# 自动化运维工具Ansible详细部署

==========================================================================================

## **一、基础介绍**

==========================================================================================

### **1、简介**

ansible是新出现的自动化运维工具，基于Python开发，集合了众多运维工具（puppet、cfengine、chef、func、fabric）的优点，实现了批量系统配置、批量程序部署、批量运行命令等功能。ansible是基于模块工作的，本身没有批量部署的能力。真正具有批量部署的是ansible所运行的模块，ansible只是提供一种框架。主要包括：

(1)、连接插件connection plugins：负责和被监控端实现通信；

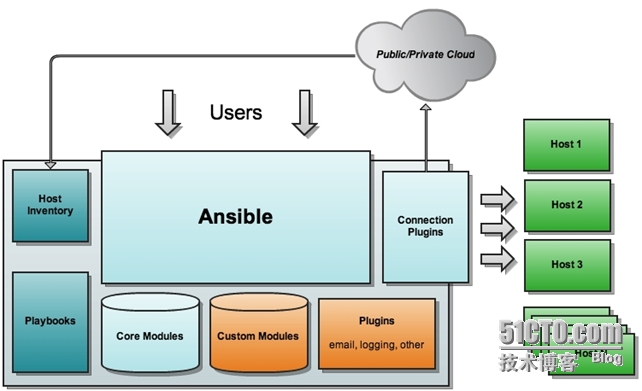
(2)、host inventory：指定操作的主机，是一个配置文件里面定义监控的主机；

(3)、各种模块核心模块、command模块、自定义模块；

(4)、借助于插件完成记录日志邮件等功能；

(5)、playbook：剧本执行多个任务时，非必需可以让节点一次性运行多个任务。

### **2、总体架构**

[](http://s3.51cto.com/wyfs02/M02/53/A7/wKiom1Rsxz3ToUCAAAGROYAM3EI989.jpg)

### **3、特性**

(1)、no agents：不需要在被管控主机上安装任何客户端；

(2)、no server：无服务器端，使用时直接运行命令即可；

(3)、modules in any languages：基于模块工作，可使用任意语言开发模块；

(4)、yaml，not code：使用yaml语言定制剧本playbook；

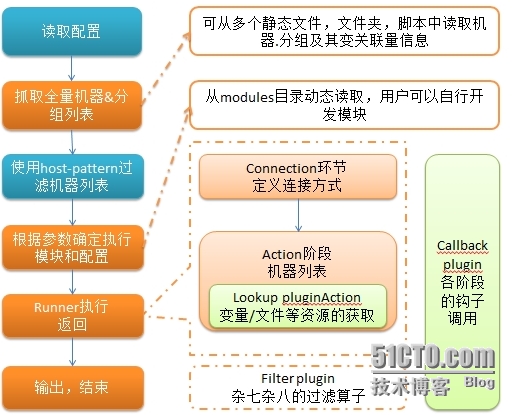
(5)、ssh by default：基于SSH工作；

(6)、strong multi-tier solution：可实现多级指挥。

### 4、优点

(1)、轻量级，无需在客户端安装agent，更新时，只需在操作机上进行一次更新即可；  
(2)、批量任务执行可以写成脚本，而且不用分发到远程就可以执行；  
(3)、使用python编写，维护更简单，ruby语法过于复杂；  
(4)、支持sudo。

### **5、任务执行流程**

[](http://s3.51cto.com/wyfs02/M01/53/A7/wKiom1Rsx2uQYJZ5AAJplY08vOQ976.jpg)

说明：

(1)、以上内容大多是基于他人分享的基础上总结而来，学习借鉴之用；

(2)、本次安装基于 CentOS 6.5 系统环境。

==========================================================================================

## **二、Ansible基础安装与配置**

==========================================================================================

