

公告

个人博客站点: <http://jkzhao.github.io/>

github地址:

<https://github.com/jkzhao>

昵称: 暴走小骚年

园龄: 1年11个月

粉丝: 41

关注: 14

+加关注

< 2019年3月 >						
日	一	二	三	四	五	六
24	25	26	27	28	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

搜索

找找看

谷歌搜索

常用链接

我的随笔

我的评论

我的参与

最新评论

我的标签

我的标签

SpringMVC(9)

Mybatis(7)

Python(6)

Spring(6)

Servlet(3)

Docker(2)

Jenkins(1)

JSP(1)

Redis(1)

Swagger(1)

更多

随笔分类

Docker(10)

Hadoop(2)

JavaEE(14)

JavaSE(3)

JSP(1)

Kubernetes(6)

Linux(7)

Mybatis(7)

NoSQL(1)

Python(4)

Spring(6)

SpringMVC(8)

持续集成(1)

大数据(9)

监控(5)

随笔-80 文章-0 评论-43

Ansible常见模块介绍

本节内容:

- ansible命令基础
- 常见模块举例

一、ansible命令基础

语法:

```
ansible <host-pattern> [-f forks] [-m module_name] [-a args] [options]
```

- host-pattern: 这次的命令对哪些主机生效;
- -f forks: 启动的并发线程数, 就是一次并行处理多少主机;
- -m module_name: 要使用的模块;
- -a args: 模块特有的参数。

常见的模块:

- user
- yum
- copy
- cron
- command: 这是默认的模块, 表示在被管理主机上运行一个命令。对于command模块, -a不再是指定参数, 而是命令本身。
- shell

二、常见模块举例

1. /etc/ansible/hosts配置文件内容

```
This is the default ansible 'hosts' file.
#
# It should live in /etc/ansible/hosts
#
# - Comments begin with the '#' character
# - Blank lines are ignored
# - Groups of hosts are delimited by [header] elements
# - You can enter hostnames or ip addresses
# - A hostname/ip can be a member of multiple groups
[mysql]
172.16.7.152

[nginx]
172.16.7.151
172.16.7.152
172.16.7.153
```

2. command模块

command模块是默认的模块, 表示在被管理主机上运行一个命令。对于command模块, -a不再是指定参数, 而是命令本身。所以这个模块有个缺陷, 运行的命令中不能使用变量或者参数。

示例:

```
[root@node1 ~]# ansible nginx -m command -a "date"
```

日志采集(1)
实时处理(2)
搜索引擎(2)
项目管理工具(1)
消息队列(2)
自动化工具(9)

随笔档案

2018年4月 (5)
2018年3月 (9)
2018年2月 (7)
2018年1月 (12)
2017年12月 (8)
2017年11月 (10)
2017年10月 (10)
2017年9月 (4)
2017年8月 (3)
2017年7月 (6)
2017年6月 (1)
2017年5月 (4)
2017年4月 (1)

最新评论

- Re:Kafka介绍及安装部署
辛辛苦苦~码的这么多字
--好想雨的云
- Re:Zabbix的通知功能以及自定义脚本告警
@旭旭会发光Python和Shell写起来更方便, 毕竟这里直接写一个脚本就可以了...
--暴走小骚年
- Re:Zabbix的通知功能以及自定义脚本告警
@旭旭会发光应该是可以的, 你可以参照着看看! ...
--暴走小骚年
- Re:Zabbix的通知功能以及自定义脚本告警
Java能用这个吗
--旭旭会发光
- Re:Kafka介绍及安装部署
@学语者不客气, 互相学习! ...
--暴走小骚年

阅读排行

- Kafka介绍及安装部署(28629)
- Jenkins+Docker持续集成(19700)
- Storm介绍及安装部署(13810)
- SpringMVC JSON数据交互(9182)
- Zookeeper介绍及安装部署(8851)

评论排行

- Kafka介绍及安装部署(16)
- Zabbix监控websphere和weblogic(8)
- Ansible实战: 部署分布式日志系统(2)
- Jenkins+Docker持续集成(6)
- Zabbix的通知功能以及自定义脚本告警(3)

