SaltStack 配置管理系统

一、SaltStack 介绍

1.1. 简介
1.2. 适用环境
1.3. 原理
1.4. 常用架构

二、SaltStack 安装

2.1.安装环境介绍
Centos 5.x，salt 版本 0.71.1
2.2.安装
2.2.1.服务端安装
wget http://dl.cpis-opt.com/huanw/shencan/epel-release-5-4.noarch.rpm
rpm -vih epel-release-5-4.noarch.rpm
yum -y install salt-master

2.2.2.客户端安装
wget http://dl.cpis-opt.com/huanw/shencan/epel-release-5-4.noarch.rpm
rpm -vih epel-release-5-4.noarch.rpm
yum -y install salt-minion

2.3.服务启动和停止
2.3.1.服务端启动和停止
/etc/init.d/salt-master start #启动
/etc/init.d/salt-master stop #停止

2.3.2.客户端启动和停止
/etc/init.d/salt-minon start #启动
/etc/init.d/salt-minon stop #停止

三、SaltStack 配置

3.1.服务端配置
3.2.客户端配置
3.3.授权验证

四、SaltStack 服务端常用组织结构

五、案例分析

待续...

六、常用模块实例详解

6.1. pkg 模块使用详解

6.1.1. 简介

pkg 模块作用是包管理，包括增删更新

6.1.2. 参数详解

name: *包名称*

6.1.3.案例

a).安装单个包，安装 httpd 包

```
httpd:
  pkg.installed
```

或者

```
lg_httpd:
  pkg.installed：
      - name: httpd
```

b).安装一系列软件包

```
mypkgs:
    pkg.installed:
        - pkgs:
            - foo
            - bar
            - baz
```

c).指定软件版本 <, <=, >=, and >

```
mypkgs:
    pkg.installed:
```

```
            - pkgs:
                - foo
                - bar: 1.2.3-4
                - baz

    mypkgs:
      pkg.installed:
        - pkgs:
            - foo
            - bar: '>=1.2.3-4'
            - baz
```

d). 如果有 rpm，那么安装 rpm

```
    mypkgs:
        pkg.installed:
            - sources:
            - foo: salt://rpms/foo.rpm
            - bar: http://somesite.org/bar.rpm
            - baz: ftp://someothersite.org/baz.rpm
            - qux: /minion/path/to/qux.rpm
```

d).升级软件包到最新

```
    mypkgs:
        pkg.latest:
            - pkgs:
            - foo
            - bar
            - baz
```

f).删除软件包

```
    foo:
        pkg.removed
```

g.删除多个软件包

```
    mypkgs_remove:
        pkg.removed:
            - pkgs:
            - foo
            - bar
```

6.2. file 模块使用详解

6.2.1. 简介

file 模块作用管理文件操作，包括同步文件，设置文件权限，所属用户组,删除文件等操作。

6.2.2. 参数详解

file.managed 文件管理所需参数详解：

name： *要同步到 minon 上的文件路径+文件名*

source: *同步文件源为 master 上的文件*

backup *参数：备份作用， 如果有改动才会备份*

#backup: .bak 会在 minion 上该文件的同级目录下加.bak 备份

#backup: minion,会备份到 minion 下的 /var/cache/salt/minion/backup_files 下

mode:*权限设置，例如 644*

owner:*所属设置*

group:*所属组设置*

file.append 向文件中添加内容，所需参数详解:

text: *向文件中添加内容信息。*

file.comment 文件注释操作参数详解:

char : *指定注释使用的字符，默认是#*

regex: 需要跟文件中每一行中信息匹配的正则表达式。

file.copy 把 minon 本地文件复制到某个目录下参数详解：

source: *minon 本地文件*

makedirs: *如果复制的目的目录不存在是否要创建*

force: *如果复制目的文件名和源文件相同的文件名，是否覆盖*

file.directory 目录创建参数详解：

user: *对新创建文件夹设置所属用户*

group: *对新创建文件夹设置所属组*

dir_mode: *对新创建文件夹设置访问权限*

file_mode: *对新创建文件夹下文件设置权限*

recurse: *递归接受以上设置，用户组等等*

makedirs: *强制建立文件夹 = mkdir -p*

#注意 以上参数只有对新建的文件夹设置相应的属性，原来存在属性都不变更

file.exists：判断文件是否存在.

