

自动化运维工具Ansible详细部署 - 人生理想在于坚持不懈

原创

2014-11-20 00:48:40

标签: [ansible](#)

原创作品，允许转载，转载时请务必以超链接形式标明文章 [原始出处](#)、作者信息和本声明。否则将追究法律责任。
<http://sofar.blog.51cto.com/353572/1579894>

=====

=====一、基础介绍=====

=====

=====

1、简介

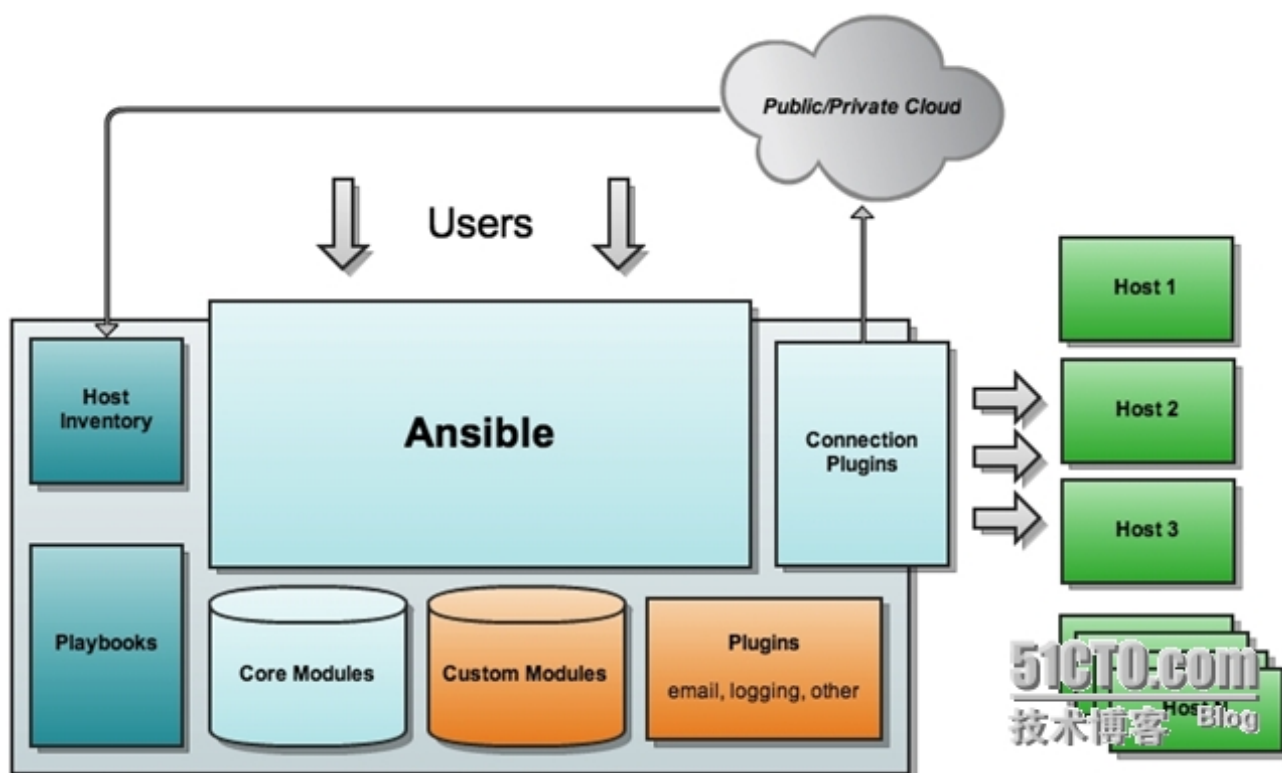
ansible是新出现的自动化运维工具，基于Python开发，集合了众多运维工具（puppet、cfengine、chef、func、fabric）的优点，实现了批量系统配置、批量程序部署、批量运行命令等功能。ansible是基于模块工作的，本身没有批量部署的能力。真正具有批量部署的是ansible所运行的模块，ansible只是提供一种框架。主要包括：