推荐排行

- Kafka介绍及安装部署(11)
- Ansible实战: 部署分布式日志系统(3)
- Jenkins+Docker持续集成(2)
- Storm介绍及安装部署(2)

```
[root@node1 ~]# ansible nginx -m command -a "date"
172.16.7.153 | SUCCESS | rc=0 >>
Fri Jul 7 09:21:58 CST 2017
```

```
172.16.7.151 | SUCCESS | rc=0 >>
Thu Jul 6 21:22:41 EDT 2017
```

```
172.16.7.152 | SUCCESS | rc=0 >>
Thu Jul 6 21:23:13 EDT 2017
```

```
[root@node1 ~]# ansible mysql -m command -a "tail -3 /etc/passwd"
```

```
[root@node1 ~]# ansible mysql -m command -a "tail -3 /etc/passwd"
172.16.7.152 | SUCCESS | rc=0 >>
zabbix:x:1000:1000:./home/zabbix:/bin/bash
hadoop:x:1001:1001:./home/hadoop:/bin/bash
es:x:1002:1002:./home/es:/bin/bash
```

3. cron模块

cron模块可以让每一个被管理节点能够自动生成一个定期任务计划。查看cron模块的用法:

```
[root@node1 ~]# ansible-doc -s cron
```

```
[root@node1 ~]# ansible-doc -s cron
- name: Manage cron.d and crontab entries.
  action: cron
    backup                # If set, create a backup of the crontab before it is modified. The location of the backup
                           # 'backup_file' variable by this module.
    cron_file              # If specified, uses this file instead of an individual user's crontab. If this is a relative
                           # path, it is interpreted with respect to /etc/cron.d. (If it is absolute, it will typically
                           # be /etc/crontab). To use the 'cron_file' parameter you must specify the 'user'
                           # as well.
    day                    # Day of the month the job should run ( 1-31, *, */2, etc )
    disabled               # If the job should be disabled (commented out) in the crontab. Only has effect if state
                           # is 'present'.
    env                    # If set, manages a crontab's environment variable. New variables are added on top of crontab
                           # environment variables.
    hour                   # Hour when the job should run ( 0-23, *, */2, etc )
    insertafter             # Used with 'state=present' and 'env'. If specified, the environment variable will be inserted
                           # after the declaration of specified environment variable.
    insertbefore            # Used with 'state=present' and 'env'. If specified, the environment variable will be inserted
                           # before the declaration of specified environment variable.
    job                    # The command to execute or, if env is set, the value of environment variable. Required.
    minute                 # Minute when the job should run ( 0-59, *, */2, etc )
    month                  # Month of the year the job should run ( 1-12, *, */2, etc )
    name                   # Description of a crontab entry or, if env is set, the name of environment variable. Required.
                           # Note that if name is not set and state=present, then a new crontab entry will
                           # always be created, regardless of existing ones.
    reboot                 # If the job should be run at reboot. This option is deprecated. Users should use special_time
    special_time            # Special time specification nickname.
    state                  # Whether to ensure the job or environment variable is present or absent.
    user                   # The specific user whose crontab should be modified.
    weekday                # Day of the week that the job should run ( 0-6 for Sunday-Saturday, *, etc )
```

示例:

```
[root@node1 ~]# ansible mysql -m cron -a 'minute="*/10" job="/usr/bin/echo hello" name="test cron job"
```

```
[root@node1 ~]# ansible mysql -m cron -a 'minute="*/10" job="/usr/bin/echo hello" name="test cron job"
172.16.7.152 | SUCCESS => {
  "changed": true,
  "envs": [],
  "jobs": [
    "clear zk logs",
    "clear kafka logs",
    "clear nginx logs",
    "test cron job"
  ]
}
```

注意:

- 所有的参数可以用""包含起来
- day之类的参数没有指定, 默认都是*
- 默认state参数的值为present

```
[root@node1 ~]# ansible mysql -a 'crontab -l'
172.16.7.152 | SUCCESS | rc=0 >>
0-59/10 * * * * /usr/sbin/ntpdate us.pool.ntp.org | logger -t NTP
#Ansible: clear zk logs
0 0 * * 0 /usr/local/zookeeper-3.4.6/clean_zklog.sh
#Ansible: clear kafka logs
0 0 * * 0 /usr/local/kafka_2.11-0.9.0.1/clean_kafkalog.sh
#Ansible: clear nginx logs
0 0 * * 0 /usr/local/openresty/ctlrnginxlog.sh
#Ansible: test cron job
*/10 * * * * /usr/bin/echo hello
```

