Puppet 集中配置管理系统

Centralized configuration management system

守住每一天
http://bbs.linuxtone.org

Puppet 安装配置 v1.0

目 录

一、关于 Puppet

1.1 什么是 Puppet?

puppet 是一种 Linux、Unix 平台的集中配置管理系统，使用自有的 puppet 描述语言，可管理配置文件、用户、cron 任务、软件包、系统服务等。puppet 把这些系统实体称之为资源，puppet 的设计目标是简化对这些资源的管理以及妥善处理资源间的依赖关系。

puppet 采用 C/S 星状的结构，所有的客户端和一个或几个服务器交互。每个客户端周期的（默认半个小时）向服务器发送请求，获得其最新的配置信息，保证和该配置信息同步。每个 puppet 客户端每半小时(可以设置 runinterval=30)连接一次服务器端,下载最新的配置文件,并且严格按照配置文件来配置服务器. 配置完成以后,puppet 客户端可以反馈给服务器端一个消息. 如果出错,也会给服务器端反馈一个消息.

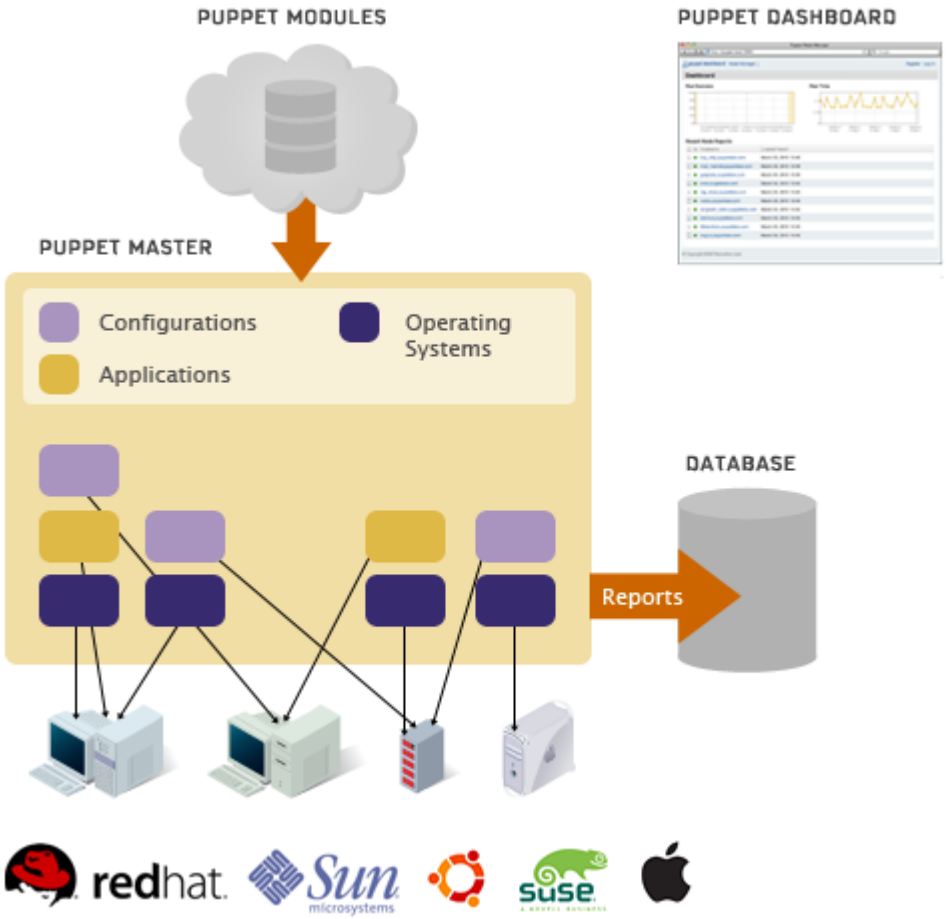
1.2 为什么要使用 puppet ？

当你去管理 10 台服务器，你肯定会说小意思。没有任何压力。

当你去管理 100 台服务器，你肯定也会说小意思。

当你去管理 1000+台服务器呢？你是不是就头痛了，不同的机器，不同的系统，使用不同的软件版本，配置也不一样。这样为了提升效率。Puppet 就派上了大用场。

1.3Puppet 架构



1.4 简单地说下工作原理：

Puppet 后台运行的时候默认是半小时执行一次，不是很方便修改。可以考虑不让它在后台跑而是使用 crontab 来调用。这样可以精确控制每台客户端的执行时间。分散执行时间也可以减轻压力