### **1、Ansible基础安装**

#### **(1)、python2.7安装**

<https://www.python.org/ftp/python/2.7.8/Python-2.7.8.tgz>

# tar xvzf Python-2.7.8.tgz

# cd Python-2.7.8

# ./configure --prefix=/usr/local

# make --jobs=`grep processor/proc/cpuinfo | wc -l`

# make install

## 将python头文件拷贝到标准目录，以避免编译ansible时，找不到所需的头文件

# cd /usr/local/include/python2.7

# cp -a ./\* /usr/local/include/

## 备份旧版本的python，并符号链接新版本的python

# cd /usr/bin

# mv python python\_

# ln -s /usr/local/bin/python

## 修改yum脚本，使其指向旧版本的python，已避免其无法运行

# vim /usr/bin/yum

#!/usr/bin/python  -->  #!/usr/bin/python\_

#### **(2)、setuptools模块安装**

<https://pypi.python.org/packages/source/s/setuptools/setuptools-7.0.tar.gz>

# tar xvzf setuptools-7.0.tar.gz

# cd setuptools-7.0

#yum install zlib\* gcc libffi-devel python-devel openssl-devel python-setuptools

# python setup.py install

#### **(3)、pycrypto模块安装**

<https://pypi.python.org/packages/source/p/pycrypto/pycrypto-2.6.1.tar.gz>

# tar xvzf pycrypto-2.6.1.tar.gz

# cd pycrypto-2.6.1

# python setup.py install

#### **(4)、PyYAML模块安装**

<http://pyyaml.org/download/libyaml/yaml-0.1.5.tar.gz>

# tar xvzf yaml-0.1.5.tar.gz

# cd yaml-0.1.5

# ./configure --prefix=/usr/local

# make --jobs=`grep processor/proc/cpuinfo | wc -l`

# make install

<https://pypi.python.org/packages/source/P/PyYAML/PyYAML-3.11.tar.gz>

# tar xvzf PyYAML-3.11.tar.gz

# cd PyYAML-3.11

# python setup.py install

#### **(5)、Jinja2模块安装**

<https://pypi.python.org/packages/source/M/MarkupSafe/MarkupSafe-0.9.3.tar.gz>

# tar xvzf MarkupSafe-0.9.3.tar.gz

# cd MarkupSafe-0.9.3

# python setup.py install

<https://pypi.python.org/packages/source/J/Jinja2/Jinja2-2.7.3.tar.gz>

# tar xvzf Jinja2-2.7.3.tar.gz

# cd Jinja2-2.7.3

# python setup.py install

#### **(6)、paramiko模块安装**

<https://pypi.python.org/packages/source/e/ecdsa/ecdsa-0.11.tar.gz>

# tar xvzf ecdsa-0.11.tar.gz

# cd ecdsa-0.11

# python setup.py install

<https://pypi.python.org/packages/source/p/paramiko/paramiko-1.15.1.tar.gz>

# tar xvzf paramiko-1.15.1.tar.gz

# cd paramiko-1.15.1

# python setup.py install

#### **(7)、simplejson模块安装**

<https://pypi.python.org/packages/source/s/simplejson/simplejson-3.6.5.tar.gz>

# tar xvzf simplejson-3.6.5.tar.gz

# cd simplejson-3.6.5

# python setup.py install

#### **(8)、ansible安装**

<https://github.com/ansible/ansible/archive/v1.7.2.tar.gz>

# tar xvzf ansible-1.7.2.tar.gz

# cd ansible-1.7.2

# python setup.py install

### **2、Ansible配置**

#### **(1)、SSH免密钥登录设置**

## 生成公钥/私钥

# ssh-keygen -t rsa

[wKioL1RsyCTC94M0AACbRu5nOSQ166.jpg](http://s3.51cto.com/wyfs02/M02/53/A5/wKioL1RsyCTC94M0AACbRu5nOSQ166.jpg)

## 写入信任文件（将/root/.ssh/id\_rsa.pub分发到其他服务器，并在所有服务器上执行如下指令）：

# cat /root/.ssh/id\_rsa.pub >> /root/.ssh/authorized\_keys

# chmod 600 /root/.ssh/authorized\_keys

#### **(2)、ansible配置**

# mkdir -p /etc/ansible

# vim /etc/ansible/ansible.cfg

……

remote\_port = 22

private\_key\_file = /root/.ssh/id\_rsa

……

## 主机组定义

# vim /etc/ansible/hosts

[hpc]

