Implementing Puppet for Configuration Management

Andrew Mallett
LINUX AUTHOR AND TRAINER

@theurbanpenguin www.theurbanpenguin.com

Objectives

Understanding Puppet

- PuppetLabs
  - Puppet Enterprise
  - Puppet CE
- Server / Client
  - Open TCP port 8140 on server
  - Client checks in every 20 mins
  - For local use only: puppet apply
- Ruby based
  - Manifests
  - Uses the .pp suffix

Puppet Repos

RedHat based systems do not ship with puppet but the agent is available from the EPEL repo.

Ubuntu 20.04 ships with version 5 of Puppet and we install the agent only on this system

```
$ sudo apt update
$ apt search '^puppet$'
$ sudo apt install -y puppet
$ puppet --version
$ puppet help
```

Installing Puppet

To maintain a simple overview of Puppet, we shall install just the Puppet Client, know as the Agent. We can apply local configuration using the agent.

Demo

Working on the Ubuntu System:

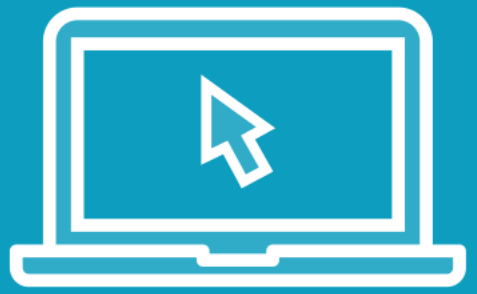
- Install puppet
- Check version
- Obtain CLI help
  - puppet help
  - puppet describe <resource>
  - puppet resource package tree

```
$ mkdir ~/puppet ; cd ~/puppet

$ sudo puppet apply -e 'package { "chrony": ensure => installed }'

$ sudo puppet apply -e 'service { "chrony": ensure => running , enable => true }'

$ sudo puppet apply -e 'host { "redhat": ip => "192.168.56.11" }'

$ puppet resource host
```

Testing Puppet Using Ad-Hoc Commands

Puppet can apply configurations directly from the CLI. We will need to elevate privileges if required by the underlying operation. Note keyword differences in the language compared to Ansible.

Demo



Applying Local Puppet Configuration

- Install software
- Manage service
- Add host entries

Modules and Manifests

- The basis of persistent puppet configuration is the manifest
- More complex configuration can be stored on a module
- Public modules are available from Puppet Forge

```
$ puppet module list
$ sudo puppet module install puppetlabs/apache -i /usr/share/puppet/modules
$ sudo puppet apply -e "include apache"
```

Install Puppet Module

Predefined code can be downloaded as module from forge.puppet.com

Demo

Working with Modules:

- List modules
- Install module
- Include module

```
$ cd ~/ansible

$ vim message.pp
# vim: set ft=ruby ts=2 sw=2 et ai :
notify {'Hello World':
  message => "Hello World!",
}

$ puppet parser validate message.pp

$ puppet apply --noop message.pp

$ puppet apply message.pp
```

Creating Manifests

Manifests make up the basis of Puppet configuration. They are a form of Ruby file, so we may choose to add the modeline.

```
$ puppet resource user vagrant > ~/puppet/user.pp
```

Copy a Resource

We can copy an existing resource to create a new manifest and edit it is required.

noroot.pp

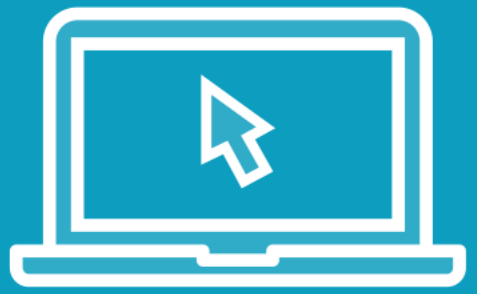
```
# vim: set ft=ruby ts=2 sw=2 et ai :
service { 'sshd':
  ensure => 'running',
  enable => true,
 }
file_line { 'root_ssh':
  path => '/etc/ssh/sshd_config',
  ensure => 'present',
  line => 'PermitRootLogin no',
  match => '^PermitRootLogin',
  notify => Service['sshd'],
 }
```

Demo

Working with Manifests:

- Create a simple manifest
- Create manifest from existing resource

Demo



Restarting Services:

- Edit SSHD configuration

Summary

Understanding Puppet from PuppetLabs

- Client / Server model
- Inbound TCP 8140 on server
- Local only with puppet apply
- Manifests are the basis of configuration
- Modules extend code and can be downloaded from Puppet Forge
- Ruby files with .pp suffix

Implementing SaltStack for Configuration Management