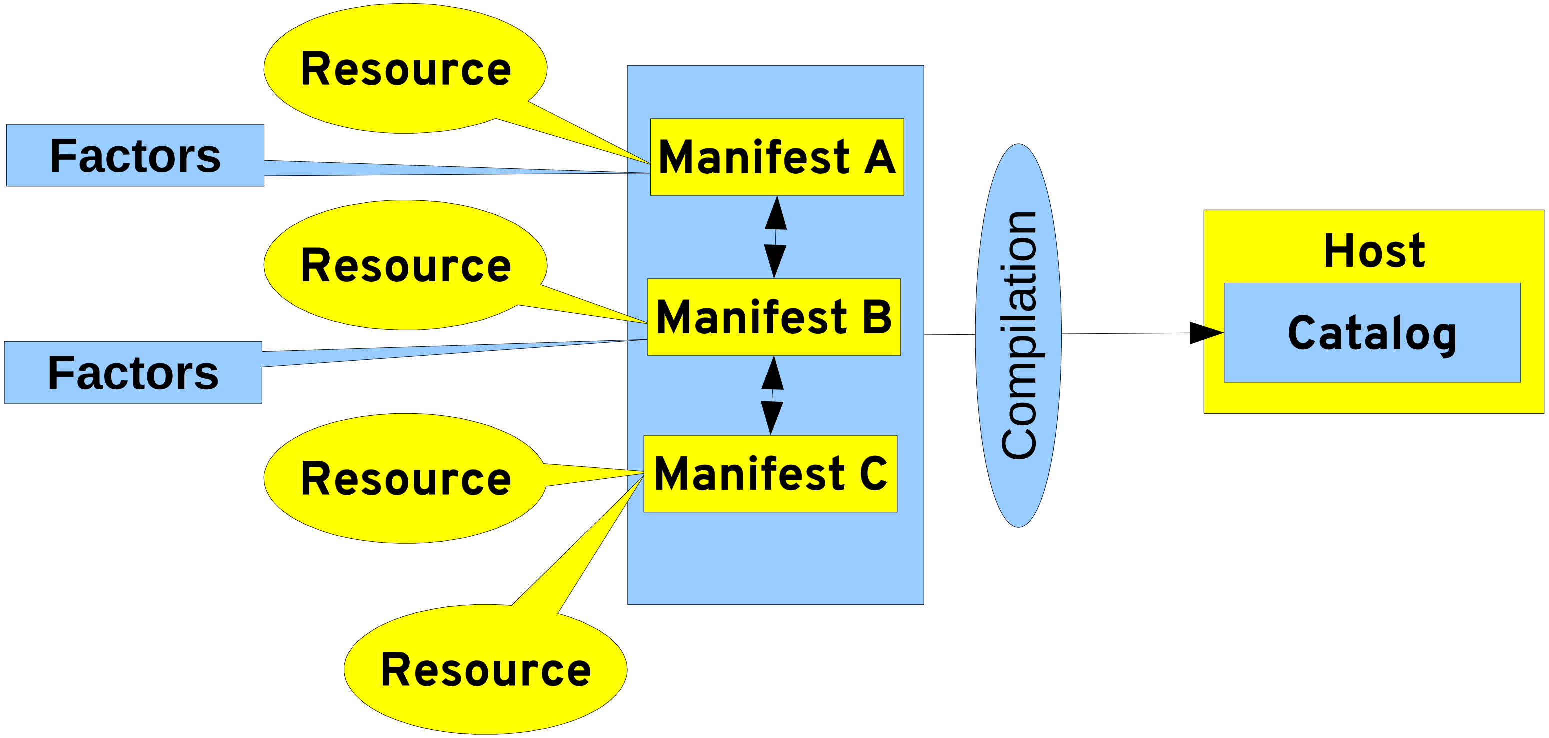
Puppet Crash Course

- Manifest – configuration file (.pp) described by a domain-specific language (DSL)
  - uses conditional logic to describe many node configurations at once
  - ideally arranged into modules
- Module - self-contained bundle of code & data.
  - pre-built modules available from Puppet Labs (Puppet Forge)
  - compiled into a catalog