10.223.55.100

10.223.55.101

10.223.38.226

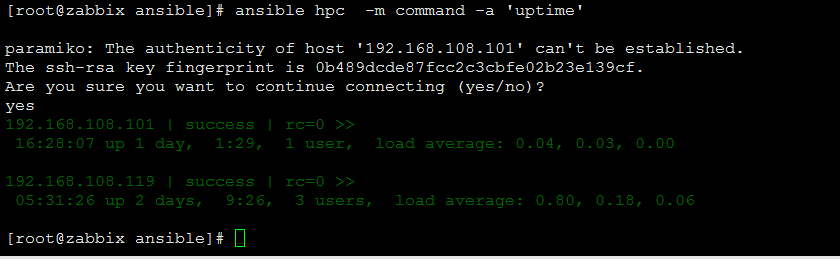
10.223.38.227

10.223.39.216

10.223.25.123

#### **(3)、简单测试**

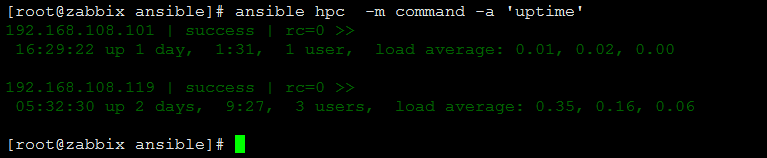
# ansible hpc -m command -a 'uptime'



说明：第一次运行时，需要输入一下“yes”【进行公钥验证】，后续无需再次输入。

## 再次运行

# ansible hpc -m command -a 'uptime'

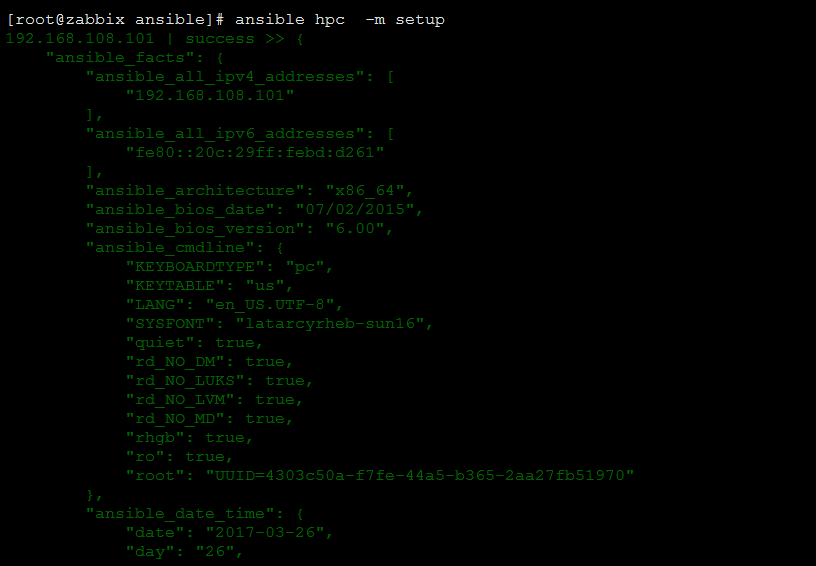


### **3、常用模块使用**

#### **(1)、setup**

## 用来查看远程主机的一些基本信息

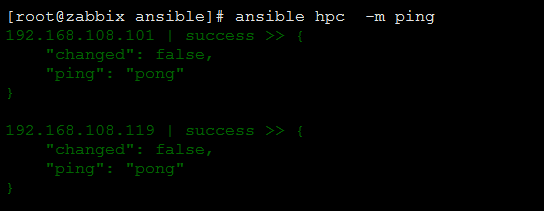
# ansible hpc -m setup



#### **(2)、ping**

## 用来测试远程主机的运行状态

# ansible hpc -m ping



#### **(3)、file**

## 设置文件的属性

相关选项如下：

force：需要在两种情况下强制创建软链接，一种是源文件不存在，但之后会建立的情况下；另一种是目标软链接已存在，需要先取消之前的软链，然后创建新的软链，有两个选项：yes|no

