Ansible部署

环境：rhel7.5虚拟机6台，关闭防火墙/selinux，配置好yum源

其中一台作为管理节点，两台web集群，两台db集群，一台cache

配置好hosts文件，并保证管理节点可以免密码登陆被管理节点

[root@ansible ~]# tail -6 /etc/hosts

192.168.4.1 ansible.tedu.cn ansible

192.168.4.11 web1.tedu.cn web1

192.168.4.12 web2.tedu.cn web2

192.168.4.21 db1.tedu.cn db1

192.168.4.22 db2.tedu.cn db2

192.168.4.33 cache.tedu.cn cache

#安装ansible

[root@ansible ~]# tar -xf ansible\_soft.tar.xz -C /mnt/

[root@ansible ~]# ls /mnt/

ansible\_soft

[root@ansible ~]# yum -y install createrepo

[root@ansible ~]# createrepo /mnt/ansible\_soft/

Spawning worker 0 with 6 pkgs

Workers Finished

Saving Primary metadata

Saving file lists metadata

Saving other metadata

Generating sqlite DBs

Sqlite DBs complete

[root@ansible ~]# vim /etc/yum.repos.d/ansible.repo

[root@ansible ~]# cat /etc/yum.repos.d/ansible.repo

[ansible]

name=ansible\_soft

baseurl=file:///mnt/ansible\_soft/

enabled=1

gpgcheck=0

[root@ansible ~]# yum repolist | tail -1

repolist: 5,105

[root@ansible ~]# yum -y install ansible

[root@ansible ~]# ansible --version

ansible 2.4.2.0

config file = /etc/ansible/ansible.cfg

configured module search path = [u'/root/.ansible/plugins/modules', u'/usr/share/ansible/plugins/modules']

ansible python module location = /usr/lib/python2.7/site-packages/ansible

executable location = /usr/bin/ansible

python version = 2.7.5 (default, Feb 20 2018, 09:19:12) [GCC 4.8.5 20150623 (Red Hat 4.8.5-28)]

#配置ansible

[root@ansible ~]# sed -n '14p' /etc/ansible/ansible.cfg

#inventory = /etc/ansible/hosts

[root@ansible ~]# vim /etc/ansible/hosts

[root@ansible ~]# cat /etc/ansible/hosts | grep -v ^$ | grep -v ^#

[web]

web1

web2

[db]

db[1:2]

[other]

cache

#测试配置文件

[root@ansible ~]# ansible web --list-host

hosts (2):

web1

web2

[root@ansible ~]# ansible db --list-host

hosts (2):

db1

db2

[root@ansible ~]# ansible other --list-host

hosts (1):

cache

#测试inventory参数，单台

[root@cache ~]# rm -rf /root/.ssh/authorized\_keys

[root@ansible ~]# ansible other -m ping

cache | UNREACHABLE! => {

"changed": false,

"msg": "Failed to connect to the host via ssh: Permission denied (publickey,password).\r\n",

"unreachable": true

}

[root@ansible ~]# vim /etc/ansible/hosts

[root@ansible ~]# tail -2 /etc/ansible/hosts

[other]

cache ansible\_ssh\_user="root" ansible\_ssh\_pass="1"

[root@ansible ~]# ansible other -m ping

cache | SUCCESS => {

"changed": false,

"ping": "pong"

}

#还原正常配置

[root@ansible ~]# ssh-copy-id root@cache

[root@ansible ~]# vim /etc/ansible/hosts

[root@ansible ~]# tail -2 /etc/ansible/hosts

[other]

cache

[root@ansible ~]# ansible other -m ping

cache | SUCCESS => {

"changed": false,

"ping": "pong"

}

#测试inventory参数，分组配置

[root@ansible ~]# ssh web1

Last login: Wed Jan 9 00:05:51 2019 from 192.168.4.1

[root@web1 ~]# rm -rf /root/.ssh/authorized\_keys

[root@web1 ~]# exit

登出

Connection to web1 closed.

[root@ansible ~]# ssh web2

Last login: Wed Jan 9 00:05:51 2019 from 192.168.4.1

[root@web2 ~]# rm -rf /root/.ssh/authorized\_keys

[root@web2 ~]# exit

登出

Connection to web2 closed.

[root@ansible ~]# ansible web -m ping

web2 | UNREACHABLE! => {

"changed": false,

"msg": "Failed to connect to the host via ssh: Permission denied (publickey,password).\r\n",

"unreachable": true

}

web1 | UNREACHABLE! => {

"changed": false,

"msg": "Failed to connect to the host via ssh: Permission denied (publickey,password).\r\n",

"unreachable": true

}

[root@ansible ~]# vim /etc/ansible/hosts

[root@ansible ~]# tail -12 /etc/ansible/hosts

[web]

web1

web2

[web:vars]

ansible\_ssh\_user="root"

ansible\_ssh\_pass="1"

[db]

db[1:2]

[other]

cache

[root@ansible ~]# ansible web -m ping

web1 | SUCCESS => {

"changed": false,

"ping": "pong"

}

web2 | SUCCESS => {

"changed": false,

"ping": "pong"

}

#ansible连接缓存位置/root/.ansible/cp/

#还原正常配置

[root@ansible ~]# ssh-copy-id root@web1

[root@ansible ~]# ssh-copy-id root@web2

[root@ansible ~]# vim /etc/ansible/hosts

[root@ansible ~]# tail -9 /etc/ansible/hosts

[web]

web1

web2

[db]

db[1:2]