Puppet 的工作细节分成如下几个步骤：

1、 客户端 puppetd 调用 facter，facter 会探测出这台主机的一些变量如主机名、内存大小、IP 地址等。然后 puppetd 把这些信息发送到服务器端。

2、 服务器端的 puppetmaster 检测到客户端的主机名，然后会到 manifest 里面对应的 node 配置，然后对这段内容进行解析，facter 送过来的信息可以作为变量进行处理的，node 牵涉到的代码才解析，其它的代码不不解析，解析分几个过程：语法检查、然后会生成一个中间的伪代码，然后再把伪代码发给客户机。

3、 客户端接收到伪代码之后就会执行，客户端再把执行结果发送给服务器。

4、 服务器再把客户端的执行结果写入日志。

二、安装 Puppet

2.1 安装需求

注：本文都是在 Centos5 下进行安装与配置。其它系统请参考官网。

Ruby 1.8.2+ facter

其它的库:

base64

cgi

digest/md5

etc

fileutils

ipaddr

openssl

strscan

syslog

uri

webrick

webrick/https

xmlrpc

2.2 Puppet 版本

2.6.4

0.25.5

2.6.4 和 0.25.5 有功能和基本命令上有一些变化区别如下：

puppetmasterd → puppet master

puppetd → puppet agent

puppet → puppet apply

puppetca → puppet cert

ralsh → puppet resource

puppetrun → puppet kick

puppetqd → puppet queue

filebucket → puppet filebucket

puppetdoc → puppet doc
pi → puppet describe

通常我们使用 epel 安装的 puppet 都是 0.25.5 。在安装前做注意版本的区别。本文采用 2.6.4 版本，以下配置都是在 2.6.4 版本上完成。
其它系统相关的情况详见：（遗憾的是没有看到对 windows 的支持）
http://projects.puppetlabs.com/projects/puppet/wiki/Downloading_Puppet

2.3 源码包安装

源码包安装时，版本没有太多的区别。

2.3.1 安装 ruby 最好使用 RPM 包安装。可以采用 epel 然后 yum
http://mirrors.sohu.com/fedora-epel/5Server/ 下载相应版本的 epel
yum install ruby ruby-devel ruby-doc*

2.3.2 安装 facter
下载最新的版本:

```
$ wget http://puppetlabs.com/downloads/facter/facter-latest.tgz
$ gzip -d -c facter-latest.tgz | tar xf -
$ cd facter-*
$ sudo ruby install.rb #使用 root 账号执行
```

2.3.3

```
# 下载最新版
$ wget http://puppetlabs.com/downloads/puppet/puppet-latest.tgz
# untar and install it
$ gzip -d -c puppet-latest.tgz | tar xf -
$ cd puppet-*
$ sudo ruby install.rb # 使用 rot 安装
```

2.4 yum 安装

1. Epel 安装 0.25.5
2. 采用如下方式安装 2.6.4

2.4.1 配置 yum 源

```
cd /etc/yum.repos.d/
vim puppet.repo
[puppetlabs]
name=Puppet Labs Packages
baseurl=http://yum.puppetlabs.com/base/
enabled=0
gpgcheck=0</code></pre>

vim epel.repo
name=Extra Packages for Enterprise Linux 5 -$basearch
#baseurl=http://download.fedoraproject.org/pub/epel/5/$basearch
mirrorlist=http://mirrors.fedoraproject.org/mirrorlist?repo=epel-5&amp;arch
=$basearch
```

failovermethod=priority
enabled=0
gpgcheck=0

[epel-puppet]
name=epel puppet
baseurl=http://tmz.fedorapeople.org/repo/puppet/epel/5/$basearch/
enabled=0
gpgcheck=0

vim ruby.repo
[ruby]
name=ruby
baseurl=http://repo.premiumhelp.eu/ruby/
gpgcheck=0
enabled=0