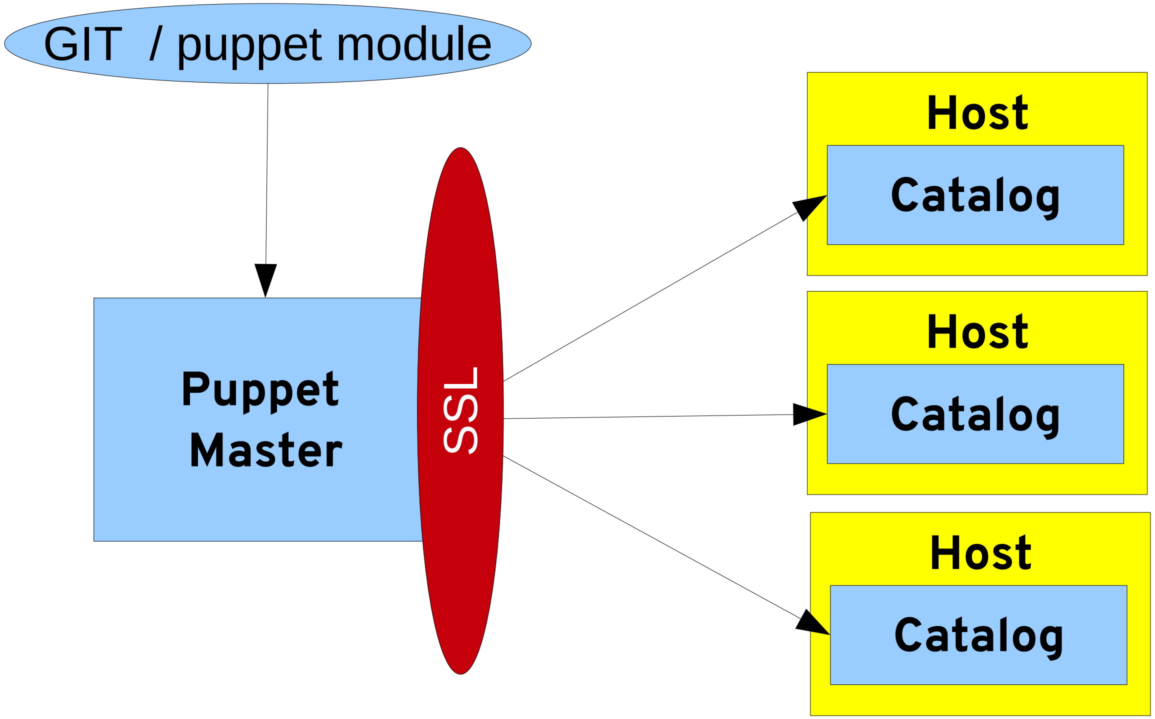
Puppet Crash Course

- Manifests can define dependencies on other manifests
- Does order matter?



Puppet Architecture

- Client agent pulls on schedule



Why Puppet?

- Declarative vs Procedural language
  - Procedural : require more coding
  - Declarative : declare action, think resources

- Pull vs Push models
  - Re-using something like SSH sounds great...
    - But what if SSH breaks?
    - Using a second route means Puppet can fix SSH
      - and you can use SSH to fix Puppet

Working with localhost

Facts: because not everything is known ahead of time:

- operating system
- hardware vendor, model, serial, virt-what
- network data, IPs, MACs
- memory, disk sizes
- software versions

And it's extensible (https://url.corp.redhat.com/18adf9c)

```
# /etc/facter/facts.d/foo.sh
key=value
# facter key
value
```

Fundamental building blocks

Resources are the basic abstractions in Puppet, representing:

- packages
- services
- files
- users, groups

They map directly to objects on the host being managed.

Each has a set of properties and parameters:

```
service { 'iptables':
  ensure => 'running',
  enable => 'true',
}

file { '/etc/passwd':
  ensure => 'present',
  mode   => '644',
  owner  => root,
  type   => 'file',
}
```

Evaluating and executing resources

Each resource in the catalog for a host is individually evaluated in order, then each properties is evaluated.

```
# ll /etc/passwd
-rw-------. 1 root root 2025 Jun  7 19:09 /etc/passwd
# cat passwd.pp
file { '/etc/passwd':
  mode => 0644,
}
# puppet apply passwd.pp
Notice: /File[/etc/passwd]/mode: mode changed '0600' to '0644'
Notice: Finished catalog run in 0.13 seconds
# ll /etc/passwd
-rw-r--r--. 1 root root 2025 Jun  7 19:09 /etc/passwd
```

Handling multiple platforms

Providers are the implementations associated with each type. First it identifies a provider suitable for the platform, then it provides a mechanism to change each property.

```
# puppet resource package vim-enhanced ensure=installed --debug
debug: Puppet::Type::Package::ProviderRpm: Executing '/usr/bin/rpm --version'
debug: Puppet::Type::Package::ProviderSun: file /usr/bin/pkginfo does not exist
debug: Puppet::Type::Package::ProviderApt: file /usr/bin/apt-get does not exist
debug: Package[vim-enhanced](provider=yum): Ensuring => present
debug: Puppet::Type::Package::ProviderYum: Executing '/usr/bin/yum -d 0 -e 0 -y
   install vim-enhanced'
notice: /Package[vim-enhanced]/ensure: created
package { 'vim-enhanced':
  ensure => '7.3.944-1.fc18',
}
```

