The Puppet Language

Lưu ý 1 số định nghĩa:

Manifests

- Trong puppet tất cả programs được viết bằng Ruby và được save dưới dạng .pp. Nó được gọi là Manifests

- Nó được xây dựng cho việc quản lý hoặc tạo mới bất kỳ máy chủ nào.

- Tất cả chương trình được puppet viết ra tuần theo Puppet coding style

- Core của puppet là các resource được định nghĩa hoặc các resource đại diện cho cách trạng thái. Trong bất kỳ manifests nào, user có thể có 1 tập các resource, được quản lý = class.

Manifest File Workflow:

- Files

- Resources

- Templates

- Nodes

- Classes

Writing Manifests

Làm việc với các biến

- Khi làm việc với các biến, user có thể định nghĩa các giá trị mới, Puppet hỗ trợ rất nhiều kiểu dữ liệu, thường sử dụng string hoặc hặc array string

VD:

$package = "vim"

package { $package:

ensure => "installed"

}

Sử dụng vòng lặp

- Sử dụng cho việc lặp đi lặp lại 1 quá trình:

VD:

$packages = ['vim', 'git', 'curl']

package { $packages:

ensure => "installed"

}

Sử dụng biểu thức điều kiện

- Puppet hỗ trợ hầu như các biểu thúc so sánh truyền thống

if $OperatingSystem != 'Linux' {

warning('This manifest is not supported on this other OS apart from linux.')

} else {

notify { 'the OS is Linux. We are good to go!': } }

Puppet – Module

- Sử dụng để định nghĩa 1 tập các resource, class, file, template.

VD:

class autofs {

package { autofs: ensure => latest }

service { autofs: ensure => running }

file { "/etc/auto.homes":

source => "puppet://$servername/modules/autofs/auto.homes"

}

file { "/etc/auto.master":

content => template("autofs/auto.master.erb")

}

}

Puppet - Facter & Facts

- Puppet hỗ trợ việc giữ nhiều giá trị như biến môi trường. Tính năng này được hỗ trợ bởi facter

- Cơ chế hoạt động giống như /bash /sh trong linux

- cmd facter sử dụng để liệt kết tất cả biến giá trị hiện tại

# facter

Example to Test

[root@puppetmaster modules]# tree brcle\_account

brcle\_account

└── manifests └── init.pp [root@puppetmaster modules]# cat brcle\_account/manifests/init.pp

class brcle\_account {

user { 'G01063908':

ensure => 'present',

uid => '121',

shell => '/bin/bash',

home => '/home/G01063908',

}

file {'/tmp/userfile.txt':

ensure => file,

content => "the value for the 'OperatingSystem' fact is: $OperatingSystem \n",

}

}

Puppet – Resource

- Resources là 1 trong những đơn vị cơ sở của Puppet. Nó được sử dụng cho viết thiết kế, xây dựng cho các cơ sở hạ tầng.

- Puppet có nhiều loại resource, có thể sử dụng để định nghĩa kiến trúc hệ thống hoặc sự dụng cho việc xây dựng hệ thống phân tầng

- 1 khối trong puppet code trong manifest file hoặc bất cứ loại file nào đều được gọi lại resource declaration

- Block code được viết bởi được gọi là DML.

user { 'vipin':

ensure => present,

uid => '552',

shell => '/bin/bash',

home => '/home/vipin',

}

- Việc định nghĩa bất kỳ loại resource được có thể làm được = code block. Bao gồm

+ Resource Type ( Tại đoạn code trên là user)

+ Resource Parameter ( là vipin tại đoạn code trên )

+ Attributes ( là ensure, uid, shell, home )

+ Values ( là các giá trị (pre, 552, /bin/bash, /home/vipin)

Resource Type

- Để hiện thị ra các loại resource type có sẵn, sử dụng cmd “puppet describe --list”

Resource Title

- sử dụng cmd để list ra các resource có sẵn sử dụng cmd: “puppet resource user”

- Tím chính xác 1 resource: “puppet resource user tomcat”

Attributes & Values

- 1 resource được tại từ 1 tập cac attr, để xem chi tiết các attr trong resource sử dụng

#puppet describe user

Puppet - Resource Abstraction Layer

- Trong puppet, Resource Abstraction Layer (RAL) có thể coi là mô hình khái niệm cốt lõi trên toàn bộ hạ tầng cơ sở và là cách settup Puppet.

- Mỗi ký tự trong mô hình RAL đề có ý nghĩa riêng

Resource [R]

- Đại diện cho tất cả các resource sẽ sử dụng cho bất cứ cấu hình nào

- Nó đơn giản là các resource mặc định có trong puppet

- Có thể xem chúng là các tài nguyên được định nghĩa trước

- Có thể hiếu chúng là các đối tượng với các thể hiện

Abstraction [A]

- Sự trừu tượng có thể coi là tính năng quan trọng, nơi các đối tượng được định nghĩa độc lập với các hệ điều hành

- Nói cách khác, khi viết manifest file, user có thể thoải mái sử dụng các resouce mà ko cần quan tâm đến đối tượng target hoặc hệ điều hành.

- Puppet sẽ giải quyết tất cả vấn đề bên dưới (sử dụng nguồn, OS, ..) mà user sẽ không cần phải quan tâm nhưng gì nó đang thực hiện.

Layer [L]

- Nó có thể coi là 1 định nghĩa cho tất cả các machine setup, các config về tập các resources

VD: puppet describe user --providers