[other]

cache

[root@ansible ~]# ansible web -m ping

web2 | SUCCESS => {

"changed": false,

"ping": "pong"

}

web1 | SUCCESS => {

"changed": false,

"ping": "pong"

}

#子组定义

[root@ansible ~]# vim /etc/ansible/hosts

[root@ansible ~]# tail -3 /etc/ansible/hosts

[app:children]

web

db

[root@ansible ~]# ansible app --list-host

hosts (4):

web1

web2

db1

db2

#自定义主机分组

[root@ansible ~]# mkdir /tmp/ooxx

[root@ansible ~]# cd /tmp/ooxx

[root@ansible ooxx]# pwd

/tmp/ooxx

[root@ansible ooxx]# vim ansible.cfg

[root@ansible ooxx]# cat ansible.cfg

[defaults]

inventory = myhosts

[root@ansible ooxx]# vim myhosts

[root@ansible ooxx]# cat myhosts

[app1]

web1

db1

[app2]

web2

db2

cache

[app:children]

app1

app2

[root@ansible ooxx]# ansible app1 --list-hosts

hosts (2):

web1

db1

[root@ansible ooxx]# ansible app2 --list-host

hosts (3):

web2

db2

cache

[root@ansible ooxx]# ansible app --list-host

hosts (5):

web1

db1

web2

db2

cache

[root@ansible ooxx]# cd

[root@ansible ~]# ansible app --list-host

hosts (4):

web1

web2

db1

db2

#注意不同路径下的区别

##ansible查找配置文件的顺序

-首先检测ANSIBLE\_CONFIG变量定义的配置文件

-其次检测当前目录下的./ansible.cfg文件

-再次检测当前用户家目录下的~/ansible.cfg文件

-最后检测/etc/ansible/ansible.cfg文件

#动态主机--以json格式输出主机信息

#shell脚本

[root@ansible ~]# cd /tmp/ooxx/

[root@ansible ooxx]# pwd

/tmp/ooxx

[root@ansible ooxx]# vim a.sh

[root@ansible ooxx]# cat a.sh

#!/bin/bash

echo '

{

"webs" : { "hosts" : [ "web1" , "web2" ] },

"dbs" : { "hosts" : [ "db1" , "db2" ] },

"other" : ["cache"]

}'

[root@ansible ooxx]# chmod +x a.sh

[root@ansible ooxx]# vim ansible.cfg

[root@ansible ooxx]# cat ansible.cfg

[defaults]

inventory = a.sh

[root@ansible ooxx]# ansible webs --list-host

hosts (2):

web1

web2

#python脚本

[root@ansible ooxx]# vim a.py

[root@ansible ooxx]# chmod +x a.py

[root@ansible ooxx]# ./a.py

{"webs": ["web1", "web2"], "dbs": ["db1", "db2"], "other": ["cache"]}

[root@ansible ooxx]# cat a.py

#!/usr/bin/python

import json

hostlist = {}

hostlist["webs"] = ["web1","web2"]

hostlist["dbs"] = ["db1","db2"]

hostlist["other"] = ["cache"]

print(json.dumps(hostlist))

[root@ansible ooxx]# vim ansible.cfg

[root@ansible ooxx]# cat ansible.cfg

[defaults]

inventory = a.py

[root@ansible ooxx]# ansible dbs --list-hosts

hosts (2):

db1

db2

#批量执行命令

#检测当前环境

[root@ansible ooxx]# cd

[root@ansible ~]# ansible all -m ping

web1 | SUCCESS => {

"changed": false,

"ping": "pong"

}

web2 | SUCCESS => {

"changed": false,

"ping": "pong"

}

cache | SUCCESS => {

"changed": false,

"ping": "pong"

}

db1 | SUCCESS => {

"changed": false,

"ping": "pong"

}

db2 | SUCCESS => {

"changed": false,

"ping": "pong"

}

##可以用-k交互式输入密码

[root@ansible ~]# ansible all -m command -a "uptime"

db1 | SUCCESS | rc=0 >>

02:36:14 up 5:35, 2 users, load average: 0.00, 0.01, 0.05

cache | SUCCESS | rc=0 >>

02:36:14 up 5:26, 2 users, load average: 0.00, 0.01, 0.05

web1 | SUCCESS | rc=0 >>

02:36:15 up 5:37, 2 users, load average: 0.00, 0.01, 0.05

db2 | SUCCESS | rc=0 >>

02:36:15 up 5:32, 2 users, load average: 0.00, 0.01, 0.05

web2 | SUCCESS | rc=0 >>

02:36:15 up 5:05, 2 users, load average: 0.00, 0.01, 0.05

##管理节点部署ssh密钥(批量部署ssh密钥，适合初始化环境或有新增节点时）

[root@cache ~]# rm -rf .ssh/

[root@ansible ~]# ls /root/.ssh/

authorized\_keys id\_rsa id\_rsa.pub known\_hosts

[root@ansible ~]# ansible all -m authorized\_key -a "user=root exclusive=true manage\_dir=true key='$(< /root/.ssh/id\_rsa.pub)'" -k