group：定义文件/目录的属组

mode：定义文件/目录的权限

owner：定义文件/目录的属主

path：必选项，定义文件/目录的路径

recurse：递归设置文件的属性，只对目录有效

src：被链接的源文件路径，只应用于state=link的情况

dest：被链接到的路径，只应用于state=link的情况

state：

       directory：如果目录不存在，就创建目录

       file：即使文件不存在，也不会被创建

       link：创建软链接

       hard：创建硬链接

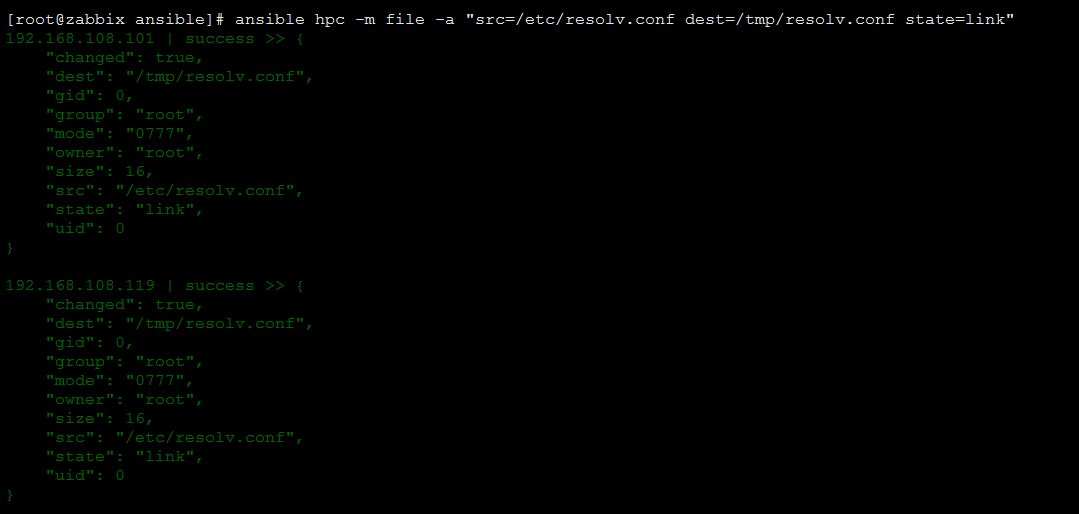
       touch：如果文件不存在，则会创建一个新的文件，如果文件或目录已存在，则更新其最后修改时间