2.4.2 升级 ruby

puppet 2.6 需要 ruby 1.8.6 去运行 puppet-dashboard
升级 ruby
yum --enablerepo="ruby" update ruby

如果你没有安装 ruby 请使用:
#yum -y install ruby

2.4.3 安装 puppet

安装 Puppet Server
Server 端：On your puppetmaster server:
yum --enablerepo=epel,epel-puppet install puppet-server>
Client 端：On your puppet client
yum --enablerepo="epel,epel-puppet" install puppet

2.5 gem 安装

$ wget http://puppetlabs.com/downloads/gems/facter-1.5.7.gem
$ sudo gem install facter-1.5.7.gem
$ wget http://puppetlabs.com/downloads/gems/puppet-0.25.1.gem
$ sudo gem install puppet-0.25.1.gem

三、配置 Puppet

3.1 配置服务器端

3.1.1 设置 hostname

#echo “puppetmaster.linuxtone.org” > /etc/hostname
#hostname –F /etc/hostname

3.1.2 配置 site.pp

vim /etc/puppet/site.pp
node default {

file { “/tmp/temp1.txt”: content => “hello,first puppet manifest”; }
}

3.2 配置客户端

3.2.1 设置 hostname

#echo “puppetclient.linuxtone.org” > /etc/hostname
#hostname –F /etc/hostname

3.1.2 指定 hosts 或使用 dns 解析

echo “192.168.1.100 puppetmaster.linuxtone.org” >> /etc/hosts

3.3 验证

3.3.1 客户端运行：

puppetd --server master.example.com（puppetmaster.linuxtone.org） --test

上面的命令让 puppetd 从 master.example.com（puppetmaster.linuxtone.org）去读取 puppet 配置文件. 第一次连接,双方会进行 ssl 证书的验证,这是一个新的客户端,在服务器端那里还没有被认证,因此需要在服务器端进行证书认证.
在服务器端的机器上执行下面的命令来认证客户端的证书

3.3.2 服务器端运行： puppetca -s client.example.com

3.3.3 客户端再次运行 puppetd –server puppetmaster.linuxtone.org --test

这样验证就算是做完了。客户端会在/tmp 目录生成内容为“hello,first puppet manifest”的 temp1.txt 文件。

3.4 自动验证

在/etc/puppet 创建 **autosign.conf**

内容 *.linuxtone.org

四、Puppet 结构

4.1 组织结构

为什么要说 puppet 的组织结构？当你安装完 puppet 后，你会发现你不知道它的目录结构是什么样的。要如何组织，怎么样才算合理？puppet 目录在/etc/puppet 下面。

树结构如下：

```
|-- puppet.conf      #主配置配置文件
|-- fileserver.conf  #文件服务器配置文件
|-- auth.conf        #认证配置文件
|-- autosign.conf    #自动验证配置文件
|-- tagmail.conf     #邮件配置文件（将错误信息发送）
|-- manifests        #文件存储目录(puppet 会先读取该目录的.PP 文件<site.pp>)
|    `--nodes
|    |     | puppetclient.pp
|    |-- site.pp       #定义 puppet 相关的变量和默认配置。
|    |-- modules.pp    #加载 class 类模块文件（include syslog）
|-- modules            #定义模块
|    `-- syslog        #以 syslog 为例
|          |-- file
|          |-- manifests
|          |     |-- init.pp
```

```
|          `-- templates        #模块配置目录
|              |-- syslog.erb   #erb 模板
```

我在 nodes 里定义了每个 cluster 的子目录。以便管理 nodes.pp；只需要在 site.pp 里添加：inclde nodes/cluster_name/*.pp 支持通配符

```
syslog modules 目录树
  syslog
  |-- file
  |-- manifests
        |     |-- init.pp     #class 类配置
        `-- templates
              |-- syslog.erb #erb 模板
```