- (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



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

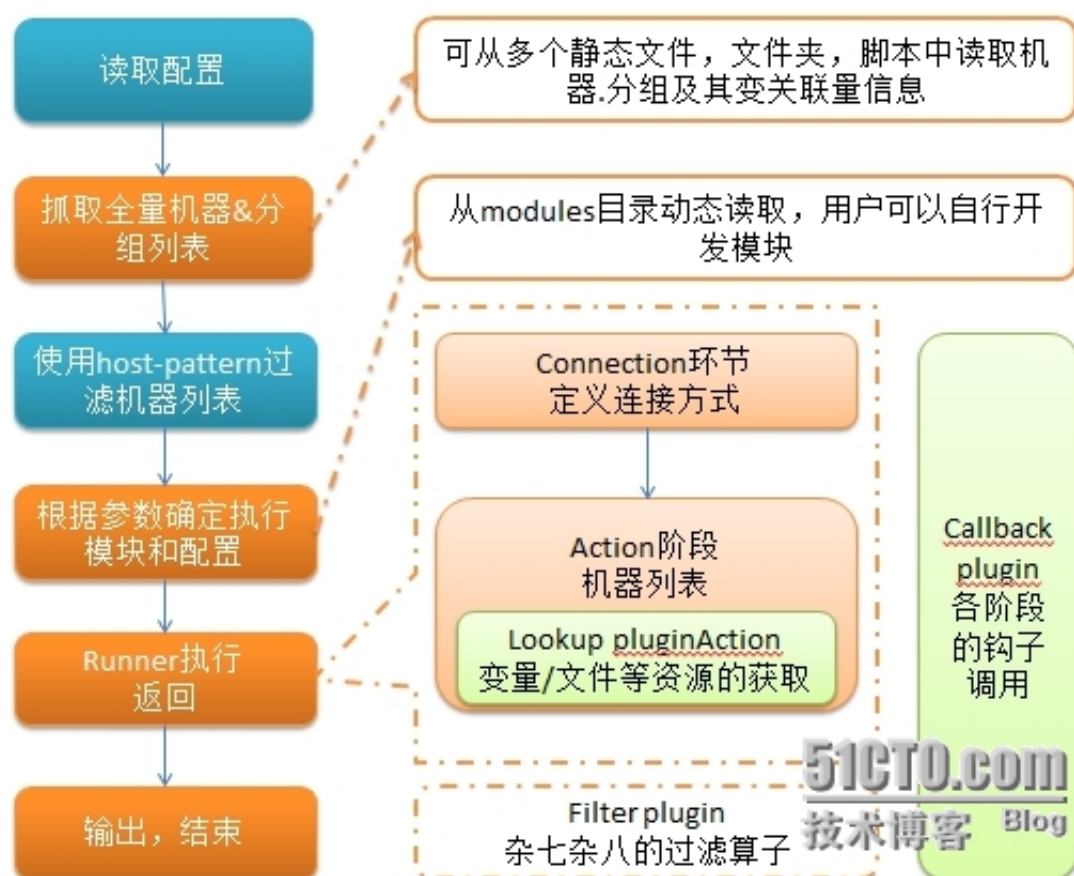
(1)、轻量级，无需在客户端安装agent，更新时，只需在操作机上进行一次更新即可；

(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



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 6.4 系统环境。

```
=====
=====
```

二、Ansible基础安装与配置

```
=====
=====
```

1、Ansible基础安装

(1)、python2.7安装

```
# tar xvf 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 python2.6# 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/python2.6
```

(2)、setuptools模块安装

```
# tar xvzf setuptools-7.0.tar.gz# cd setuptools-7.0# python  
setup.py install
```

(3)、pycrypto模块安装

```
# tar xvzf pycrypto-2.6.1.tar.gz# cd pycrypto-2.6.1# python  
setup.py install
```

(4)、PyYAML模块安装

```
# 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# tar xvzf PyYAML-3.11.tar.gz# cd  
PyYAML-3.11# python setup.py install
```

(5)、Jinja2模块安装

```
# tar xvzf MarkupSafe-0.9.3.tar.gz# cd MarkupSafe-0.9.3#  
python setup.py install# tar xvzf Jinja2-2.7.3.tar.gz # cd  
Jinja2-2.7.3# python setup.py install
```