       absent：删除目录、文件或者取消链接文件

示例：

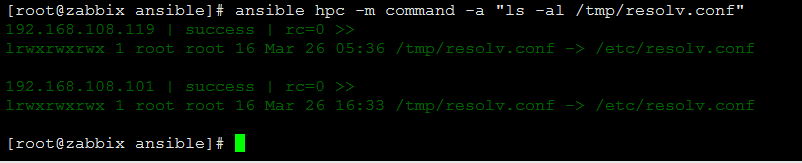
## 远程文件符号链接创建

# ansible hpc -m file -a "src=/etc/resolv.conf dest=/tmp/resolv.conf state=link"



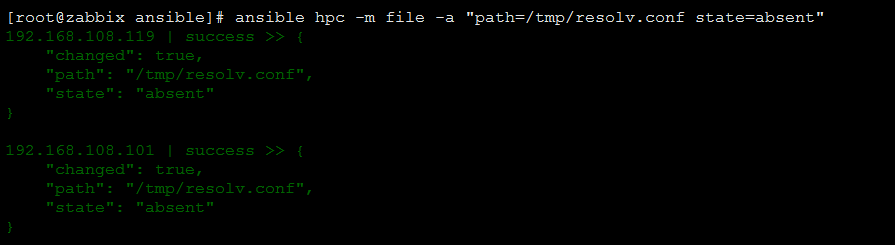
## 远程文件信息查看

# ansible hpc -m command -a "ls –al /tmp/resolv.conf"



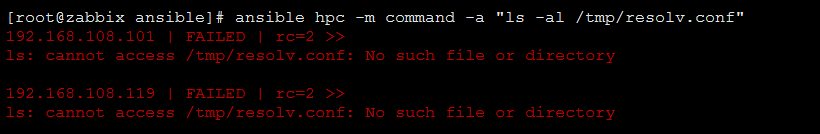
## 远程文件符号链接删除

# ansible hpc -m file -a "path=/tmp/resolv.conf state=absent"



## 远程文件信息查看

# ansible hpc -m command -a "ls -al /tmp/resolv.conf"



说明：如上显示，代表文件或链接已经删除。

#### **(4)、copy**

## 复制文件到远程主机

相关选项如下：

backup：在覆盖之前，将源文件备份，备份文件包含时间信息。有两个选项：yes|no

content：用于替代“src”，可以直接设定指定文件的值

dest：必选项。要将源文件复制到的远程主机的绝对路径，如果源文件是一个目录，那么该路径也必须是个目录

directory\_mode：递归设定目录的权限，默认为系统默认权限

force：如果目标主机包含该文件，但内容不同，如果设置为yes，则强制覆盖，如果为no，则只有当目标主机的目标位置不存在该文件时，才复制。默认为yes

others：所有的file模块里的选项都可以在这里使用

src：被复制到远程主机的本地文件，可以是绝对路径，也可以是相对路径。如果路径是一个目录，它将递归复制。在这种情况下，如果路径使用“/”来结尾，则只复制目录里的内容，如果没有使用“/”来结尾，则包含目录在内的整个内容全部复制，类似于rsync。

示例：

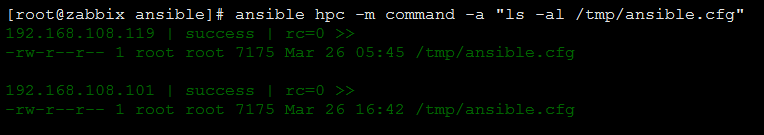
## 将本地文件“/etc/ansible/ansible.cfg”复制到远程服务器

# ansible hpc -m copy -a "src=/etc/ansible/ansible.cfg dest=/tmp/ansible.cfg owner=root group=root mode=0644"



## 远程文件信息查看

# ansible hpc -m command -a "ls -al /tmp/ansible.cfg"



#### **(5)、command**

## 在远程主机上执行命令

相关选项如下：

creates：一个文件名，当该文件存在，则该命令不执行

free\_form：要执行的linux指令

chdir：在执行指令之前，先切换到该目录

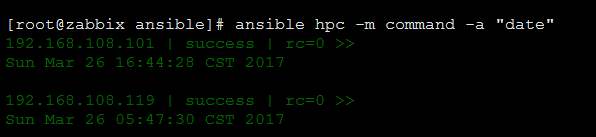
removes：一个文件名，当该文件不存在，则该选项不执行

executable：切换she

ll来执行指令，该执行路径必须是一个绝对路径

示例：

# ansible hpc -m command -a "date"



#### **(6)、shell**

## 切换到某个shell执行指定的指令，参数与command相同。

与command不同的是，此模块可以支持命令管道，同时还有另一个模块也具备此功能：raw

示例：

## 先在本地创建一个SHELL脚本

# vim /tmp/guxing.sh

#!/bin/sh

date +%F\_%H:%M:%S

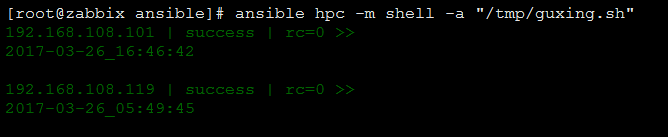
#chmod +x /tmp/guxing.sh

## 将创建的脚本文件分发到远程

# ansible hpc -m copy -a "src=/tmp/guxing.sh dest=/tmp/guxing.sh owner=root group=root mode=0755"

## 远程执行

# ansible hpc -m shell -a "/tmp/guxing.sh"



#### **(7)、更多模块**

其他常用模块，比如：service、cron、yum、synchronize就不一一例举，可以结合自身的系统环境进行测试。

service：系统服务管理

cron：计划任务管理

yum：yum软件包安装管理

synchronize：使用rsync同步文件

user：系统用户管理

group：系统用户组管理

ad-hoc常用命令是用来解决一些常用的简单的配置命令，而且这些命令的执行速度很快。adhoc可以不用写playbooks，使用方便简单。

man ansible

ansible <pattern> [-f forks] [-m module] [-a args]