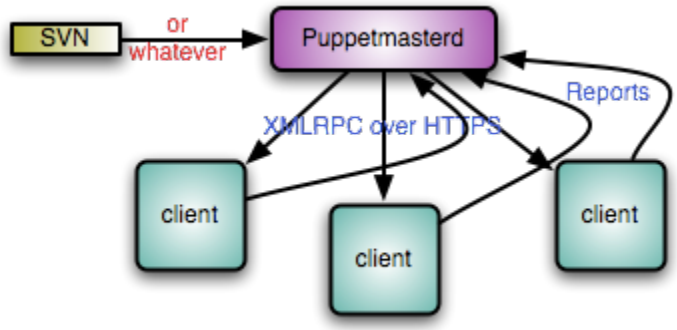
4.2 使用 svn or git

svn 和 git 原理都是一样的，都是先把 puppet 内容存在仓库里，然后提交更新至 puppet 目录；我采用的方式是：本机 svn 至 svn 服务器，puppet 服务器端采用 cron svn update 下来。每分钟执行一次。

每次提交更新前，先会使用其中的一台 client 进行测试，确保无误。避免大的故障。

下文只列出 svn 的配置方法（常用）。Git 与其一样。所以忽略。

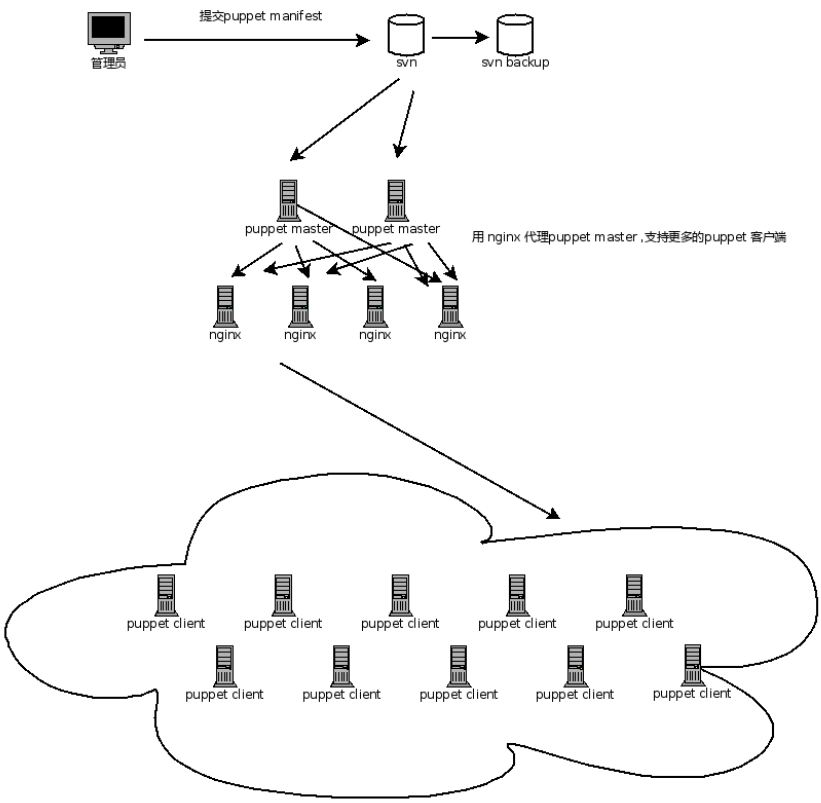
使用 svn 管理 puppet



我的做法更省事。就是使用 svn 创建一个仓库，然后把/etc/puppet 放到仓库里。

五、 Nginx or pound ?

先来看下工作原理图。



5.1 为什么要使用 nginx?

性能：nginx 非常的小并且速度快。它比 pound 更快捷

http://projects.puppetlabs.com/projects/1/wiki/Using_Mongrel_Pound

日志调试：优于 pound，更加简洁

灵活性：nginx 处理应用层的 ssl 客户端验证，而不是终止 ssl 连接

如果不使用 pound 你不需要打 mongrel 补丁

安装很简单，配置语法也很直观

缺点：配置后不能删除 ssl 证书，解决方法是代理 pound 或 apache ssl 代理。也可以删除客户端的 ssl 目录来解决

5.2 安装 rubygem-mongrel

如果你有 epel 源，可以直接 yum –y install rubygem-mongrel

或者：

http://download.fedora.redhat.com/pub/epel/5/i386/rubygem-mongrel-1.0.1-6.el5.i386.rpm

