Puppet
configuration management and changes control system

Chris Kruk, STFC-RAL

2nd July 2009

Topics:

- Why do we need configuration management system?
- What is Puppet?
- How Puppet works.
- Puppet components.
- Getting started with Puppet-introduction
- Question

Science & Technology Facilities Council

Why do we need configuration management?

- Large number of servers ~350, and growing
- Servers with different architecture, configuration, types etc.
- Ability to replace a server which dies or add another server

Science & Technology Facilities Council

Why do we need configuration management?

- Ability to provide the same configuration across all servers
- Better way to push out changes
- Stop duplicating effort

What is Puppet?

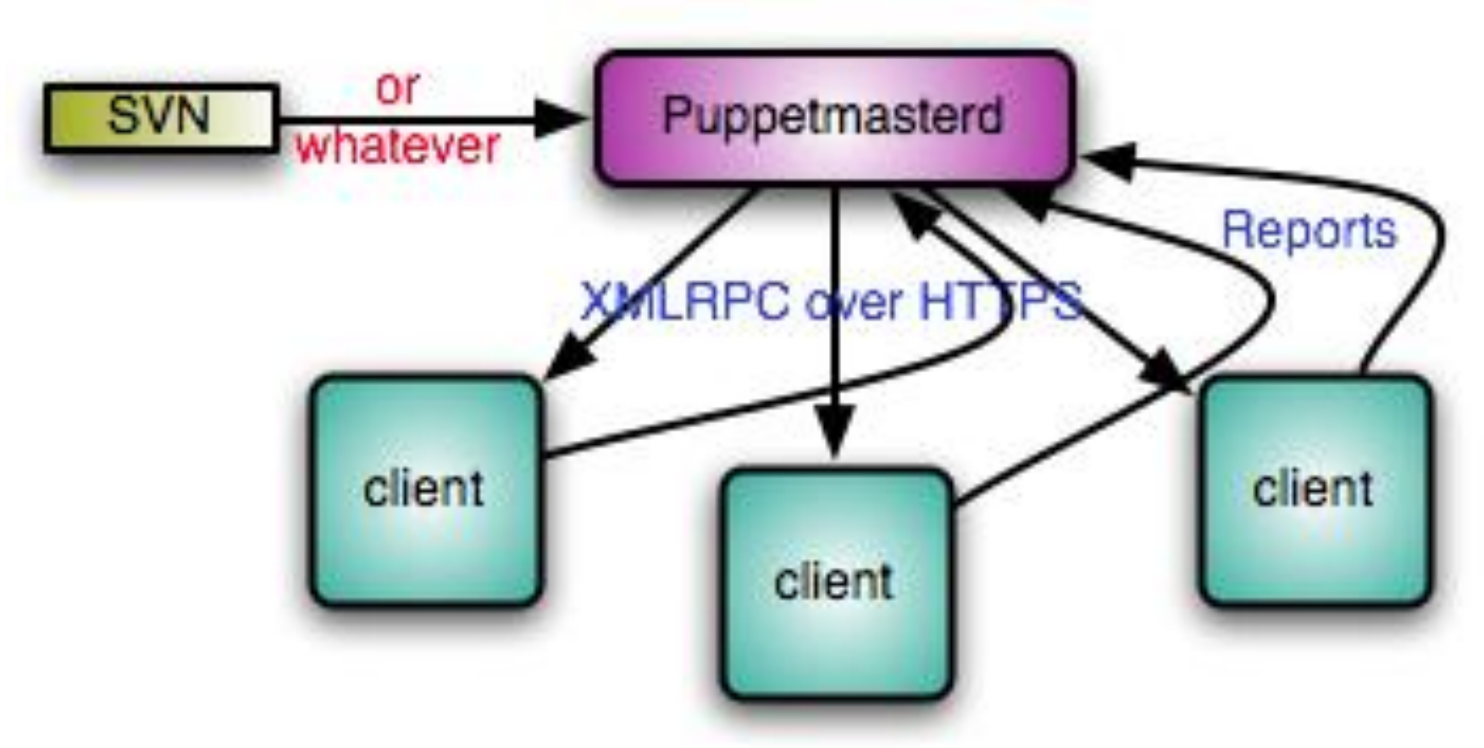
- Puppet is:
  - a declarative language for expressing system configuration
  - a client and server for distributing it
  - a library for realizing the configuration
  - an abstraction layer between the system administration and the operating system
  - written in Ruby and distributed under the GPL