SSH password:

web1 | SUCCESS => {

"changed": false,

"comment": null,

"exclusive": true,

"key": "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDA0zPqzeW6cMMnhsz2ZY/nKD3w96XF09WycD6sDjNQ+s723wPRlKZsokiYOhET2Lga9dOUi2PsJ3IBSDj+HRNpYIWKTY70w/F1ZS0jFWBUeI03lkIHgdqcwd8JbeHBVnKNNW2tLY63JzzG+11Wwg/NAX0vO/8RXkzNd1TvSy4Iaisao9M6gOoo+K52JyLQFrXCQP+0aSj5gF6t1ZqUt3wi7qMiyeq60N+cBYNB6di5CRJM5+8qBycRQo+G1cAB1HchO5x6JqXJKYSQeTOcVgurx766LnUUVBKEOnuqEFb2BHRGjhitE9+S1kXwsX0q7zcFIvHI3POZHzCBNx04f3d1 root@ansible.tedu.cn",

"key\_options": null,

"keyfile": "/root/.ssh/authorized\_keys",

"manage\_dir": true,

"path": null,

"state": "present",

"unique": false,

"user": "root",

"validate\_certs": true

}

web2 | SUCCESS => {

"changed": false,

"comment": null,

"exclusive": true,

"key": "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDA0zPqzeW6cMMnhsz2ZY/nKD3w96XF09WycD6sDjNQ+s723wPRlKZsokiYOhET2Lga9dOUi2PsJ3IBSDj+HRNpYIWKTY70w/F1ZS0jFWBUeI03lkIHgdqcwd8JbeHBVnKNNW2tLY63JzzG+11Wwg/NAX0vO/8RXkzNd1TvSy4Iaisao9M6gOoo+K52JyLQFrXCQP+0aSj5gF6t1ZqUt3wi7qMiyeq60N+cBYNB6di5CRJM5+8qBycRQo+G1cAB1HchO5x6JqXJKYSQeTOcVgurx766LnUUVBKEOnuqEFb2BHRGjhitE9+S1kXwsX0q7zcFIvHI3POZHzCBNx04f3d1 root@ansible.tedu.cn",

"key\_options": null,

"keyfile": "/root/.ssh/authorized\_keys",

"manage\_dir": true,

"path": null,

"state": "present",

"unique": false,

"user": "root",

"validate\_certs": true

}

db1 | SUCCESS => {

"changed": true,

"comment": null,

"exclusive": true,

"key": "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDA0zPqzeW6cMMnhsz2ZY/nKD3w96XF09WycD6sDjNQ+s723wPRlKZsokiYOhET2Lga9dOUi2PsJ3IBSDj+HRNpYIWKTY70w/F1ZS0jFWBUeI03lkIHgdqcwd8JbeHBVnKNNW2tLY63JzzG+11Wwg/NAX0vO/8RXkzNd1TvSy4Iaisao9M6gOoo+K52JyLQFrXCQP+0aSj5gF6t1ZqUt3wi7qMiyeq60N+cBYNB6di5CRJM5+8qBycRQo+G1cAB1HchO5x6JqXJKYSQeTOcVgurx766LnUUVBKEOnuqEFb2BHRGjhitE9+S1kXwsX0q7zcFIvHI3POZHzCBNx04f3d1 root@ansible.tedu.cn",

"key\_options": null,

"keyfile": "/root/.ssh/authorized\_keys",

"manage\_dir": true,

"path": null,

"state": "present",

"unique": false,

"user": "root",

"validate\_certs": true

}

cache | SUCCESS => {

"changed": true,

"comment": null,

"exclusive": true,

"key": "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDA0zPqzeW6cMMnhsz2ZY/nKD3w96XF09WycD6sDjNQ+s723wPRlKZsokiYOhET2Lga9dOUi2PsJ3IBSDj+HRNpYIWKTY70w/F1ZS0jFWBUeI03lkIHgdqcwd8JbeHBVnKNNW2tLY63JzzG+11Wwg/NAX0vO/8RXkzNd1TvSy4Iaisao9M6gOoo+K52JyLQFrXCQP+0aSj5gF6t1ZqUt3wi7qMiyeq60N+cBYNB6di5CRJM5+8qBycRQo+G1cAB1HchO5x6JqXJKYSQeTOcVgurx766LnUUVBKEOnuqEFb2BHRGjhitE9+S1kXwsX0q7zcFIvHI3POZHzCBNx04f3d1 root@ansible.tedu.cn",

"key\_options": null,

"keyfile": "/root/.ssh/authorized\_keys",

"manage\_dir": true,

"path": null,

"state": "present",

"unique": false,

"user": "root",

"validate\_certs": true

}

db2 | SUCCESS => {

"changed": true,

"comment": null,

"exclusive": true,

"key": "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDA0zPqzeW6cMMnhsz2ZY/nKD3w96XF09WycD6sDjNQ+s723wPRlKZsokiYOhET2Lga9dOUi2PsJ3IBSDj+HRNpYIWKTY70w/F1ZS0jFWBUeI03lkIHgdqcwd8JbeHBVnKNNW2tLY63JzzG+11Wwg/NAX0vO/8RXkzNd1TvSy4Iaisao9M6gOoo+K52JyLQFrXCQP+0aSj5gF6t1ZqUt3wi7qMiyeq60N+cBYNB6di5CRJM5+8qBycRQo+G1cAB1HchO5x6JqXJKYSQeTOcVgurx766LnUUVBKEOnuqEFb2BHRGjhitE9+S1kXwsX0q7zcFIvHI3POZHzCBNx04f3d1 root@ansible.tedu.cn",

