SCM: Puppet



(Audience Handbook)

Index

* [What is Puppet?](#_6ffr059jszu8)
* [Why Use A Configuration Management Tool?](#_rguxyycnh56d)
* [Install standalone puppet](#_lu9mhenh0gjv)
  + [On Centos](#_lqsypoxtfm6f)
  + [On Ubuntu](#_e10olfxmh0h8)
* [Puppet Resources](#_muhpyfg097xy)
  + [User](#_qeul8546b5mx)
  + [Group](#_o1ns4sk0vprk)
  + [Package](#_ukx75ln16gxq)
  + [File](#_qscmqj1cip5r)
  + [Host](#_wyqpxstf6rou)
* [Assessment](#_bjtjgwscfzzk)

## 

## What is Puppet?

Puppet is an open source configuration management software tool that allows organizations to control the exact configuration of as many as tens of thousands of nodes from a single central server. Puppet is mature and popular, and is in use by large organisations around the world. However, installing Puppet under this distributed client/server model can be complicated, requiring the setup of a central Puppet server, and its relationship to client nodes.

Puppet is not just a glorified installation script manager. When Puppet runs, it inspects the configuration of the node, identifies any differences between the configuration of the node and the configuration specified in the Puppet configuration file, and then makes the changes necessary to bring the node into the specified state. This means that it can be used both to configure nodes and repair them.

Puppet has such a strong image as a mass cloud configuration tool, that it might surprise you to know that Puppet can also be run in "standalone" mode in which it is manually run to configure a single node. In standalone mode, Puppet is an excellent tool for configuring individual nodes.

## Why Use A Configuration Management Tool?

We can identify three levels of sophistication in installing software: manual, scripted, and configuration management based.

* In a manual installation, you issue a series of commands to the command shell to install the software.
* In a scripted installation, you create a script (e.g. a Ruby script or BASH script) to install the software and execute it.
* In a configuration management based installation, you create a configuration management tool specification of the desired state of the node, and the configuration tool compares the node's state with the desired state, and drives the node into the desired state.

Manual installation should be avoided because it is a pre-automation solution that embeds all the installation knowledge in the head of one or more engineers rather than into a file (unless the engineers have written down the installation procedure).  
  
Scripted installation is far better than manual installation, but suffers from the problem that if you perform an installation and then damage some small part of it, you cannot use the script to repair the damage; you have to start again from scratch and reinstall everything.  
  
Configuration management based installation is the best solution. A configuration management tool will automate installation (as will an installation script), but it can also be used to repair the software if it gets damaged. It can also be used to change the desired configuration, and drive the node to the new desired state.

## Install standalone puppet

### On Centos

$ sudo rpm -ivh <https://yum.puppetlabs.com/puppetlabs-release-pc1-el-6.noarch.rpm>

$ sudo yum -y install puppet-agent

### On Ubuntu

$ wget <https://apt.puppetlabs.com/puppetlabs-release-pc1-trusty.deb>

$ sudo dpkg -i puppetlabs-release-pc1-trusty.deb

$ sudo apt-get update

$ sudo apt-get install puppet-agent -y   
  
***Note: logout and login then run (puppet -V) to check whether puppet install or not***

## Puppet Resources

### User

To create and manage a user.

* + To create a user   
    $ puppet apply -e "user { 'xebia': ensure => present }"
  + To update the home directory of the xebia user  
    $ puppet apply -e "user { 'xebia': ensure => present, home => '/var/lib' }"

### Group

To create and manage a group.

* + To create a group   
    $ puppet apply -e "group { 'xebia1': ensure => present }"

### Package

To install and manage a package.

* + To install openssh-server package.

$ puppet apply -e "package { 'openssh-server': ensure => present }"

### File

Create and manage a file.

* + To create a file in /tmp dir with some content.

$ puppet apply -e "file { 'message': ensure => present, path=> '/tmp/xebia', content => 'Hello Sandy'}"

### Host

Create and manage a /etc/hosts file entry

* + To create an entry in /etc/hosts file.

$ puppet apply -e "host { 'www.xebia.com': ensure => present, name => 'www.xebia.com', ip => '192.168.0.1'}"

***Note: run following above command you have to root user***

## Assessment

1. Use cron and sshkey puppet resource.
2. Create a simple puppet module install nginx and setup two different v-host.