How Puppet works:

- Client (agent) – puppetd
- Server – puppetmasterd

- Client connects to the server periodically
- Server provides the configuration based what has been written by sysAdmin
- Client configuration can be stored in LDAP, DB or puppet files

Puppet components:



Why puppet:

- Puppet supports variety of the operating systems (Linux {RH, SL, Debian ...}, OS X, Solaris ...)
- Open Source
- Puppet has a large and active user community and is being actively developed and supported
- Puppet uses SSL for secure communication between client and server

What is Facter?

- Facter gathers information about the client, which can be used as variables within puppet.
- You can add custom facts as needed.

```
 case $operatingsystem {
  freebsd: {
    $rootgroup = "wheel"
  }
  default: {
    $rootgroup = "root"
  }
}
```

Example Facts

```
architecture => i386
fqdn => debiantest.example.com
ipaddress => 192.168.1.41
macaddress => 00:0C:29:DC:FC:D9
kernelrelease => 2.6.18-4-686
memoryfree => 214.37 MB
memorysize => 250.95 MB
operatingsystem => Debian
processorcount => 1
processor0 => Intel(R) Xeon(TM) CPU 3.06GHz
processorcount => 1
puppetversion => 0.23.2
rubyversion => 1.8.5
swapfree => 235.29 MB
swapsize => 235.29 MB
```

What Can Puppet Do?

- Puppet manages objects on the client, called resources.
- Resources are come in different types. Some of the types are:
  - Cron
  - file
  - group
  - Host
  - user

Science & Technology Facilities Council

Getting Started

- Install puppet (and facter and ruby).
- Set up the server (puppetmasterd).
- Write a manifest for your node and upload it to the server.
- Start puppetmasterd on the server.
- Run puppetd on the node.

Science & Technology Facilities Council

What is a Manifest?

- Manifest are recipes that Puppet uses to build the client configuration.
- Let's create a simple manifest:

```
class bacula-client {}

node debiantest {
     include bacula-client
}
```

- This creates a class called bacula-client, and node (puppet client) called debiantest which includes that class. Of course, it doesn't do anything yet.

Useful Class

```
class bacula-client {
  package { "bacula-client": ensure => present; }

  service { "bacula-fd":
    ensure  => running,
    enable  => true,
 }

  file {"/etc/bacula/bacula-fd.conf":
    mode    => "644",
    owner   => "root",
    group   => "root",
    ensure  => present,
    content => template("bacula/bacula-fd.conf"),
  }
}
```

Dependencies

- Dependencies allow you specify the order objects will be applied. For example, you don't want to try to put a file in /etc/apache2 if apache2 is not installed yet.

```
class apache2 {
   define simple-vhost ( $admin = "webmaster",
  $aliases, $docroot) {
     file { "/etc/apache2/sites-available/$name":
       mode    => "644",
       require => [ Package["apache2"],
  Service["apache2"] ],
       content => template("apache/vhost.conf"),
     }
  }
}
```

Science & Technology Facilities Council

Inheritance

- Puppet types can inherit from similar types and override if needed:

```
class bacula-special inherits bacula-client {
  File[“/etc/bacula/bacula-fd.conf”] {
    content => template("bacula/bacula-fd.conf-
   special"),
   }
}
```

Modules

- Modules allow you to group both the logic and the files for an application together.
- Modules can contain four types of files:
  - Manifests - must be stored in manifests/
  - Templates - must be stored in templates/, and the module name must be added to the template name
  - Files - stored in files/, these are available from the file server under modules/<module name>/files/<file name>
  - Plugins – additional types, providers or facts.

Science & Technology Facilities Council

Organisations using Puppet:

- PSCSG/Castor Team – STFC-RAL
- Google
- The Ohio State University - Department of Mathematics
- Fedora Project
- Stanford University
- University of California
- Red Hat

Science & Technology
Facilities Council

References:

- http://reductivelabs.com
- My presentation mainly based on Martha's Greenberg talk from http://www.mit.edu/people/marthag/talks/puppet/

Science & Technology
Facilities Council

Questions?