(6)、paramiko模块安装

```
# tar xvzf ecdsa-0.11.tar.gz# cd ecdsa-0.11# python setup.py  
install# tar xvzf paramiko-1.15.1.tar.gz# cd paramiko-  
1.15.1# python setup.py install
```

(7)、simplejson模块安装

```
# tar xvzf simplejson-3.6.5.tar.gz# cd simplejson-3.6.5#  
python setup.py install
```

(8)、ansible安装

```
# tar xvzf ansible-1.7.2.tar.gz# cd ansible-1.7.2# python
```

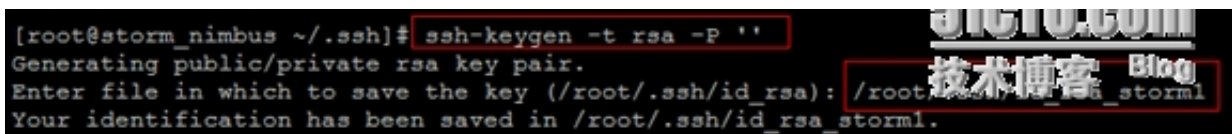
```
setup.py install
```

2、Ansible配置

(1)、SSH免密钥登录设置

生成公钥/私钥

```
# ssh-keygen -t rsa -P ''
```



```
[root@storm_nimbus ~/.ssh]# ssh-keygen -t rsa -P ''
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): /root/.ssh/id_rsa_storm1
Your identification has been saved in /root/.ssh/id_rsa_storm1.
```

写入信任文件（将/root/.ssh/id_rsa_storm1.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_storm1.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 =
```

```
36000private_key_file = /root/.ssh/id_rsa_storm1.....
```

主机组定义

```
# vim /etc/ansible/hosts
```

```
[storm_cluster]
```

```
10.223.55.10010.223.55.10110.223.38.22610.223.38.22710.223.3
```


```
9.21610.223.25.123
```

(3)、简单测试

```
# ansible storm_cluster -m command -a 'uptime'
```

```
[root@storm_nimbus /etc/ansible]# ansible storm_cluster -m command -a 'uptime'
```

paramiko: The authenticity of host '[10.223.55.100]:36000' can't be established.
The ssh-rsa key fingerprint is d2b1d47971f689403b1e2985815da
Are you sure you want to continue connecting (yes/no)?



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 storm_cluster -m command -a 'uptime'
```

```
[root@storm_nimbus /etc/ansible]# ansible storm_cluster -m command -a 'uptime'
```

| Host | Result | rc |
|---------------|---------|----|
| 10.223.55.101 | success | 0 |
| 10.223.55.100 | success | 0 |
| 10.223.38.226 | success | 0 |
| 10.223.38.227 | success | 0 |
| 10.223.25.123 | success | 0 |
| 10.223.39.216 | success | 0 |

20:04:19 up 31 days, 5:20, 2 users, load average: 5.83, 5.23, 5.71


20:04:19 up 31 days, 5:20, 2 users, load average: 5.09, 3.82, 3.45

20:04:19 up 31 days, 5:20, 1 user, load average: 2.41, 3.03, 2.97

20:04:19 up 31 days, 5:20, 1 user, load average: 6.04, 5.09, 4.19

20:04:19 up 31 days, 5:20, 1 user, load average: 2.35, 3.37, 3.61

20:04:19 up 31 days, 5:20, 1 user, load average: 2.68, 3.37, 3.61



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 storm_cluster -m setup
```

```
[root@storm_nimbus /etc/ansible]# ansible storm_cluster -m setup
10.223.38.226 | success >> {
  "ansible_facts": {
    "ansible_all_ipv4_addresses": [
      "10.223.38.226"
    ],
    "ansible_all_ipv6_addresses": [],
    "ansible_architecture": "x86_64",
    "ansible_bios_date": "01/23/2014",
    "ansible_bios_version": "1.00",
    "ansible_cmdline": {
      "console": "tty0",
      "crashkernel": "512M-2G:64M,2G-12G:128M",
      "i8042.noaux": true,
      "ro": true,
      "root": "/dev/sda1"
    }
  }
}
```

51CTO.com
技术博客 Blog

(2)、ping

用来测试远程主机的运行状态

ansible storm_cluster -m ping