"key\_options": null,

"keyfile": "/root/.ssh/authorized\_keys",

"manage\_dir": true,

"path": null,

"state": "present",

"unique": false,

"user": "root",

"validate\_certs": true

}

[root@cache ~]# ls .ssh/

authorized\_keys

##ansible模块

-m调用模块，默认是command，建议保留

#ansible远程管理节点时，在sshd进程下不开bash进程，直接执行command，

├─sshd───sshd───bash───pstree

|

|-command

所以只能执行硬盘上有的命令，bash命令无法执行，如<>|&;tab等

#shell模块可以执行任意shell命令，因为shell模块通过/bin/sh执行命令

#给db组主机创建zhangsan用户，初始化密码123，登陆时强制修改密码

[root@ansible ~]# ansible db -m ping

db2 | SUCCESS => {

"changed": false,

"ping": "pong"

}

db1 | SUCCESS => {

"changed": false,

"ping": "pong"

}

[root@ansible ~]# ansible db -m shell -a 'useradd zhangsan'

db1 | SUCCESS | rc=0 >>

db2 | SUCCESS | rc=0 >>

[root@ansible ~]# ansible db -m shell -a 'echo 123 | passwd --stdin zhangsan ; chage -d 0 zhangsan'

db1 | SUCCESS | rc=0 >>

更改用户 zhangsan 的密码 。

passwd：所有的身份验证令牌已经成功更新。

db2 | SUCCESS | rc=0 >>

更改用户 zhangsan 的密码 。

passwd：所有的身份验证令牌已经成功更新。

#ansible要上天，script模块在远端执行本地脚本(不限于shell)

#在xxoo目录下对app1组创建lisi用户，前提是系统里没有zhangsan用户

[root@ansible ~]# cd /tmp/ooxx/

[root@ansible ooxx]# vim ansible.cfg

[root@ansible ooxx]# cat ansible.cfg

[defaults]

inventory = myhosts

[root@ansible ooxx]# ansible app1 -m ping

db1 | SUCCESS => {

"changed": false,

"ping": "pong"

}

web1 | SUCCESS => {

"changed": false,

"ping": "pong"

}

[root@ansible ooxx]# vim lisi.sh

[root@ansible ooxx]# cat lisi.sh

#!/bin/bash

id zhangsan

if [ $? -ne 0 ];then

useradd lisi

echo 123 | passwd --stdin lisi

fi

[root@ansible ooxx]# ansible app1 -m script -a "./lisi.sh"

db1 | SUCCESS => {

"changed": true,

"rc": 0,

"stderr": "Shared connection to db1 closed.\r\n",

"stdout": "uid=1000(zhangsan) gid=1000(zhangsan) 组=1000(zhangsan)\r\n",

"stdout\_lines": [

"uid=1000(zhangsan) gid=1000(zhangsan) 组=1000(zhangsan)"

]

}

web1 | SUCCESS => {

"changed": true,

"rc": 0,

"stderr": "Shared connection to web1 closed.\r\n",

"stdout": "id: zhangsan: no such user\r\n更改用户 lisi 的密码 。\r\npasswd：所有的身份验证令牌已经成功更新。\r\n",

"stdout\_lines": [

"id: zhangsan: no such user",

"更改用户 lisi 的密码 。",

"passwd：所有的身份验证令牌已经成功更新。"

]

}

[root@db1 ~]# id zhangsan

uid=1000(zhangsan) gid=1000(zhangsan) 组=1000(zhangsan)

[root@db1 ~]# id lisi

id: lisi: no such user

[root@web1 ~]# id zhangsan

id: zhangsan: no such user

[root@web1 ~]# id lisi

uid=1000(lisi) gid=1000(lisi) 组=1000(lisi)

#copy模块

[root@ansible ~]# cp /etc/resolv.conf ./dns.conf

[root@ansible ~]# vim dns.conf

[root@ansible ~]# cat dns.conf

# Generated by NetworkManager

nameserver 192.168.4.254

[root@ansible ~]# ansible web -m copy -a "src=dns.conf dest=/etc/resolv.conf"

web1 | SUCCESS => {

"changed": true,

"checksum": "e6fb9b2f96bbf00f1fc173d86394f898c5ab998d",

"dest": "/etc/resolv.conf",

"gid": 0,

"group": "root",

"md5sum": "7709083aee149f19191097bff12f685e",

"mode": "0644",

"owner": "root",

"size": 55,

"src": "/root/.ansible/tmp/ansible-tmp-1547084863.19-248618557488442/source",

"state": "file",

"uid": 0

}

web2 | SUCCESS => {

"changed": true,

"checksum": "e6fb9b2f96bbf00f1fc173d86394f898c5ab998d",

"dest": "/etc/resolv.conf",

"gid": 0,

"group": "root",

"md5sum": "7709083aee149f19191097bff12f685e",

"mode": "0644",

"owner": "root",

"size": 55,

"src": "/root/.ansible/tmp/ansible-tmp-1547084863.22-145238373098661/source",

"state": "file",

"uid": 0

}

[root@ansible ~]# ansible web -m shell -a "cat /etc/resolv.conf"

web2 | SUCCESS | rc=0 >>

# Generated by NetworkManager

nameserver 192.168.4.254

web1 | SUCCESS | rc=0 >>

# Generated by NetworkManager

nameserver 192.168.4.254

#copy模块适合同步配置文件，也就是比较小的文件，重点是内容一样的配置文件

如mysql的配置文件，就不适合用copy模块分发

#批量修改配置文件用lineinfile模块，line重写整行

[root@ansible ~]# ansible other -m shell -a "cat /etc/sysconfig/network-scripts/ifcfg-eth0 | grep -i onboot"

cache | SUCCESS | rc=0 >>

ONBOOT=yes

[root@ansible ~]# ansible other -m lineinfile -a 'path=/etc/sysconfig/network-scripts/ifcfg-eth0 regexp="^ONBOOT" line="ONBOOT=\"NO\""'

cache | SUCCESS => {

"backup": "",

"changed": true,

"msg": "line replaced"