5.3 配置 puppetmaster 让它启动多个端口支持。

编辑 /etc/sysconfig/puppetmaster 添加以下两行

PUPPETMASTER_PORTS=(18140 18141 18142 18143)

PUPPETMASTER_EXTRA_OPTS=" --servertype=mongrel --ssl_client_header=HTTP_X_SSL_SUBJECT"

/etc/init.d/puppetmaster restart 重启生效

5.4 安装配置 nginx

wget http://nginx.org/download/nginx-latest.tar.gz
tar zxf nginx-latest.tar.gz
./configure —with-http_stub_status_module —with-http_ssl_module
make && make install
注：如果你是采用 0.7 版本，可以参考：打二个包即可
http://www.masterzen.fr/2009/07/21/new-ssl-features-for-nginx/
$ cd nginx-0.7.59
$ patch -p1 < ../0001-Support-ssl_client_verify-optional-and-ssl_client_v.patch
$ patch -p1 < ../0002-Add-SSL-CRL-verifications.patch

Nginx 配置文件参考：

```
  upstream puppetmaster {
      server 127.0.0.1:18140;
      server 127.0.0.1:18141;
      server 127.0.0.1:18142;
      server 127.0.0.1:18143;
  }

 server {
     listen 8140;
     root                        /etc/puppet;

     ssl                         on;
     ssl_session_timeout         5m;
     ssl_certificate /var/lib/puppet /ssl/certs/puppet.example.com.cn.pem;
     ssl_certificate_key   /var/lib/puppet/ssl/private_keys/puppet.example.com.c
n.pem;
     ssl_client_certificate   /var/lib/puppet/ssl/ca/ca_crt.pem;
     ssl_crl                  /var/lib/puppet/ssl/ca/ca_crl.pem;
     ssl_verify_client          optional;

     # File sections
     location /production/file_content/files/ {
          types { }
          default_type application/x-raw;
          alias /etc/puppet/manifests/files/;
     }

     # Modules files sections
     location ~ /production/file_content/modules/.+/ {
          root /etc/puppet/modules;
          types { }
          default_type application/x-raw;
          rewrite ^/production/file_content/modules/(.+)/(.+)$ /$1/files/$2 break;
```

```
    }

    # Ask the puppetmaster for everything else
    location / {
        proxy_pass          http://puppetmaster;
        proxy_redirect      off;
        proxy_set_header    Host             $host;
    proxy_set_header    X-Real-IP        $remote_addr;
    proxy_set_header    X-Forwarded-For  $proxy_add_x_forwarded_for;
    proxy_set_header    X-Client-Verify  $ssl_client_verify;
    proxy_set_header    X-SSL-Subject    $ssl_client_s_dn;
    proxy_set_header    X-SSL-Issuer     $ssl_client_i_dn;
    proxy_buffer_size          16k;
    proxy_buffers               8 32k;
    proxy_busy_buffers_size    64k;
    proxy_temp_file_write_size 64k;
    proxy_read_timeout          65;
    }
}
```

同理：代理的话，可以把 puppetmaster 分开，安装多台。然后 upstream

六、使用 Puppet

6.1 核心配置文件

Puppet.conf 主配置文件
fileserver.conf 允许访问的文件控制
auth.conf 访问权限配置
site.conf 全局配置文件
tagmail.conf 邮件报警配置

6.2 文件服务

6.3 类

6.4 模块

可以把一个类，资源，文件整合到一个模块里。然后在 PP 里 import 这个模块就可以使用这个模块里定义的所有资源

七、web gui

7.1 Dashboard

7.1.1

Puppet Dashboard 可以为你的 Puppet 环境添加一个图形用户界面（GUI）。Puppet Dashboard 可以显示主机上 Puppet 的运行结果.如果不成功就会 fail。你也可以用来管理你的主机，但我这里并没有使用。

本文按网方文档安装完成：http://docs.puppetlabs.com/guides/installing_dashboard.html

7.1.2 安装 EPEL 和 MYSQL

rpm -Uvh http://download.fedora.redhat.com/pub/epel/5/i386/epel-release-5-3.noarch.rpm

yum install -y mysql mysql-devel mysql-server ruby ruby-devel ruby-irb ruby-mysql ruby-rdoc ruby-ri