```
[root@storm_nimbus /etc/ansible]# ansible storm_cluster -m ping
10.223.38.226 | success >> {
  "changed": false,
  "ping": "pong"
}

10.223.55.100 | success >> {
  "changed": false,
  "ping": "pong"
}

10.223.38.227 | success >> {
  "changed": false,
  "ping": "pong"
}

10.223.55.101 | success >> {
  "changed": false,
  "ping": "pong"
}

10.223.25.123 | success >> {
  "changed": false,
  "ping": "pong"
}

10.223.39.216 | success >> {
  "changed": false,
  "ping": "pong"
}
```

51CTO.com
技术博客 Blog

(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 storm_cluster -m file -a "src=/etc/resolv.conf dest=/tmp/resolv.conf state=link"
```


```
[root@storm_nimbus /etc/ansible]# ansible storm_cluster -m file -a "src=/etc/resolv.conf dest=/tmp/resolv.conf state=link"
10.223.55.100 | success >> [
  "changed": true,
  "dest": "/tmp/resolv.conf",
  "gid": 0,
  "group": "root",
  "mode": "0777",
  "owner": "root",
  "size": 16,
  "src": "/etc/resolv.conf",
  "state": "link",
  "uid": 0
]
10.223.55.101 | success >> [
  "changed": true,
  "dest": "/tmp/resolv.conf",
  "gid": 0,
  "group": "root",
  "mode": "0777",
  "owner": "root",
  "size": 16,
  "src": "/etc/resolv.conf"
]
```



远程文件信息查看

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


```
[root@storm_nimbus /etc/ansible]# ansible storm_cluster -m command -a "ls -al /tmp/resolv.conf"
10.223.38.226 | success | rc=0 >>
lrwxrwxrwx 1 root root 16 Nov 15 20:16 /tmp/resolv.conf -> /etc/resolv.conf
10.223.55.100 | success | rc=0 >>
lrwxrwxrwx 1 root root 16 Nov 15 20:16 /tmp/resolv.conf -> /etc/resolv.conf
10.223.38.227 | success | rc=0 >>
lrwxrwxrwx 1 root root 16 Nov 15 20:16 /tmp/resolv.conf -> /etc/resolv.conf
10.223.55.101 | success | rc=0 >>
lrwxrwxrwx 1 root root 16 Nov 15 20:16 /tmp/resolv.conf -> /etc/resolv.conf
10.223.25.123 | success | rc=0 >>
lrwxrwxrwx 1 root root 16 Nov 15 20:16 /tmp/resolv.conf -> /etc/resolv.conf
10.223.39.216 | success | rc=0 >>
lrwxrwxrwx 1 root root 16 Nov 15 20:16 /tmp/resolv.conf -> /etc/resolv.conf
```



远程文件符号链接删除

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


```
[root@storm_nimbus /etc/ansible]# ansible storm_cluster -m file -a "path=/tmp/resolv.conf state=absent"
10.223.38.227 | success >> {
  "changed": true,
  "path": "/tmp/resolv.conf",
  "state": "absent"
}
10.223.38.226 | success >> {
  "changed": true,
  "path": "/tmp/resolv.conf",
  "state": "absent"
}
10.223.55.100 | success >> {
  "changed": true,
  "path": "/tmp/resolv.conf",
  "state": "absent"
}
10.223.55.101 | success >> {
  "changed": true,
  "path": "/tmp/resolv.conf",
  "state": "absent"
}
```



远程文件信息查看

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

```
[root@storm_nimbus /etc/ansible]# ansible storm_cluster -m command -a "ls -al /tmp/resolv.conf"
10.223.38.226 | FAILED | rc=2 >>
ls: cannot access /tmp/resolv.conf: No such file or directory
10.223.55.100 | FAILED | rc=2 >>
ls: cannot access /tmp/resolv.conf: No such file or directory
10.223.55.101 | FAILED | rc=2 >>
ls: cannot access /tmp/resolv.conf: No such file or directory
10.223.38.227 | FAILED | rc=2 >>
ls: cannot access /tmp/resolv.conf: No such file or directory
10.223.39.216 | FAILED | rc=2 >>
ls: cannot access /tmp/resolv.conf: No such file or directory
10.223.25.123 | FAILED | rc=2 >>
ls: cannot access /tmp/resolv.conf: No such file or directory
```