查看同步任务是否

4. user模块

user模块实现用户账号管理。查看user模块的用法：

```
[root@node1 ~]# ansible-doc -s user
```

几个主要参数：

- name=：用户名
- uid：用户的uid
- group：所属组，即私有组
- groups：附加组。
- state：状态。

示例：

```
[root@node1 ~]# ansible mysql -m user -a 'name="jack"'
```

```
[root@node1 ~]# ansible mysql -m user -a 'name="jack"'
172.16.7.152 | SUCCESS => {
  "changed": true,
  "comment": "",
  "createhome": true,
  "group": 1003,
  "home": "/home/jack",
  "name": "jack",
  "shell": "/bin/bash",
  "state": "present",
  "system": false,
  "uid": 1003
}
```

在mysql组中的主机上创建一个

```
[root@node1 ~]# ansible mysql -m user -a 'name="jack" state=absent'
```

```
[root@node1 ~]# ansible mysql -m user -a 'name="jack" state=absent'
172.16.7.152 | SUCCESS => {
  "changed": true,
  "force": false,
  "name": "jack",
  "remove": false,
  "state": "absent"
}
```

删除jack用户

5. group模块

group模块：组管理。查看group模块的用法：

示例：

```
[root@node1 ~]# ansible-doc -s group
```

```
[root@node1 ~]# ansible-doc -s group
- name: Add or remove groups
  action: group
    gid                # Optional `GID' to set for the group.
    name=              # Name of the group to manage.
    state              # Whether the group should be present or not on t
    system             # If `yes', indicates that the group created is a
```

```
[root@node1 ~]# ansible mysql -m group -a 'name=mysql gid=306 system=yes'
```

```
[root@node1 ~]# ansible mysql -m group -a 'name=mysql gid=306 s
172.16.7.152 | SUCCESS => {
  "changed": true,
  "gid": 306,
  "name": "mysql",
  "state": "present",
  "system": true
}
```

创建一个gid=306的系统组

6. copy模块

copy模块实现文件复制。查看copy模块的用法：

```
[root@node1 ~]# ansible-doc -s copy
```

几个主要参数：

- src=：指明源文件本地路径。可以是绝对路径，也可以是相对路径。可以不使用src，使用content。就是说用content内容来生成文件。
- dest=：定义远程目标文件路径，只能使用绝对路径。
- content=：可以不使用src，使用content。就是说用content内容来生成文件。

示例：

```
[root@node1 ~]# ansible mysql -m copy -a 'src=/etc/fstab dest=/tmp/fstab.ansible owner=root
mode=640'
```

```
[root@node1 ~]# ansible mysql -m copy -a 'src=/etc/fstab dest=/tmp/fstab.ansible own
172.16.7.152 | SUCCESS => {
  "changed": true,
  "checksum": "eefa7ce54d84769fe85c54e55bc3d0cce6eba64a",
  "dest": "/tmp/fstab.ansible",
  "gid": 0,
  "group": "root",
  "md5sum": "a8543f2e209b7f478219ac6027918aed",
  "mode": "0640",
  "owner": "root",
  "secontext": "unconfined_u:object_r:admin_home_t:s0",
  "size": 465,
  "src": "/root/.ansible/tmp/ansible-tmp-1499392621.45-77589248160726/source",
  "state": "file",
  "uid": 0
}
```

```
[root@node1 ~]# ansible mysql -m copy -a 'content="Hello World\nGood boy" dest=/tmp/test.txt
owner=root mode=640'
```

```
[root@node1 ~]# ansible mysql -m copy -a 'content="Hello World\nGood boy" dest=/tmp/test.txt c
172.16.7.152 | SUCCESS => {
  "changed": true,
  "checksum": "13fdb334c5c881148af89de3e4320d8991a5b628",
  "dest": "/tmp/test.txt",
  "gid": 0,
  "group": "root",
  "md5sum": "9971628d716ebae83754c905a09f6cab",
  "mode": "0640",
  "owner": "root",
  "secontext": "unconfined_u:object_r:admin_home_t:s0",
  "size": 20,
  "src": "/root/.ansible/tmp/ansible-tmp-1499392756.96-22687655509037/source",
  "state": "file",
  "uid": 0
}
```

content=取代src=

7. file模块

file模块可以设定文件属性，还可以创建文件的符号链接。查看file模块的用法：

```
[root@node1 ~]# ansible-doc -s file
```