file.absent:删除文件

file.symlink 建立 link 链接参数详解：

name:*源文件*

target: *目的文件*
force: *是否强制创建*

file.touch 创建空文件,或者修改文件 三个时间
name:*文件名*
atime： *访问时间*
mtime: *修改时间*
makedirs: *如果目录不存在，是否创建目录*

file.sed 使用 minon 的 sed 命令操作，参数详解:
before: *源字符串*
after: *替换为字符串*
limit: *该行中匹配正则表达式*

6.2.3.案例

a).yum 安装源配置文件同步

```
yum:
    file.managed:
        - name: /etc/repo.d/centos-base.repo
        - source: salt://files/centos-base.repo
        - mode: 644
        - owner: root
        - group: root
```

b).同步前先备份文件

```
/tmp/c.log:
  file.managed:
    - source:
      - salt://a.log
    - backup: .bak
```

或者

```
/tmp/dd.log:
  file.managed:
    - source:
      - salt://a.log
    - backup: minion
```

c).向 minon 的文件中追加内容

```
/tmp/d.log:
  file.append:
    - text: |
        Thou hadst better eat salt with the Philosophers of Greece,
        than sugar with the Courtiers of Italy.
```

```
            - Benjamin Franklin
```

或者:

```
/tmp/e.log:
  file.append:
    - text:
      - Trust no one unless you have eaten much salt with him.
      - "Salt is born of the purest of parents: the sun and the sea."
```

或者

```
#a.log b.log append /tmp/bb.log
/tmp/bb.log:
  file:
    - append
    - sources:
      - salt://a.log
      - salt://b.log
```

d)添加注释 ,默认添加#

```
/tmp/d.log:
  file.comment:
    - regex: ^s
```

e)把本地的文件拷贝到其他的目录中

```
/tmp/abc/h.log:
  file.copy:
    - source: /tmp/c.log
    - makedirs: True
    - force: True
```

f)创建目录，并作相应设置

```
/tmp/aa/cc/dd/ff:
    file.directory:
        - user: root
        - group: root
        - dir_mode: 755
        - file_mode: 644
        - makedirs: True
        - recurse:
            - user
            - group
            - mode
```

g).判断文件存在不存在,也可以是目录

```
/tmp/a.log:
    file.exists
```

h)设置链接

```
/tmp/a1.log:
  file.symlink:
    - target: /tmp/alink.log
    - force: True
```

i)创建一个空文件

```
/tmp/fff.log:
  file.touch
```

j).删除文件：

```
/tmp/ss.log:
  file.absent
```

k)使用 sed 替换文件中字符串

```
/tmp/sed.log:
  file.sed:
    - before: 'ldap'
    - after: 'ABC'
    - limit: 's'
```

l)高级运用 使用模板

```
/tmp/template_test.conf:
  file.managed:
    - source: salt://template_test.conf
    - mode: 644
    - owner: root
    - group: root
    - template: jinja
```

template_test.conf 文件中的内容---------->

```
{% if grains['os'] == 'Ubuntu' %}
host: {{ grains['host'] }}
{% elif grains['os'] == 'CentOS' %}
host: {{ grains['fqdn'] }}
{% endif %}
```

注意点：

文件回滚操作？ 思考？临时解决办法：

1.可以编写模块等等方法,

2.备份文件覆盖

6.3. cmd 模块使用详解

6.3.1. 简介

cmd 模块作用在 minon 上执行命令或者脚本。

6.3.2. 参数详解

cmd.run 参数有：

name: *执行脚本或者命令的名字*
onlyif: *测试命令，如果执行测试命令返回 true，cmd.run 的命令才有可能执行。*
unless: *与 onlyif 相反，如果执行后为 false，cmd.run 的命令才有可能执行。*
cwd: *执行命令当前目录设置，默认为 /root*
user： *执行命令的用户,默认 root*
group： *执行命令的组,默认 root*
shell： *执行命令使用的 shell*
env： *执行命令的环境设置。*
umask： *运行命令时候 umask 设置。*
output_loglevel： *执行命令日志输出级别，其中特殊的设置是，quiet，那么就不会输出日志。*
timeout： *执行命令的超时时间，如果超时就发送 kill -l SIGTERM 命令，如果"kill -l SIGTERM"被忽略了，那么紧接着发送，kill 命令.*

cmd.wait 参数：

与 cmd.run 中的参数相比，只是 wait 中没有 timeout 参数，其他参数意义一样。

注意点：

a). cmd.run 和 cmd.wait 区别，多个判断条件时候，
只要有一个满足就会执行 cmd.run
只有所有的都满足(more desirable 超符合条件)才会执行 cmd.wait
官方解释如下:
Should I use cmd.run or cmd.wait?