如果 MYSQL 是源码包安装则：

yum install -y ruby ruby-devel ruby-irb ruby-mysql ruby-rdoc ruby-ri

7.1.3 安装 rubygem 1.3.5 和 rake git

安装 rubygem 1.3.5 请不要安装 1.3.6 及更高版本。不支持。。哈哈

http://production.cf.rubygems.org/rubygems/rubygems-1.3.5.tgz

tar xfz rubygems-1.3.5.tgz

cd rubygems-1.3.5

ruby setup.rb

使用 gem 安装 rake

gem install rake

安装 git 下载 dashboard 包

yum -y install git

7.1.4 安装和配置 dashboard

下载 puppet-dashboard

我把它放在/data/www/wwwroot/

mkdir -p /data/www/wwwroot/

cd /data/www/wwwroot/

git clone git://github.com/puppetlabs/puppet-dashboard.git

修改数据配置 YML 文件

/data/www/wwwroot/puppet-dashboard/config/database.yml (放那自己定，注意看配置文件)

cp database.yml.example database.yml

```
production:
  database: root
  username: dashboard
  password: linuxtone
  encoding: utf8
adapter: mysql
```

7.1.5 配置 DB

创建库和表：

1. 使用 rake 创建：

回到 puppet-dashboard 目录执行

rake RAILS_ENV=production db:create

2 手动创建

CREATE DATABASE dashboard CHARACTER SET utf8;
CREATE USER 'dashboard'@'localhost' IDENTIFIED BY 'my_password';
GRANT ALL PRIVILEGES ON dashboard.* TO 'dashboard'@'localhost';

创建表：rake RAILS_ENV=production db:migrate

我在使用 development 的时候老是报错。如下：

rake db:migrate db:test:prepare --trace
(in /data/www/wwwroot/puppet-dashboard)
** Invoke db:migrate (first_time)
** Invoke environment (first_time)
** Execute environment
** Execute db:migrate
rake aborted!
Access denied for user 'root'@'localhost' (using password: NO)

因此我换成了 production。大家可以仔细看官网说明。尽管很烂，但也能解决问题

7.1.6 production 与 development 区别：

启动时：

production ：Start a production server on port 3000:
./script/server -e production

development：start a development server on port 8080, where the development environment is used by default:
/script/server -p 8080

7.1.7 启动与运行

运行：先查看位置：puppetmasterd --configprint libdir
cp ext/puppet/puppet_dashboard.rb /var/lib/puppet/reports/
chmod 644 /var/lib/puppet/reports/puppet_dashboard.rb
#more puppet_dashboard.rb
HOST = 'localhost' #主机
PORT = 3000 #启动端口

7.1.8 可以导入之前的数据

导入：Import existing reports 详见官网
rake RAILS_ENV=production reports:import REPORT_DIR=/path/to/your/reports
本例：rake RAILS_ENV=production reports:import REPORT_DIR=/var/lib/puppet/reports/

注：/var/lib/puppet/reports/ 为 puppet.conf 里 vardir 的定义目录

7.1.9 配置 puppet

配置 puppetmaster

```
[master]
reports = http, store
reporturl = http://IP:3000/reports
modulepath = /etc/puppet/modules

puppetclient
server = puppetmaster.linuxtone.org
listen = true
report = true
runinterval = 10
```