几个重要参数：

- path=：指明对哪个文件做管理。也可以使用dest和name。
- 创建文件的符号链接：src=：指明源文件，path=：指明符号链接文件路径

示例：

```
[root@node1 ~]# ansible mysql -m file -a 'owner=root group=mysql mode=644 path=/tmp/test.txt'
```

```
[root@node1 ~]# ansible mysql -m file -a 'owner=root group=mysql mode=644 path=/tmp/test.txt'
172.16.7.152 | SUCCESS => {
  "changed": true,
  "gid": 306,
  "group": "mysql",
  "mode": "0644",
  "owner": "root",
  "path": "/tmp/test.txt",
  "secontext": "unconfined_u:object_r:admin_home_t:s0",
  "size": 20,
  "state": "file",
  "uid": 0
}
```

文件必3

```
[root@node1 ~]# ansible mysql -m file -a 'path=/tmp/test.link src=/tmp/test.txt state=link'
```

```
[root@node1 ~]# ansible mysql -m file -a 'path=/tmp/test.link src=/tmp/test.txt state=link'
172.16.7.152 | SUCCESS => {
  "changed": true,
  "dest": "/tmp/test.link",
  "gid": 0,
  "group": "root",
  "mode": "0777",
  "owner": "root",
  "secontext": "unconfined_u:object_r:user_tmp_t:s0",
  "size": 13,
  "src": "/tmp/test.txt",
  "state": "link",
  "uid": 0
}
```

创建一个符号链接指向某文件

8. ping模块

ping模块测试指定主机是否能连接。查看ping模块的用法：

```
[root@node1 ~]# ansible-doc -s ping
```

示例：

```
[root@node1 ~]# ansible nginx -m ping
```

```
[root@node1 ~]# ansible nginx -m ping
172.16.7.153 | SUCCESS => {
  "changed": false,
  "ping": "pong"
}
172.16.7.151 | SUCCESS => {
  "changed": false,
  "ping": "pong"
}
172.16.7.152 | SUCCESS => {
  "changed": false,
  "ping": "pong"
}
```

9. service模块

service模块是管理服务的。查看service模块的用法：

```
[root@node1 ~]# ansible-doc -s service
```

```
[root@node1 ~]# ansible-doc -s service
- name: Manage services.
  action: service
    arguments      # Additional arguments provided on the command line
    enabled        # Whether the service should start on boot. *At least one of state and enabled are requ
    name=          # Name of the service.
    pattern        # If the service does not respond to the status command, name a substring to look for as
                  # output of the 'ps' command as a stand-in for a status result. If the string is
                  # found, the service will be assumed to be running.
    runlevel       # For OpenRC init scripts (ex: Gentoo) only. The runlevel that this service belongs to.
    sleep          # If the service is being 'restarted' then sleep this many seconds between the stop and
                  # to workaround badly behaving init scripts that exit immediately after signaling
                  # a process to stop.
    state          # 'started'/'stopped' are idempotent actions that will not run commands unless necessary
                  # always bounce the service. 'reloaded' will always reload. *At least one of
                  # state and enabled are required.*
```

几个重要参数：

- enabled=: 是否开机自动启动，取值为true或false；
- name=: 服务名字；
- state=: 状态，取值有started, stopped, restarted。