These two states are often confused. The important thing to remember about them is that cmd.run states are run each time the SLS file that contains them is applied.
If it is more desirable to have a command that only runs after some other state changes, then cmd.wait does just that. cmd.wait is designed to watch other states, and is executed when the state it is watching changes.

官方在解释 cmd.run 时候有一句话，使用 watch 时候请使用 cmd.wait，原文如下：
Run a command if certain circumstances are met. Use cmd.wait if you want to use the watch requisite.

6.3.3.案例详解

a).执行再/tmp 目录下，使用 nobody 用户执行脚本/tmp/myscript.

```
Run myscript:
  cmd.run:
    - name: /tmp/myscript
    - cwd: /
    - user: nobody
```

b).判断 httpd 配置文件是否改动，如果改动，然后同步，然后测试配置文件是否可用，如果可用，重新加载配置文件。

```
"/etc/httpd/conf/httpd.conf":
  file.managed:
    - source: salt://httpd/conf/httpd.conf

reloadhttpd:
  cmd.wait:
    - name: "/etc/init.d/httpd graceful"
    - onlyif: "/etc/init.d/httpd configtest"
    - watch:
      - file: /etc/httpd/conf/httpd.conf
```

注意：

"/etc/httpd/conf/httpd.conf" 配置文件这块，最好用引号，因为有时候你的这个里边有什么特殊字符，那么下边就不认识，造成问题，0.17.1 中有这个 bug。

6.4.user 模块使用详解

6.4.1. 简介

user 模块作用管理系统账户操作。

6.4.2. 参数详解

user.present 参数中 uid、gid、home 等选项没有指定是系统默认指定，参数详解：

name: *要创建的用户名.*

uid: *指定 uid。*

gid: *指定默认的组 gid*

groups: *指定用户其他所属组，如果组在 minion 上不存在，则会报错. 如果设置*

会空，将会删除除默认组外的其他所属组。

home: *是否创建用户家目录。*

password: *设置用户 hash 之后的密码.*

enforce_password: *如果为 False，password 参数的密码与原密码不同，将保持原密码不做更改.如果没有设置 password 选项，该选项将自动忽略掉.*

shell: *指定用户的登陆后的 shell，默认将设置为系统默认 shell。*

user.absent 用于删除用户参数详解：

name: *指定需要删除的用户名。*

purge: *是否删除用户家目录。*

force: *如果用户当前已登录，则 absent state 会失败. 选项为 True 时，就算用户当前处于登录状态也会删除本用户.*

6.4.3. 案例详解

a).创建用户 laoseng，执行密码，shell，home 目录，uid，gid，其他所属组。

```
laoseng:
  user.present:
    - password: '6$JyhDBiOi5ZyvaDWm'
    - shell: /bin/bash
    - home: /home/laoseng
    - uid: 888
    - gid: 888
    - groups:
      - wheel
      - storage
```

b).创建用户 laoseng

```
laoseng:
  user.absent
```

6.5.service 模块使用详解

6.5.1. 简介

Service 模块作用管理系统服务操作，包括 start,stop,reload,restart

6.5.2. 参数详解

name: *服务名称*
enable: *设置自启动 True，默认不设置*
watch： *监控一些条件，如果成立默认执行 restart*
reload: *默认为 false，如果设置为 True 时候，watch 条件成立，会 reload，不会 restart*

6.5.3. 案例

a).apache 服务管理

```
httpd:
  service:
    - running
    - enable: True
    - reload: True
    - watch:
      - pkg: httpd
```

6.6.cron 模块使用详解

6.6.1. 简介

Cron 模块作用管理 cron 服务操作

6.6.2. 参数详解

cron.present 添加 cron 任务， 参数:

name: *资源名字*
user: *默认 root，或者指定用户执行*
minute,hour,daymonth,month,dayweek: *参数默认是**

cron.absent 删除 cron 任务。

6.6.3.案例

a).添加 cron 任务

```
/path/to/cron/script:
  cron.present:
    - user: root
```

```
        - minute: random
        - hour: 2
        - daymonth: '*'
        - month: '*/2'
        - dayweek: 4
        - comment: "this script is rsync pic"
```