7.1.10 dashborad init 启动脚本（注意修改内容）

```
# more /etc/init.d/puppet-dashboard   注意修改脚本里的 IP
#!/bin/bash
# Description: Puppet Dashboard init.d script

# Get function from functions library
. /etc/init.d/functions

# Start the service Puppet Dashboard
start() {
        echo -n "Starting Puppet Dashboard: "
        /usr/bin/ruby  /data/www/wwwroot/puppet-dashboard/script/server  -e  production
-b IP >/dev/null 2>&1 &
        ### Create the lock file ###
        touch /var/lock/subsys/puppetdb
        success $"Puppet Dashboard startup"
        echo
}

# Restart the service Puppet Dashboard
stop() {
        echo -n "Stopping Puppet Dashboard: "
        kill        -9        `ps        ax        |        grep        "/usr/bin/ruby
/data/www/wwwroot/puppet-dashboard/script/server" | grep -v grep | awk '{ print $1 }'
` >/dev/null 2>&1
        ### Now, delete the lock file ###
        rm -f /var/lock/subsys/puppetdb
        success $"Puppet Dashboard shutdown"
        echo
}
```

```
### main logic ###
case "$1" in
  start)
        start
        ;;
  stop)
        stop
        ;;
  status)
        status Puppet DB
        ;;
  restart|reload|condrestart)
        stop
        start
        ;;
  *)
        echo $"Usage: $0 {start|stop|restart|reload|status}"
        exit 1
esac

exit 0
```

chmod 755 /etc/init.d/puppet-dashboard
/etc/init.d/puppet-dashboard start

7.2 foreman

简单说下安装的顺序：

1. 安装 puppet
2. 升级 ruby 至 1.8.6
3. 安装相关的依赖 gem rake rails i18n
4. 创建库和用户并授权
5. 安装 foreman
6. 修改 database.yml
7. 创建 foreman 表
8. 配置 foreman.rb 文件，并放至 puppet reports 目录
9. 配置 puppet server client 配置文件
10. 启动

注意版本的要求。特别是 ruby 只能是 1.8 升级 ruby 参考：http://bubbyroom.com/2011/01/centos-yum-update-ruby/

Ruby 1.9 is not supported yet. You have to use Ruby 1.8.x as stated above.
RubyGems 1.3.1 or higher is required
Rake 0.8.3 or higher is required

Rack 1.0.1 is required. If you don't have this exact version, database migration would fail.
I18n 0.4.2 is required for Redmine >= 1.0.5
rpm -Uvh http://download.fedora.redhat.com/pub/epel/5/i386/epel-release-5-3.noarch.rpm
yum install -y mysql mysql-devel mysql-server ruby ruby-devel ruby-irb ruby-mysql ruby-rdoc ruby-ri
如果 MYSQL 是源码包安装则：

yum install -y ruby ruby-devel ruby-irb ruby-mysql ruby-rdoc ruby-ri
安装 rubygem 1.3.5

http://production.cf.rubygems.org/rubygems/rubygems-1.3.5.tgz
tar xfz rubygems-1.3.5.tgz
cd rubygems-1.3.5
ruby setup.rb
安装 rails 和 rack

gem install rails -v=2.3.5
gem install rack -v=1.0.1
gem install -v=0.4.2 i18n
下载 foreman
地址：http://www.redmine.org/projects/redmine/wiki/Download

我喜欢用 GIT 下载：
有人问过我 git 怎么安装：

yum -y install git
git clone git://github.com/edavis10/redmine.git
源码包地址：

wget http://rubyforge.org/frs/download.php/73900/redmine-1.1.0.tar.gz
下载 后解压。可以放在/usr/local/redmine 目录

Mysql 配置： 创建库和用户

create database redmine character set utf8;
create user 'redmine'@'localhost' identified by 'my_password';
grant all privileges on redmine.* to 'redmine'@'localhost';
mysql 5 以上

grant all privileges on redmine.* to 'redmine'@'localhost' identified by 'my_password';
进去 foreman 配置 database.yml
config/database.yml

production:

adapter: mysql
database: redmine
host: localhost
username: redmine
password: my_password
建表：

RAILS_ENV=production rake db:migrate
启动：

ruby script/server webrick -e production
库备份：

/usr/bin/mysqldump-u -p
|gzip > /path/to/backup/db/redmine_`date +%y_%m_%d`.gz
puppet master 配置

reports=log, foreman
puppet client 配置

report = true
配置 puppet 提交至 foreman
cp extras/puppet/foreman/files/foreman-report.rb /usr/lib/ruby/site_ruby/1.8/puppet/reports/foreman.rb
chmod 644 /usr/lib/ruby/site_ruby/1.8/puppet/reports/foreman.rb
vim /usr/lib/ruby/site_ruby/1.8/puppet/reports/foreman.rb