}

[root@ansible ~]# ansible other -m shell -a "cat /etc/sysconfig/network-scripts/ifcfg-eth0 | grep -i onboot"

cache | SUCCESS | rc=0 >>

ONBOOT="NO"

#replace模块，替换匹配的位置，用法跟sed一样

[root@ansible ~]# ansible other -m shell -a "cat /etc/sysconfig/network-scripts/ifcfg-eth0 | grep -i onboot"

cache | SUCCESS | rc=0 >>

ONBOOT="NO"

[root@ansible ~]# ansible other -m replace -a 'path=/etc/sysconfig/network-scripts/ifcfg-eth0 regexp="^(ONBOOT=).\*" replace="\1\"yes\""'

cache | SUCCESS => {

"changed": true,

"msg": "1 replacements made"

}

[root@ansible ~]# ansible other -m shell -a "cat /etc/sysconfig/network-scripts/ifcfg-eth0 | grep -i onboot"

cache | SUCCESS | rc=0 >>

ONBOOT="yes"

##yum模块管理软件包

#卸载软件包

[root@ansible ~]# ansible other -m shell -a "rpm -qa | grep vsftpd"

[WARNING]: Consider using yum, dnf or zypper module rather than running rpm

cache | SUCCESS | rc=0 >>

vsftpd-3.0.2-22.el7.x86\_64

[root@ansible ~]# ansible other -m yum -a 'name="vsftpd" state=removed'

cache | SUCCESS => {

"changed": true,

"msg": "",

"rc": 0,

"results": [

"已加载插件：product-id, search-disabled-repos, subscription-manager\nThis system is not registered with an entitlement server. You can use subscription-manager to register.\n正在解决依赖关系\n--> 正在检查事务\n---> 软件包 vsftpd.x86\_64.0.3.0.2-22.el7 将被 删除\n--> 解决依赖关系完成\n\n依赖关系解决\n\n================================================================================\n Package 架构 版本 源 大小\n================================================================================\n正在删除:\n vsftpd x86\_64 3.0.2-22.el7 @development 348 k\n\n事务概要\n================================================================================\n移除 1 软件包\n\n安装大小：348 k\nDownloading packages:\nRunning transaction check\nRunning transaction test\nTransaction test succeeded\nRunning transaction\n 正在删除 : vsftpd-3.0.2-22.el7.x86\_64 1/1 \n 验证中 : vsftpd-3.0.2-22.el7.x86\_64 1/1 \n\n删除:\n vsftpd.x86\_64 0:3.0.2-22.el7 \n\n完毕！\n"

]

}

[root@ansible ~]# ansible other -m shell -a "rpm -qa | grep vsftpd"

[WARNING]: Consider using yum, dnf or zypper module rather than running rpm

cache | FAILED | rc=1 >>

non-zero return code

#安装软件包

[root@ansible ~]# ansible other -m yum -a 'name="vsftpd" state=installed'

cache | SUCCESS => {

"changed": true,

"msg": "",

"rc": 0,

"results": [

"Loaded plugins: product-id, search-disabled-repos, subscription-manager\nThis system is not registered with an entitlement server. You can use subscription-manager to register.\nResolving Dependencies\n--> Running transaction check\n---> Package vsftpd.x86\_64 0:3.0.2-22.el7 will be installed\n--> Finished Dependency Resolution\n\nDependencies Resolved\n\n================================================================================\n Package Arch Version Repository Size\n================================================================================\nInstalling:\n vsftpd x86\_64 3.0.2-22.el7 development 169 k\n\nTransaction Summary\n================================================================================\nInstall 1 Package\n\nTotal download size: 169 k\nInstalled size: 348 k\nDownloading packages:\nRunning transaction check\nRunning transaction test\nTransaction test succeeded\nRunning transaction\n Installing : vsftpd-3.0.2-22.el7.x86\_64 1/1 \n Verifying : vsftpd-3.0.2-22.el7.x86\_64 1/1 \n\nInstalled:\n vsftpd.x86\_64 0:3.0.2-22.el7 \n\nComplete!\n"

]

}

[root@ansible ~]# ansible other -m shell -a "rpm -qa | grep vsftpd"

[WARNING]: Consider using yum, dnf or zypper module rather than running rpm

cache | SUCCESS | rc=0 >>

vsftpd-3.0.2-22.el7.x86\_64

##service控制服务

[root@cache ~]# systemctl is-enabled vsftpd

enabled

[root@cache ~]# systemctl status vsftpd

● vsftpd.service - Vsftpd ftp daemon

Loaded: loaded (/usr/lib/systemd/system/vsftpd.service; enabled; vendor preset: disabled)