或者

```
rsync_pic:
   cron.present:
        - name: "/bin/sh /tmp/script.sh >/dev/null 2>&1"
        - comment: "this script.sh is rsync log"
        - user: root
        - minute: random
        - hour: 2
        - month: '*/2'
        - dayweek: 4
```

b).删除 cron 任务

```
        /path/to/cron/script:
           cron.absent
```

6.7. hosts 模块使用详解

6.7.1. 简介

hosts 模块作用管理/etc/hosts 文件

6.7.2. 参数详解

host.present 添加域名，参数：

name: *域名*

names: *多个域名*

ip: *ip 地址*

host.absent 参数域名。

6.7.1 案例

a).添加单个 ip 与单个域名对应

```
        host2:
           host.present:
              - ip: 192.168.0.42
```

```
        - name:
          - 1.abc.com
```

b).添加单个 ip 与多域名对应

```
    host1:
      host.present:
        - ip: 192.168.0.42
        - names:
          - 1.abc.com
          - test.test.com
```

c):删除 ip 对应的域名

```
    host3:
      host.absent:
        - ip: 192.168.0.42
        - name: 1.abc.com
```

七、高级综合应用案例详解

7.1. 综合一

yum 安装软件包,,设置 apache 用户和组,设置同步配置文件,并启动服务,这里以 apache 为例。

a). 分析需要使用的模块

pkg 模块，user 模块，组模块，service 模块，配置文件模块。

b). 建立之间关系。

首先，安装模块，然后建立用户，设置 httpd 目录，然后是同步配置文件，然后才能启动服务。

c).编写代码

```
apache:
  pkg:
    - installed
  service:
    - running
    - watch:
      - pkg: apache
    - file: /etc/httpd/conf/httpd.conf
      - user: apache
  user.present:
    - uid: 87
    - gid: 87
    - home: /var/www/html
    - shell: /bin/nologin
    - require:
      - group: apache
```

```
    group.present:
      - gid: 87
      - require:
        - pkg: apache

/etc/httpd/conf/httpd.conf:
    file.managed:
      - source: salt://apache/httpd.conf
      - user: root
      - group: root
      - mode: 644
```

7.2 综合二

安装 http 软件包，设置用户组，以及 rootdocment 目录，最重要的是监控配置文件当配置文件发生变化时候，同步配置文件，然后检测配置文件是否正常，如果配置文件 ok，reload 服务，反之不要 reload 服务。

a). 分析需要使用的模块

pkg 模块，user 模块，组模块，cmd 模块，file 模块。

b). 建立之间关系。

首先，安装模块，然后建立用户，设置 httpd 目录，然后是同步配置文件，最重要是监控配置文件，检测配置文件，reload，这个完全与命令有关，所以 cmd 命令模块，搞定。

c).编写代码

```
httpd:
    pkg:
      - installed
    user.present:
      - uid: 87
        - gid: 87
      - home: /var/www/html
      - shell: /bin/nologin
      - require:
        - group: httpd
    group.present:
      - gid: 87
      - require:
        - pkg: httpd
"/etc/httpd/conf/httpd.conf":
  file.managed:
    - source: salt://httpd/conf/httpd.conf
    - user: httpd
    - group: httpd
```

```
      - require:
        - pkg: httpd
"/etc/httpd/conf.d/httpd-vhost.conf":
  file.managed:
      - source: salt://httpd/conf/httpd-vhost.conf
      - user: httpd
      - group: httpd
      - require:
        - pkg: httpd
reloadhttpd:
  cmd.wait:
      - name: "/etc/init.d/httpd graceful"
      - onlyif: "/etc/init.d/httpd configtest"
      - watch:
        - file: /etc/httpd/conf/httpd.conf
        - file: /etc/httpd/conf.d/httpd-vhost.conf
```

八、FAQ

待续...

九、参考

SaltStack 常用模块详解

官方文档：

http://docs.saltstack.com/

官方文档中模块

http://docs.saltstack.com/ref/modules/all/index.html

安装参考：

http://wiki.saltstack.cn/installation