URL of your Foreman installation
$foreman_url="http://" + `hostname`.strip + ":8000"
cron 清数据：

rake reports:expire days=7 RAILS_ENV="production"
##

foreman 可以采用 yum 安装

配置源：

cat > /etc/yum.repos.d/foreman.repo << EOF
[foreman]
name=Foreman Repo
baseurl=http://theforeman.org/repo
gpgcheck=0
enabled=1

EOF
yum install foreman
使用 yum 安装后会产生/etc/init.d/foreman 配置文件也在/etc/foreman 目录 。其它的配置一样。

我在安装时遇到过一些错误，但主要是 ruby 版本 和 mysql 依赖 建议都用 yum 来安装

八、案例

8.1 cron

8.2 syslog

写 2 个复杂的吧。以上 2 个先忽略了。

8.3 Haproxy

先创建一个 haproxy.conf 的 ruby 写的 erb 模板

贴出配置以供参考：

```
more modules/haproxy/templates/haproxy.conf.erb
# this config needs haproxy-1.1.28 or haproxy-1.2.1

global
    log 127.0.0.1 local3 info
    maxconn 4096
    chroot /var/lib/haproxy
    uid 99
    gid 99
    daemon
    #debug
    #quiet

defaults
    log  global
        mode    http
        option  httplog
        option  dontlognull
        option  httpclose
        option  forwardfor
        retries 3
        balance uri
        maxconn 2000
        stats uri /stats
        stats auth admin:linuxtone
        contimeout      5000
        clitimeout      50000
        srvtimeout      50000

listen  web_proxy 0.0.0.0:80
```

```
    option httpchk    GET /test.jpg

        <% instance=1 %>
        <% proxy_name.each do |ip| %>server app1_<%=instance%> <%=ip%>:80
    cookie app1inst<%=instance%> check inter 2000 rise 2 fall 5
        <% instance+=1
        end %>    #这里用了一个循环语法。写法很简单。看完完整的配置一般都能明
    白。

    对应的 INIT 如下：
    #class: haproxy
#   haproxy.conf.erb    => haproxy.conf

class haproxy {
    package { haproxy: ensure => installed }
     #安装
     file {
            "/etc/init.d/haproxy":
            mode => 755, owner => root, group => root,
            require => Package[haproxy],
            source => "puppet://$fileserver/files/haproxy/haproxy"
     }
     #通过 file  资源取 init  启动脚本

    file { "haproxy.conf":
           name => "/etc/haproxy/haproxy.conf",
           mode => "644",owner => root, group => root,
           ensure => $ensure,
           require => Package["haproxy"],
           content => template("haproxy/haproxy.conf.erb"),
     }
     #通过 erb 模板取配置文件

     service {
           "haproxy":
           ensure => running,
           enable => true,
           hasrestart => true,
           hasstatus => true,
           subscribe => File["haproxy.conf"],
     }
     #配置是否启动。配置文件更新后自动重启。服务停止后，是否启动。
}
```

最后在 manifests 里的 nodes.pp 里加上 proxy_name 相关的信息。
$proxy_name = [“proxy’ip1”, “proxy’ip2”] #定义后。ERB 模板里的变量就能自动获取了。

8.4 Apache Traffic Server

ATS 的配置和 HA 的配置类似。引用等也相同。就不准备贴了。
如果有兴趣可以和我沟通。

九、 example42

http://www.example42.com/ **有大量的例子可以参考。**

十、 高级应用

十一、 FAQ

十二、 参考及致谢

http://blog.chinaunix.net/space.php?uid=16480950
http://puppet.chinaec2.com
http://projects.reductivelabs.com/projects/puppet/wiki/Using_Mongrel_Nginx
http://dywer.blog.51cto.com

所有的配置参考过以上的 BLOG，感谢他们。我也在学习过程当中。目前由于工作一直忙。加上前段时间家里有事。文档一直没有写全。现在也没有全。因为我也在探索。我也需要更多的运维经验，我才能跟大家更好的沟通与交流。

第一版就先这样吧。如果有错误。字打错了。或完全的配置错误。或更好的改进方法。请发 MAIL 给我。

由于 GMAIL 被 XX 了。就发到 LT 的吧。Liuyu#linuxtone.org bubbyroom#163.com