示例：

```
[root@node1 ~]# ansible 172.16.7.151 -m service -a 'enabled=true name=httpd state=stopped'
```

```
[root@node1 ~]# ansible 172.16.7.151 -m service -a 'enabled=true name=httpd state=stopped'
172.16.7.151 | SUCCESS => {
  "changed": true,
  "enabled": true,
  "name": "httpd",
  "state": "stopped",
  "status": {
    "ActiveEnterTimestamp": "Fri 2016-05-27 02:01:02 EDT",
    "ActiveEnterTimestampMonotonic": "2746888160338",
    "ActiveExitTimestamp": "Fri 2016-05-27 02:01:01 EDT",
    "ActiveExitTimestampMonotonic": "2746886855355",
    "ActiveState": "active",
    "After": "tmp.mount basic.target -.mount system.slice systemd-journald.socket remote-fs.target nss-lookup
    "AllowIsolate": "no",
    "AssertResult": "yes",
    "AssertTimestamp": "Fri 2016-05-27 02:01:02 EDT",
    "AssertTimestampMonotonic": "2746887947171",
    "Before": "shutdown.target multi-user.target",
    "BlockIOAccounting": "no",
    "BlockIOWeight": "18446744073709551615",
```

原先172.16.7.151机器上的httpd是开启的，现在把它关闭，并设置

10. shell模块

和command模块类似，但是可以使用变量。用于执行一些复杂的命令。查看shell模块的使用方法：

```
[root@node1 ~]# ansible-doc -s shell
```

示例：

```
[root@node1 ~]# ansible mysql -m user -a 'name="test"'
[root@node1 ~]# ansible mysql -m command -a 'echo wisedu | passwd --stdin test'
[root@node1 ~]# ansible mysql -m command -a 'tail -1 /etc/passwd'
```

```
[root@node1 ~]# ansible mysql -m user -a 'name="user1"'
172.16.7.152 | SUCCESS => {
  "changed": true,
  "comment": "",
  "createhome": true,
  "group": 1005,
  "home": "/home/user1",
  "name": "user1",
  "shell": "/bin/bash",
  "state": "present",
  "stderr": "useradd: warning: the home directory already exists.\nNot copyin
  File exists\n",
  "system": false,
  "uid": 1005
}
[root@node1 ~]# ansible mysql -m command -a 'echo wisedu | passwd --stdin user1'
172.16.7.152 | SUCCESS | rc=0 >> 使用command模块给用户1设置密码
wisedu | passwd --stdin user1

[root@node1 ~]# ansible mysql -m command -a 'tail -1 /etc/shadow'
172.16.7.152 | SUCCESS | rc=0 >> 你会发现密码没有设置成功
user1:O:17354:0:99999:7:::
```

```
[root@node1 ~]# ansible mysql -m shell -a 'echo wisedu | passwd --stdin user1'
[root@node1 ~]# ansible mysql -m command -a 'tail -1 /etc/shadow'
```

```
[root@node1 ~]# ansible mysql -m shell -a 'echo wisedu | passwd --stdin user1'
172.16.7.152 | SUCCESS | rc=0 >>
Changing password for user user1.
passwd: all authentication tokens updated successfully.

[root@node1 ~]# ansible mysql -m command -a 'tail -1 /etc/shadow'
172.16.7.152 | SUCCESS | rc=0 >>
user1:$6$37R3DXdD$YDv0kMFNfqJvwtDZPKbKONR9N3Pran6xQDwf/NNNN3SHN694MMuho.VuN2oYfgcELVTqQpnLkto0aT/qRK9
```

改用shell模块设置user1密码

设置成功

所以一旦有管道、变量之类的，你最好使用shell模块，而不要用command模块。

11. script模块

script模块将本地脚本复制到远程主机并运行之。查看script模块的用法：

```
[root@node1 ~]# ansible-doc -s script
```

示例：

```
[root@node1 ~]# vim test.sh
#!/bin/bash
echo "hello world" >/tmp/nba.txt
[root@node1 ~]# chmod +x test.sh
[root@node1 ~]# ansible mysql -m script -a '/root/test.sh'
```

```
[root@node1 ~]# ansible mysql -m script -a '/root/test.sh'
172.16.7.152 | SUCCESS => {
  "changed": true,
  "rc": 0,
  "stderr": "Shared connection to 172.16.7.152 closed.\r\n",
  "stdout": "",
  "stdout_lines": []
}
```