Multiple hosts - add a Puppetmaster

Agent/Master Communications

- Security
  - Local manifests can be compromised
  - SSL is used to secure communications
    - Simple SSL setup
- Offloading
  - Manifest compiled on puppetmaster
  - Modules only need to exist on the puppetmaster
- Reporting
  - Master collects inventory and results
- External systems
  - Drive puppet from business logic/data

More on Modules

- Modules: the core of Puppet - reusable, sharable blocks
- Lots of good modules published
- `puppet module` to search/install (2.7.14 or higher)

```
# puppet module search foreman-puppet
Notice: Searching https://forge.puppetlabs.com ...
NAME               DESCRIPTION                              AUTHOR        KEYWORDS

theforeman-puppet  Puppet agent and server configuration    @theforeman   foreman puppet

# puppet module install theforeman-puppet
Notice: Preparing to install into /etc/puppet/environments/production/modules ...
Notice: Created target directory /etc/puppet/environments/production/modules
Notice: Downloading from https://forge.puppetlabs.com ...
Notice: Installing -- do not interrupt ...
/etc/puppet/environments/production/modules
└─┬ theforeman-puppet (v1.2.0)
  ├── theforeman-apache (v1.2.0)
  ├── theforeman-concat_native (v1.2.0)
  ├── theforeman-git (v1.2.0)
  └── theforeman-passenger (v1.2.0)

# ls /etc/puppet/environments/production/modules
apache   concat_native  git  passenger  puppet
```

Writing Modules - Structure

- <modulename> is the TLD
- init.pp must exist and the class has the same name as the module
- Each other class lives in a .pp file matching its class name
- puppet module generate <modulename>

```
$ head -n1 foreman/manifests/init.pp
class foreman {
$ head -n1 foreman/manifests/install.pp
class foreman::install {
$ head -n1 foreman/manifests/install/repos.pp
class foreman::install::repos {
```

```
foreman
├── lib
│   └── puppet
│       ├── parser
│       │   └── functions
│       │       └── foreman.rb
│       │
├── manifests
│   ├── config
│   │   ├── enc.pp
│   ├── config.pp
│   ├── database.pp
│   ├── init.pp
│   ├── install
│   │   └── repos.pp
│   ├── install.pp
│   ├── params.pp
│   ├── puppetmaster.pp
│   └── service.pp
└── templates
    ├── database.yml.erb
    ├── external_node.rb.erb
    ├── foreman-apache.conf.erb
    ├── foreman.default.erb
    ├── foreman-report.rb.erb
    ├── foreman.sysconfig.erb
    ├── foreman-vhost.conf.erb
    └── settings.yaml.erb
```

Publishing Modules

- Modulefile
- `puppet module build`
- http://forge.puppetlabs.com
- Upload to puppet forge

```
name    'theforeman-foreman'
   version '1.2.0'
   source 'git://github.com/theforeman/puppet-foreman'
   author 'theforeman'
   license 'GPLv3+'
   summary 'Foreman server configuration'
   description 'Module for configuring Foreman'
   project_page 'http://github.com/theforeman/foreman-installer'

   dependency 'theforeman/apache', '>= 1.2.0'
   dependency 'theforeman/concat_native', '>= 1.2.0'
   dependency 'theforeman/passenger', '>= 1.2.0'
```

External Node Classifiers

- A way for Puppet to ask for external data **NOT** Facts
- Just an executable
  - As simple as "echo"
  - As complex as LDAP queries or HTTP GET
- Requirements
  - Must output correct format on STDOUT
  - Must exit with status 0

Writing Modules - Class Params

- Namespaced input for modules
- Can be thought of as an API
- Supplied via an ENC
- Can have default values

```
---
  classes:
    ntp:
      server: "ntp.org"
  parameters:
    owner_name: "Greg"
```

```
class ntp (
  $server = ‘0.pool.ntp.org’
) {
  notify { 'ntp':
    message => "server=[$server]"
  }
}
```

Useful Flags to Puppet

- -vt (verbose test)
  - Run one execution of Puppet in the foreground
  - Good for testing manifests
- No-op mode (--noop)
  - Useful for testing runs to see if there's any unintentional side-effects of your manifests
  - Can be used as a form of compliance report
- Tags (--tag <class>)
  - Used to apply only part of a catalogue
  - Combine with no-op to test/apply a manifest in stages

Cool Stuff: Puppet Resource

- Generates Puppet snippets for the existing configuration of a system
- Useful for seeing what's currently defined
  - and how to define it

```
# puppet resource file /etc/hosts

file { '/etc/hosts':
  ensure  => 'file',
  content => '{md5}85140..',
  ctime   => '2013-06-10..',
  group   => '0',
  mode    => '644',
  mtime   => '2013-06-10..',
  owner   => '0',
  type    => 'file',
}
```

```
# puppet resource service sshd

service { 'sshd':
  ensure => 'running',
}
```

Questions?