ARGUMENTS

pattern

组名，或者主机名，匹配hosts文件。

OPTIONS

-v --verbose

详细模式，如果命令执行成功，输出详细的结果，

-i PATH --inventory=PATH

指定host文件的路径，默认是在/etc/ansible/hosts

-f NUM --forks=NUM

NUM是指定一个整数，默认是5，指定fork开启同步进程的个数。

--private-key=PRIVATE\_KEY\_FILE\_PATH

使用指定的文件建立认证连接

-m NAME --module-name=NAME

指定使用的module名称

-m DIRECTORY --module-path=DIRECTORY

指定module的目录来加载module，默认是/usr/share/ansible,

-a ARGUMENTS

指定module模块的参数

-k --ask-pass

提示输入ssh的密码，而不是使用基于ssh的密钥认证

-K --ask-sudo-pass

提示输入sudo密码，与--sudo一起使用

-o --one-line

在一行内输出所以的内容

-s --sudo

-u指定用户的时候，使用sudo获得root权限

-t DIRECTORY --tree=DIRECTORY

将输出放在指定的目录下，命名为每一个主机的名称

-T SECONDS --timeout=SECONDS

超时时长

-B NUM --background=NUM

在后台运行命令，在指定NUM秒后kill该任务

-P NUM --poll=NUM

每隔NUM秒，poll一个后台任务，跟-B一起使用

-u USERNAME --user=USERNAME

指定移动端的执行用户

-U SUDO\_USERNAME --sudo-user=USERNAME

sudo到SUDO\_USERNAME，替代root用户

-c CONNECTION --connection=CONNECTION

指定建立连接的类型，一般有ssh ，localhost

FILES

/etc/ansible/hosts 默认的host文件路径

/usr/share/ansible 默认模块文件存放路径

/etc/ansible/ansible.cfg 配置文件

File Transfer

ansible通过scp可以将文件传输到远程主机上

#ansible webservers -m copy -a "src=/etc/hosts dest=/tmp/hosts"

将本机的/etc/hosts文件拷贝到远程主机组webservers中所有主机的 /tmp/hosts下。

file模块可以改变文件的权限和所属用户组

#ansible webservers -m file -a "dest=/usr/local/src/test mode=600 owner=www group=www"

file模块还可以创建目录，类似mkdir -p

#ansible webservers -m file -a "dest=/a/b/c/d mode=755 owner=www group=www state=new"

同样也可以删除文件

#ansible webserver -m file -a "dest=/tmp/hosts state=absent"

Managing Packages

ansible同样提供了yum模块来安装软件包

确保一个包已经安装，但是不进行update操作

#ansible webservers -m yum -a "name=acme state=installed"

查看已经安装了指定版本的软件包

#ansible webservers -m yum -a "name=acme-1.5 state=installed"

查看安装软件包是否为最新版本

#ansible webservers -m yum -a "name=acme state=latest"

确保一个软件包是没有安装的

#ansible webservers -m yum -a “name=acme state=removed"

ansible在platform上提供了模块用于包管理，如果你要安装的软件包在模块中不可用，你可以通过commond模块来安装包。

Users & Groups

'user'模块可以方便的创建或者操作一个用户帐号，同样的也可以删除一个用户

#ansible all -m user -a "name=test password=<abc>"

#ansible all -m user -a "name=test state=absent"

Managing Services

确定服务都是开启的

#ansible all -m service -a "name=httpd state=started"