12. yum模块

yum模块管理程序包。查看yum模块的用法：

```
[root@node1 ~]# ansible-doc -s yum
```

几个重要参数：

- name=：指定要安装的程序包，可以带上版本号，否则安装最新版本；
- state=：present表示安装，absent表示卸载。

示例：

```
[root@node1 ~]# ansible mysql -m yum -a 'name=ksh'
```

```
[root@node1 ~]# ansible mysql -m yum -a 'name=ksh'
172.16.7.152 | SUCCESS => {
  "changed": true,
  "msg": "",
  "rc": 0,
  "results": [
    "Loaded plugins: fastestmirror, langpacks\nLoading mirror speeds from cached hostfile\n * base: mirrors.cn99
r.rackspace.com\n * extras: mirrors.163.com\n * updates: centos.ustc.edu.cn\nResolving Dependencies\n--> Running tra
ge ksh.x86_64 0:20120801-26.el7 will be installed\n--> Finished Dependency Resolution\n\nDependencies Resolved\n\n=====
\nPackage           Arch           Version           Rep
\n\nInstalling:\n ksh                               x86_64
\n\nTransaction Summary\n\n
Package\n\nTotal download size: 883 k\nInstalled size: 3.1 M\nDownloading packages:\nRunning transaction check\nRun
nsaction test succeeded\nRunning transaction\n  Installing : ksh-20120801-26.el7.x86_64
ksh-20120801-26.el7.x86_64                               1/1 \n\nInstalled:\n  ksh.x86_64 0:20120801-26.el7
\n\nComplete!\n"
  ]
}
```

13. setup模块

setup模块：收集远程主机的facts。ansible在管理每一个主机时，这些主机在被运行管理命令之前，会首先向ansible节点报告自己主机当前的各种可能被ansible主机用到的状态信息，如操作系统版本、ip地址等信息，这些信息都是以变量的形式，ansible主机可以在jinja2中调用，为不同的服务器生成不同的配置文件。

```
[root@node1 ~]# ansible mysql -m setup
```


```
172.16.7.152 | SUCCESS => {
  "ansible_facts": {
    "ansible_all_ipv4_addresses": [
      "172.16.7.152",
      "172.16.2.225",
      "172.17.0.1",
      "172.18.0.1"
    ],
    "ansible_all_ipv6_addresses": [
      "fe80::250:56ff:feb7:708f",
      "fe80::42:53ff:fee5:426d",
      "fe80::42:aeff:fe81:1df1"
    ],
    "ansible_architecture": "x86_64",
    "ansible_bios_date": "06/22/2012",
    "ansible_bios_version": "6.00",
    "ansible_cmdline": {
      "BOOT_IMAGE": "/vmlinuz-4.5.2-1.el7.elrepo.x86_64",
      "LANG": "en_US.UTF-8",
```

里面都是这种变量

分类: 自动化工具

好文要顶 关注我 收藏该文





暴走小骚年
关注 - 14
粉丝 - 41
[+加关注](#)

0

0

« 上一篇: [Ansible介绍及安装部署](#)
» 下一篇: [Ansible的基础元素和YAML介绍](#)

posted @ 2017-09-28 14:08 暴走小骚年 阅读(684) 评论(0) 编辑 收藏

[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论, 请 [登录](#) 或 [注册](#), [访问网站首页](#)。

- 【幸运】99%的人不知道我们有可以帮你薪资翻倍的秘笈!
- 【推荐】超50万C++/C#源码: 大型实时仿真组态图形源码
- 【推荐】百度云“猪”你开年行大运, 红包疯狂拿
- 【推荐】55K刚面完Java架构师岗, 这些技术你必须掌握

相关博文:

- [Ansible常用模块介绍](#)
- [ansible模块介绍](#)
- [ansible常用模块介绍](#)
- [Ansible小结 \(三\) ---常用模块介绍](#)
- [Ansible常用模块介绍](#)

最新新闻:

- [骨髓移植有望治愈血液病](#)
- [裁员潮背后, 互联网公司借机清洗“人才泡沫”](#)
- [都说要“补铁”, 你真的需要铁吗?](#)
- [时代抛弃了拉卡拉 连声再见都不说](#)

· 网易考拉、雅诗兰黛互相起诉 海淘“罗生门”有解吗
» 更多新闻...