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 storm_cluster -m copy -a
```

```
"src=/etc/ansible/ansible.cfg dest=/tmp/ansible.cfg
```

```
owner=root group=root mode=0644"
```

```
[root@storm nimbus /etc/ansible]# ansible storm_cluster -m copy -a "src=/etc/ansible/ansible.cfg dest=/tmp/ansible.cfg owner=root group=root mode=0644"
10.223.38.226 | success >> {
  "changed": true,
  "dest": "/tmp/ansible.cfg",
  "gid": 0,
  "group": "root",
  "md5sum": "32c09e4962725ac278c00236c1ef96d2",
  "mode": "0644",
  "owner": "root",
  "size": 7484,
  "src": "/root/.ansible/tmp/ansible-tmp-1416054674.97-233153745699583/source",
  "state": "file",
  "uid": 0
}
10.223.55.101 | success >> {
  "changed": true,
  "dest": "/tmp/ansible.cfg",
  "gid": 0,
  "group": "root",
  "md5sum": "32c09e4962725ac278c00236c1ef96d2",
  "mode": "0644",
  "owner": "root",
  "size": 7484,
  "src": "/root/.ansible/tmp/ansible-tmp-1416054674.96-142188170569555/source",
  "state": "file",
  "uid": 0
}
```

51CTO.com
技术博客 Blog

远程文件信息查看

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

```
[root@storm nimbus /etc/ansible]# ansible storm_cluster -m command -a "ls -al /tmp/ansible.cfg"
10.223.38.226 | success | rc=0 >>
-rw-r--r-- 1 root root 7484 Nov 15 20:31 /tmp/ansible.cfg

10.223.55.101 | success | rc=0 >>
-rw-r--r-- 1 root root 7484 Nov 15 20:31 /tmp/ansible.cfg

10.223.55.100 | success | rc=0 >>
-rw-r--r-- 1 root root 7484 Nov 15 20:31 /tmp/ansible.cfg

10.223.38.227 | success | rc=0 >>
-rw-r--r-- 1 root root 7484 Nov 15 20:31 /tmp/ansible.cfg

10.223.25.123 | success | rc=0 >>
-rw-r--r-- 1 root root 7484 Nov 15 20:31 /tmp/ansible.cfg

10.223.39.216 | success | rc=0 >>
-rw-r--r-- 1 root root 7484 Nov 15 20:31 /tmp/ansible.cfg
```

51CTO.com
技术博客 Blog

(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shell来执行指令，该执行路径必须是一个绝对路径

示例: # ansible storm_cluster -m command -a "uptime"

```
[root@storm_nimbus /etc/ansible]# ansible storm_cluster -m command -a "uptime"
10.223.38.226 | success | rc=0 >>
  20:37:38 up 31 days,  5:53,  1 user,  load average: 3.23, 3.34, 3.17
10.223.55.100 | success | rc=0 >>
  20:37:38 up 31 days,  5:53,  2 users, load average: 4.49, 4.29, 3.96
10.223.38.227 | success | rc=0 >>
  20:37:38 up 31 days,  5:53,  1 user,  load average: 3.83, 3.76, 3.65
10.223.55.101 | success | rc=0 >>
  20:37:38 up 31 days,  5:53,  2 users, load average: 4.62, 4.60, 4.77
10.223.39.216 | success | rc=0 >>
  20:37:39 up 31 days,  5:53,  1 user,  load average: 5.48, 5.48, 5.48
10.223.25.123 | success | rc=0 >>
  20:37:39 up 31 days,  5:53,  1 user,  load average: 4.97, 5.79, 5.05
```



(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/rocketzhang_test.sh#!/bin/shdate
```

```
+%F_%H:%M:%S#chmod +x /tmp/rocketzhang_test.sh
```