重启服务

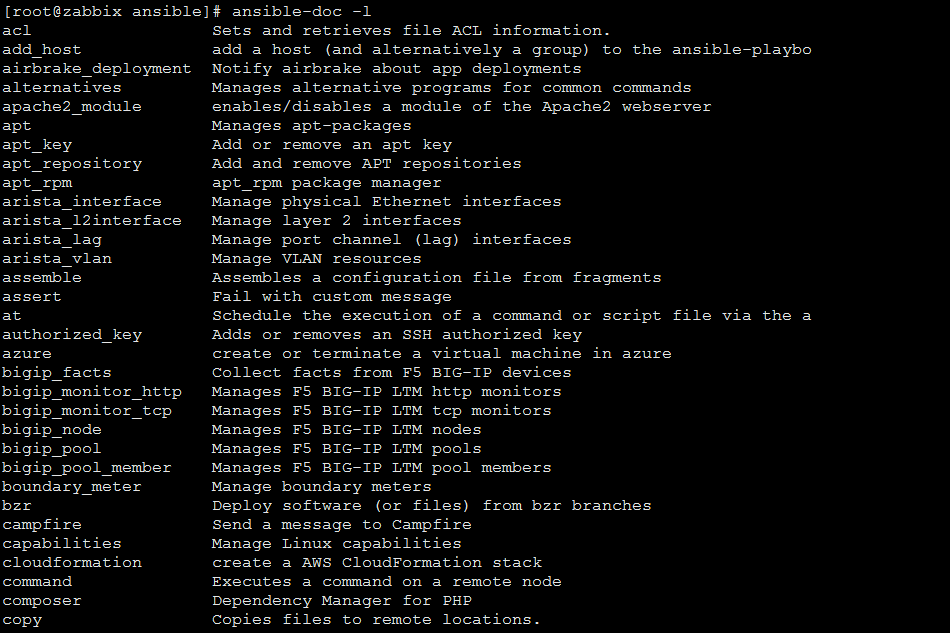
#ansibel all -m service -a "name=httpd state=restarted"

关闭服务

#ansible all -m service -a "name=httpd state=stoped"

更多模块可以参考：

#ansible-doc –l



<http://docs.ansible.com/modules_by_category.html>

<http://www.ansible.cn/docs/>

(国内的一个镜像站点，避免被墙 ^\_^ )

#### **(8)、一些概念补充**

**playbook的组成**：playbook是由一个或多个“play”组成的列表，可以让它们联同起来按事先编排的机制执行；所谓task无非是调用ansible的一个module，而在模块参数中可以使用变量；模块执行是幂等的，这意味着多次执行是安全的，因为其结果均一致；

**执行模型**：task list中的各任务按次序逐个在hosts中指定的所有主机上执行，即在所有主机上完成第一个任务后再开始第二个。在顺序运行某playbook时，如果中途发生错误，所有已执行任务都将回滚，因此，在修改playbook后重新执行一次即可；

**task组成**：每个task都应该有其name，用于playbook的执行结果输出，建议其内容尽可能清晰地描述任务执行步骤。如果未提供name，则action的结果将用于输出；

**notify指定handler的执行机制**：“notify”这个action可用于在每个play的最后被触发，在notify中列出的操作称为handler，仅在所有的变化发生完成后一次性地执行指定操作。

==========================================================================================

## **三、后续工作**

==========================================================================================

1、深入学习ansible的playbook以及扩展模块；

2、 结合业务环境，初步实现基础监控，以取代目前调用自动化部署平台API的方式；

3、 尝试自动化运维工具saltstack，并将其与ansible进行对比。

一些学习资料：

<http://blog.xiaorui.cc/category/ansible/>

<http://lixcto.blog.51cto.com/4834175/d-4>

<https://github.com/ansible/ansible-examples>

<http://rfyiamcool.blog.51cto.com/1030776/d-51>

<http://dl528888.blog.51cto.com/2382721/d-4/p-1>

<http://edu.51cto.com/course/course_id-2220.html>

<http://edu.51cto.com/course/course_id-2032.html>

[http://www.shencan.net/index.php/category/%e8%87%aa%e5%8a%a8%e5%8c%96%e8%bf%90%e7%bb%b4/ansible/](http://www.shencan.net/index.php/category/%E8%87%AA%E5%8A%A8%E5%8C%96%E8%BF%90%E7%BB%B4/ansible/)