Active: active (running) since 三 2019-01-09 21:33:01 EST; 31s ago

[root@ansible ~]# ansible other -m service -a 'name=vsftpd state=stopped enabled=no'

cache | SUCCESS => {

"changed": true,

"enabled": false,

"name": "vsftpd",

"state": "stopped",

[root@cache ~]# systemctl is-enabled vsftpd

disabled

[root@cache ~]# systemctl status vsftpd

● vsftpd.service - Vsftpd ftp daemon

Loaded: loaded (/usr/lib/systemd/system/vsftpd.service; disabled; vendor preset: disabled)

Active: inactive (dead)

[root@ansible ~]# ansible other -m service -a 'name=vsftpd state=started enabled=yes'

cache | SUCCESS => {

"changed": true,

"enabled": true,

"name": "vsftpd",

"state": "started",

[root@cache ~]# systemctl is-enabled vsftpd

enabled

[root@cache ~]# systemctl status vsftpd

● vsftpd.service - Vsftpd ftp daemon

Loaded: loaded (/usr/lib/systemd/system/vsftpd.service; enabled; vendor preset: disabled)

Active: active (running) since 三 2019-01-09 21:37:06 EST; 20s ago

##综合练习，装包，改配置，启动服务

#web组安装httpd，启动服务，开机自启，监听8080端口

[root@ansible ~]# ansible web --list-hosts

hosts (2):

web1

web2

[root@ansible ~]# ansible web -m shell -a "rpm -qa | grep httpd"

[WARNING]: Consider using yum, dnf or zypper module rather than running rpm

web1 | FAILED | rc=1 >>

non-zero return code

web2 | FAILED | rc=1 >>

non-zero return code

[root@ansible ~]# ansible web -m yum -a 'name=httpd state=installed'

web2 | SUCCESS => {

"changed": true,

[root@ansible ~]# ansible web -m shell -a "rpm -qa | grep httpd"

[WARNING]: Consider using yum, dnf or zypper module rather than running rpm

web1 | SUCCESS | rc=0 >>

httpd-tools-2.4.6-80.el7.x86\_64

httpd-2.4.6-80.el7.x86\_64

web2 | SUCCESS | rc=0 >>

httpd-tools-2.4.6-80.el7.x86\_64

httpd-2.4.6-80.el7.x86\_64

[root@ansible ~]# ansible web -m service -a 'name=httpd enabled=yes'

[root@ansible ~]# ansible web -m lineinfile -a 'path=/etc/httpd/conf/httpd.conf regexp="^Listen" line="Listen 8080"'

web2 | SUCCESS => {

"backup": "",

"changed": true,

"msg": "line replaced"

}

web1 | SUCCESS => {

"backup": "",

"changed": true,

"msg": "line replaced"

}[root@ansible ~]# ansible web -m shell -a 'netstat -anptu | grep 8080'

web2 | SUCCESS | rc=0 >>

tcp6 0 0 :::8080 :::\* LISTEN 2552/httpd

web1 | SUCCESS | rc=0 >>

tcp6 0 0 :::8080 :::\* LISTEN 2914/httpd

[root@ansible ~]# ansible web -m service -a 'name=httpd state=started'

[root@web1 ~]# systemctl is-enabled httpd

enabled

[root@web1 ~]# systemctl status httpd

● httpd.service - The Apache HTTP Server

Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)

Active: active (running) since 三 2019-01-09 21:52:31 EST; 1min 28s ago

[root@web1 ~]# ss -ntplu | grep httpd

tcp LISTEN 0 128 :::8080 :::\* users:(("httpd",pid=2919,fd=4),("httpd",pid=2918,fd=4),("httpd",pid=2917,fd=4),("httpd",pid=2916,fd=4),("httpd",pid=2915,fd=4),("httpd",pid=2914,fd=4))

[root@web2 ~]# systemctl is-enabled httpd

enabled

[root@web2 ~]# systemctl status httpd

● httpd.service - The Apache HTTP Server

Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)

Active: active (running) since 三 2019-01-09 21:52:31 EST; 2min 36s ago

[root@web2 ~]# ss -nptlu| grep 8080

tcp LISTEN 0 128 :::8080 :::\* users:(("httpd",pid=2557,fd=4),("httpd",pid=2556,fd=4),("httpd",pid=2555,fd=4),("httpd",pid=2554,fd=4),("httpd",pid=2553,fd=4),("httpd",pid=2552,fd=4))

##使用playbook自动化管理托管主机

###重点：编写playbook时严禁使用tab键

[root@ansible ~]# mkdir playbook

[root@ansible ~]# cd playbook/

[root@ansible playbook]# vim ping.yml

[root@ansible playbook]# cat ping.yml

---

- hosts: all

remote\_user: root

tasks:

- ping:

[root@ansible playbook]# ansible-playbook ping.yml -f 2

#返回结果全绿表示正常

#使用playbook给web组添加用户，并设置初始密码

[root@ansible playbook]# vim addu1.yml

[root@ansible playbook]# cat addu1.yml

---

- hosts: web

remote\_user: root

tasks:

- name: create user wangdachui

user: name=wangdachui group=wheel uid=2000

- name: set password for wangdachui

shell: echo "123456" | passwd --stdin wangdachui

- name: set force modify password

shell: chage -d 0 wangdachui

[root@ansible playbook]# ansible-playbook addu1.yml -f 2

PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

web1 : ok=4 changed=3 unreachable=0 failed=0

web2 : ok=4 changed=3 unreachable=0 failed=0

##使用playbook给db组安装httpd服务，修改端口8080，servername为localhost，设置首页，启动服务并设置开机启动

[root@ansible playbook]# vim install\_httpd.yml

[root@ansible playbook]# cat install\_httpd.yml

---

- hosts: db

remote\_user: root

tasks:

- name: install httpd server

yum:

name: httpd

state: installed

- name: config httpd port

lineinfile:

path: /etc/httpd/conf/httpd.conf

regexp: ^Listen

line: Listen 8080

- name: config httpd servername

lineinfile:

path: /etc/httpd/conf/httpd.conf

regexp: ^#ServerName

line: ServerName localhost

- name: config index page

shell:

echo "hello world" > /var/www/html/index.html

- name: start service

service:

name: httpd

state: started

enabled: yes

[root@ansible playbook]# ansible-playbook install\_httpd.yml -f 2

PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

db1 : ok=6 changed=5 unreachable=0 failed=0

db2 : ok=6 changed=5 unreachable=0 failed=0

[root@ansible playbook]# curl http://db1:8080/

hello world

[root@ansible playbook]# curl http://db2:8080/

hello world

#playbook调用变量

[root@ansible playbook]# vim addu2.yml

[root@ansible playbook]# cat addu2.yml

---

- hosts: web

remote\_user: root

vars:

username: dd

tasks:

- name: create user

user:

name: "{{username}}"

group: users

- name: config password

shell:

echo "123456" | passwd --stdin "{{username}}"

- name: force change password

shell:

chage -d 0 "{{username}}"

[root@ansible playbook]# ansible-playbook addu2.yml

[root@ansible playbook]# ansible web -m shell -a "id dd"

web2 | SUCCESS | rc=0 >>

uid=2001(dd) gid=100(users) 组=100(users)

web1 | SUCCESS | rc=0 >>

uid=2001(dd) gid=100(users) 组=100(users)

##playbook调用过滤器

[root@ansible playbook]# vim addu3.yml

[root@ansible playbook]# cat addu3.yml

---

- hosts: other

remote\_user: root

vars:

username: nb

tasks:

- name: create user

user:

name: "{{username}}"

group: users

password: "{{'123456' | password\_hash('sha512')}}"

- name: force change password

shell:

chage -d 0 "{{username}}"

[root@ansible playbook]# ansible-playbook addu3.yml

[root@ansible playbook]# ansible other -m shell -a "id nb"

cache | SUCCESS | rc=0 >>

uid=1000(nb) gid=100(users) 组=100(users)

##playbook的错误处理

[root@ansible playbook]# cat addu4.yml

---

- hosts: other

remote\_user: root

vars:

un: nb

tasks:

- shell: adduser "{{un}}"

- shell: echo "123456" | passwd --stdin "{{un}}"

- shell: chage -d 0 "{{un}}"

#如果other主机上已经存在nb用户，则这个pb将会报错

#方法1

[root@ansible playbook]# cat addu4.yml

---

- hosts: other

remote\_user: root

vars:

un: nb

tasks:

- shell: adduser "{{un}}" || echo

- shell: echo "123456" | passwd --stdin "{{un}}"

- shell: chage -d 0 "{{un}}"

[root@ansible playbook]# ansible-playbook addu4.yml

#方法二

[root@ansible playbook]# cat addu4.yml

---

- hosts: other

remote\_user: root

vars:

un: nb

tasks:

- shell: adduser "{{un}}" || true

- shell: echo "123456" | passwd --stdin "{{un}}"

- shell: chage -d 0 "{{un}}"

[root@ansible playbook]# ansible-playbook addu4.yml

#方法三

[root@ansible playbook]# cat addu4.yml

---

- hosts: other

remote\_user: root

vars:

un: nb

tasks:

- shell: adduser "{{un}}"

ignore\_errors: True

- shell: echo "123456" | passwd --stdin "{{un}}"

- shell: chage -d 0 "{{un}}"

[root@ansible playbook]# ansible-playbook addu4.yml

##建议采用第三种方法，能够将报错信息打印到屏幕，是ansible的内置方法

##handler方法

[root@ansible playbook]# cat httpd\_hand.yml

---

- hosts: web

remote\_user: root

tasks:

- name: install httpd

yum:

name: httpd

state: installed

- lineinfile:

path: /etc/httpd/conf/httpd.conf

regexp: "^#Listen"

line: "Listen 8080"

notify:

- restart\_httpd

- lineinfile:

path: /etc/httpd/conf/httpd.conf

regexp: "^#ServerName"

line: "ServerName localhost"

notify:

- restart\_httpd

- shell:

echo "hello" > /var/www/html/index.html

handlers:

- name: restart\_httpd

service:

name: httpd

state: restarted

enabled: yes

##pb条件判断

[root@ansible playbook]# vim user01.yml

[root@ansible playbook]# cat user01.yml

---

- hosts: all

remote\_user: root

vars:

un: nb

tasks:

- shell: id "{{un}}"

register: res

- user:

name: "{{un}}"

group: users

password: "{{'123456' | password\_hash('sha512')}}"

when: res

[root@ansible playbook]# ansible-playbook user01.yml

##res的值为0的时候执行前边的user的命令

#当web服务的一分钟负载高于0.7时停止web服务

[root@ansible playbook]# vim load.yml

[root@ansible playbook]# cat load.yml

---

- hosts: web

remote\_user: root

tasks:

- shell: uptime | awk '{printf("%.2f",$(NF-2))}'

register: res

- service:

name: httpd

state: stopped

when:

res.stdout | float > 0.7

[root@ansible playbook]# ansible-playbook load.yml

[root@ansible playbook]# curl http://web1:8080

hello

[root@ansible playbook]# curl http://web2:8080

hello

[root@web1 ~]# awk 'BEGIN{while(1){}}' &

[root@web1 ~]# uptime

03:45:15 up 2:04, 3 users, load average: 0.72, 0.24, 0.12

[root@ansible playbook]# ansible-playbook load.yml

[root@ansible playbook]# curl http://web1:8080

curl: (7) Failed connect to web1:8080; 拒绝连接

[root@ansible playbook]# curl http://web2:8080

hello

##循环操作

[root@ansible playbook]# vim user02.yml

[root@ansible playbook]# cat user02.yml

---

- hosts: other

remote\_user: root

tasks:

- user:

name: "{{item}}"

group: users

password: "{{'123456' | password\_hash('sha512')}}"

with\_items: ["tt","zz","qq","aa"]

[root@ansible playbook]# ansible-playbook user02.yml

[root@ansible playbook]# ansible other -m shell -a "tail -4 /etc/passwd"

cache | SUCCESS | rc=0 >>

tt:x:1001:100::/home/tt:/bin/bash

zz:x:1002:100::/home/zz:/bin/bash

qq:x:1003:100::/home/qq:/bin/bash

aa:x:1004:100::/home/aa:/bin/bash

##循环属性复杂情况

[root@ansible playbook]# vim user03.yml

[root@ansible playbook]# cat user03.yml

---

- hosts: other

remote\_user: root

tasks:

- user:

name: "{{item.name}}"

group: "{{item.group}}"

password: "{{'123456' | password\_hash('sha512')}}"

with\_items:

- {name: "qa", group: "users"}

- {name: "az", group: "wheel"}

- {name: "qw", group: "root"}

[root@ansible playbook]# ansible-playbook user03.yml

[root@ansible playbook]# ansible other -m shell -a "tail -3 /etc/passwd"

cache | SUCCESS | rc=0 >>

qa:x:1005:100::/home/qa:/bin/bash

az:x:1006:10::/home/az:/bin/bash

qw:x:1007:0::/home/qw:/bin/bash

#嵌套循环

[root@ansible playbook]# vim qt.yml

[root@ansible playbook]# cat qt.yml

---

- hosts: cache

remote\_user: root

vars:

un: [a,b,c]

id: [1,2,3]

tasks:

- name: add user

shell: echo "{{item}}"

with\_nested:

- "{{un}}"

- "{{id}}"

[root@ansible playbook]# ansible-playbook qt.yml

PLAY [cache] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [cache]

TASK [add user] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [cache] => (item=[u'a', 1])

changed: [cache] => (item=[u'a', 2])

changed: [cache] => (item=[u'a', 3])

changed: [cache] => (item=[u'b', 1])

changed: [cache] => (item=[u'b', 2])

changed: [cache] => (item=[u'b', 3])

changed: [cache] => (item=[u'c', 1])

changed: [cache] => (item=[u'c', 2])

changed: [cache] => (item=[u'c', 3])

PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

cache : ok=2 changed=1 unreachable=0 failed=0

##特殊用法

tags：给指定的任务定义一个调用标识

include和roles：引用其他的yml文件，roles可以引用项目

--syntax-check：语法检查，精准定位yml文件错误位置

-C：测试执行

##debug

[root@ansible playbook]# vim deb.yml

[root@ansible playbook]# cat deb.yml

---

- hosts: web

remote\_user: root

tasks:

- shell: uptime | awk '{printf("%.2f",$(NF-2))}'

register: res

- shell: touch /tmp/isreboot

when: res.stdout | float > 0.5

- name: Show debug info

debug: var=res

[root@ansible playbook]# ansible-playbook deb.yml

PLAY [web] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [web1]

ok: [web2]

TASK [command] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [web2]

changed: [web1]

TASK [command] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

skipping: [web1]

skipping: [web2]

TASK [Show debug info] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [web1] => {

"res": {

"changed": true,

"cmd": "uptime | awk '{printf(\"%.2f\",$(NF-2))}'",

"delta": "0:00:00.010960",

"end": "2019-01-13 21:31:37.225779",

"failed": false,

"rc": 0,

"start": "2019-01-13 21:31:37.214819",

"stderr": "",

"stderr\_lines": [],

"stdout": "0.09",

"stdout\_lines": [

"0.09"

]

}

}

ok: [web2] => {

"res": {

"changed": true,

"cmd": "uptime | awk '{printf(\"%.2f\",$(NF-2))}'",

"delta": "0:00:00.010474",

"end": "2019-01-13 21:31:37.417769",

"failed": false,

"rc": 0,

"start": "2019-01-13 21:31:37.407295",

"stderr": "",

"stderr\_lines": [],

"stdout": "0.09",

"stdout\_lines": [

"0.09"

]

}

}

PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

web1 : ok=3 changed=1 unreachable=0 failed=0

web2 : ok=3 changed=1 unreachable=0 failed=0