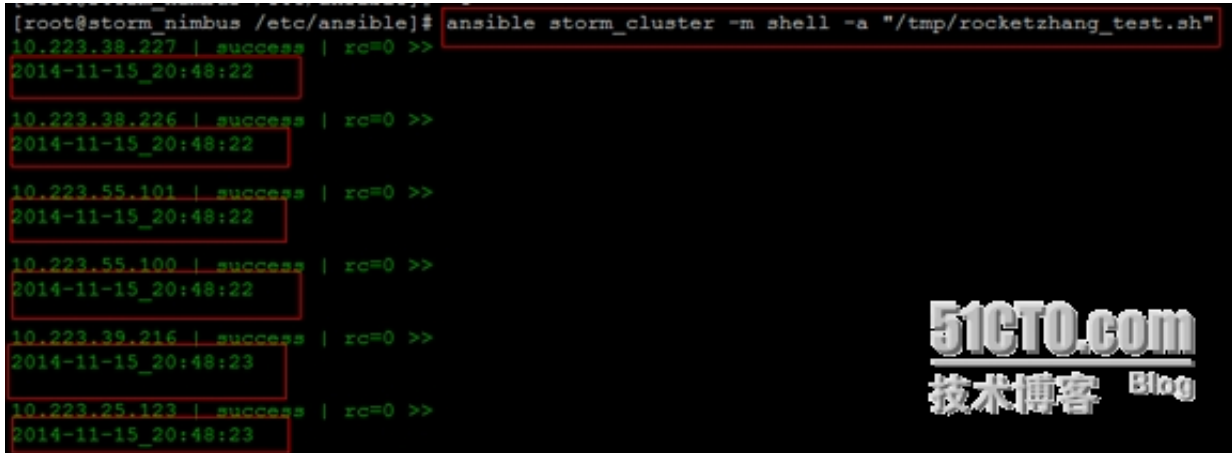
将创建的脚本文件分发到远程

```
# ansible storm_cluster -m copy -a
```

```
"src=/tmp/rocketzhang_test.sh dest=/tmp/rocketzhang_test.sh
owner=root group=root mode=0755"
```

远程执行

```
# ansible storm_cluster -m shell -a  
"/tmp/rocketzhang_test.sh"
```



A terminal window showing the execution of the Ansible command `ansible storm_cluster -m shell -a "/tmp/rocketzhang_test.sh"`. The output displays six successful executions on different hosts, each with a timestamp of 2014-11-15_20:48:22 or 2014-11-15_20:48:23. The hosts and their IP addresses are: 10.223.38.227, 10.223.38.226, 10.223.55.101, 10.223.55.100, 10.223.39.216, and 10.223.25.123. Each line is followed by `| success | rc=0 >>`. A watermark for **51CTO.com** 技术博客 Blog is visible in the bottom right corner of the terminal image.

(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

更多模块可以参考：#ansible-doc -l

```
[root@storm_nimbus /etc/ansible]# ansible-doc -l
acl                Sets and retrieves file ACL information.
add_host            add a host (and alternatively a group) to the ansible-playbo
airbrake_deployment Notify airbrake about app deployments
alternatives        Manages alternative programs for common commands
apache2_module      enables/disables a module of the Apache2 webserver
apt                 Manages apt-packages
apt_key             Add or remove an apt key
apt_repository      Add and remove APT repositories
apt_rpm             apt_rpm package manager
arista_interface    Manage physical Ethernet interfaces
arista_l2interface  Manage layer 2 interfaces
arista_lag          Manage port channel (lag) interfaces
arista_vlan         Manage VLAN resources
assemble            Assembles a configuration file from fragments
assert              Fail with custom message
at                  Schedule the execution of a command or script file via the a
authorized_key      Adds or removes an SSH authorized key
azure               create or terminate a virtual machine in azure
bigip_facts         Collect facts from F5 BIG-IP devices
bigip_monitor_http Manages F5 BIG-IP LTM http monitors
bigip_monitor_tcp   Manages F5 BIG-IP LTM tcp monitors
bigip_node          Manages F5 BIG-IP LTM nodes
bigip_pool          Manages F5 BIG-IP LTM pools
bigip_pool_member   Manages F5 BIG-IP LTM pool members
boundary_meter      Manage boundary meters
bzz                 Deploy software (or files) from bzz branches
camfire             Send a message to Camfire
```



(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：