# Ansible服务

目录

[Ansible服务 2](#_Toc10189)

[第一章ansible简介 2](#_Toc23514)

[1、ansible的起源 2](#_Toc27205)

[2、ansible的介绍 2](#_Toc1855)

[3、Ansible的组成 3](#_Toc16332)

[第二章ansible基础 3](#_Toc28452)

[1、什么是ansible 3](#_Toc32721)

[2、ansible优点 3](#_Toc13471)

[3、ansible特性 3](#_Toc22761)

[4、ansible工作流程 4](#_Toc32352)

[5、ansible安装 4](#_Toc1965)

[paramiko、PyYAML、Jinja2、httplib2、six 4](#_Toc11420)

[6、ansible配置文件 5](#_Toc12666)

[第三章、ansible环境准备 6](#_Toc25942)

[1、准备虚拟机 6](#_Toc27150)

[2、配置免密登陆和hosts解析 7](#_Toc1005)

[3、安装ansible 7](#_Toc25451)

[4、配置主机清单 7](#_Toc3604)

[5、结果测试 7](#_Toc349)

[第四章 ansible的模块 8](#_Toc8614)

[ad-hoc模式(点对点模块) 8](#_Toc5777)

[Ansible常用的模 8](#_Toc10734)

[1、ping模块 8](#_Toc32556)

[2、command命令模块 8](#_Toc24669)

[3、shell模块 8](#_Toc11579)

[4、script模块 9](#_Toc31684)

[5、copy模块 9](#_Toc7535)

[6、user模块 9](#_Toc19083)

[7、file模块 9](#_Toc7946)

[8、mount模块 10](#_Toc6019)

[9、service模块 10](#_Toc28649)

[10、setup模块 10](#_Toc30262)

[11、debug模块 11](#_Toc30211)

[12、cron计划任务模块 11](#_Toc15086)

[13、get\_url模块 11](#_Toc27108)

[14、synchronize模块 11](#_Toc17357)

[第五章 Ansible的playbook 11](#_Toc8675)

[playbook模式(剧本模式) 11](#_Toc9119)

[playbooks的组成部分 12](#_Toc27685)

[yaml语法格式 12](#_Toc1108)

[1、实验-包含多个play的playbook 13](#_Toc5554)

[2、实验-循环 13](#_Toc1608)

[3、实验-变量及条件判断 14](#_Toc28447)

**第一章ansible简介**

**1、ansible的起源**

Ansible的初始版本是0.0.1，发布于2012年3月9日，其作者兼创始人是Michael DeHaan。Michael DeHaan曾经供职于Puppet Labs、RedHat、Michael，在配置管理和架构设计方面有丰富的经验。其在RedHat任职期间主要开发了Cobble，经历了各种系统简化、自动化基础架构操作的失败和痛苦，在尝试了Puppet、Chef、Cfengine、Capistrano、Fabric、Function、Plain SSH等各式工具后，决定自己打造一款能结合众多工具优点的自动化工具，Ansible由此诞生。

Ansible版本更新速度非常快，有时会一天推出多个Dev版本，7天推出一个稳定版本。所以使用Ansible的过程中也需多留意官网的更新。

**2、ansible的介绍**

ansible是一种自动化运维工具,基于paramiko开发的,并且基于模块化工作，Ansible是一种集成IT系统的配置管理、应用部署、执行特定任务的开源平台，它是基于python语言，由Paramiko和PyYAML两个关键模块构建。集合了众多运维工具的优点，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.ansible不需要在远程主机上安装client/agents，因为它们是基于ssh来和远程主机通讯的.

ansible被定义为配置管理工具,配置管理工具通常具有以下功能:

确保所依赖的软件包已经被安装

配置文件包含正确的内容和正确的权限

相关服务被正确运行

常用的自动化运维工具技术特性比较:

| 项目 | Puppet | SaltStack | Ansible |
| --- | --- | --- | --- |
| 开发语言 | Ruby | Python | Python |
| 是否有客户端 | 有 | 有 | 无 |
| 是否支持二次开发 | 不支持 | 支持 | 支持 |
| 服务器与远程机器是否相互验证 | 是 | 是 | 是 |
| 服务器与远程机器的通信是否加密 | 是，标准的SSL协议 | 是，使用AES加密 | 是，使用OpenSSH |
| 平台支持 | AIX , BSD, HP-UX, Linux , Mac OSX , Solaris, Windows | BSD, Linux , Mac OS X , Solaris, Windows | AIX , BSD , HP-UX , Linux , Mac OS X , Solaris |
| 是否提供Web UI | 提供 | 提供 | 提供，但是是商业版本 |
| 配置文件格式 | Ruby 语法格式 | YAML | YAML |
| 命令行执行 | 不支持，大师可以通过配置模块实现 | 支持 | 支持 |

**3、Ansible的组成**

Ansible主要由一下6部分组成

（1）Ansible Playbooks：任务集，编排定义Ansible任务集的配置文件，由Ansible顺序依次执行，通常是json格式的YML文件。

（2）Inventory：Ansible管理主机的清单。

（3）Modules：Ansible执行命令的功能模块，多数为内置的核心模块，也支持自定义

（4）Plugins：模块功能的补充，如连接类型插件、循环插件、变量插件、过滤插件等，改功能不常用。

（5）API：供第三方程序调用的应用程序编程接口；

（6）Ansible：此处指的是组合nventory、Modules、Plugins、API的Ansible命令工具，其为核心执行工具

**第二章ansible基础**

**1、什么是ansible**

ansible是2013年推出的一款IT自劢化和DevOps软 件，2015年被RedHat收购。是基亍Python研发， 糅合很多老运维工具的优点，实现了批量操作系统配 置，批量程序部署，批量运行命令等功能

ansible可以实现：

自劢化部署APP、自劢化管理配置项、自劢化持续交付、自劢化（AWS）于服务管理

**2、ansible优点**

只需要SSH和Python即可使用

无客户端

ansible功能强大，模块丰富

上手容易，门槛低

基亍Python开发，做二次开发更容易

使用公司比较多，社区活跃

**3、ansible特性**

模块化设计，调用特定的模块完成特定任务

基于Python语言实现

Paramiko、PyYAML (半结构化诧言)

Jinja2

其模块支持JSON等标准输出格式，可以采用任何编 程语言重写

部署简单

主从模式工作

支持自定义模块

支持playbook

易亍使用

支持多层部署

支持异构IT环境

**4、ansible工作流程**

使用Ansible或Ansible-playbooks时，在服务器终端输入Ansible的Ad-Hoc命令集或palybook后，Ansible会遵循预先编排的规则将Playbooks逐条拆解为Play，再将paly组织成Ansible可识别的任务（Task），随后调用任务涉及的所有模块（modules）和插件（plugins），根据Inventory中定义的主机列表通过SSH将任务集以临时文件或命令的形式传输到远程客户端执行并返回执行结果，如果是临时文件，则执行完毕后自动删除。

ansible大体执行过程

**5、ansible安装**

软件依赖关系

（1）对管理主机

要求Python 2.6版本以上

ansible 使用以下模块，都需要安装

paramiko、PyYAML、Jinja2、httplib2、six

（2）对于被托管主机

<1>ansible默认通过SSH协议管理机器

<2>被管理主机要开吭ssh服务，允许ansible主机登彔

<3>在托管节点上也需要安装Python2.5以上的版本，如果托管节点上开了SElinux，需要安装libselinux-python

ansible源码安装

wget <http://releases.ansible.com/ansible/ansible-latest.tar.gz> #下载ansible源码包不同版本地址不一样

apt-get install gcc python-pip python-dev libffi-dev sshpass

#源码编译器需要安装以下包才能顺利编译

pip install paramiko PyYAML Jinja2 httplib2 six #安装ansible的依赖

tar -xf ansible-latest.tar.gz # 解压缩

cd ansible-2.9.9/ #切到ansible安装目录

python setup.py build

python setup.py install #编译安装

mkdir /etc/ansible #创建ansible目录

cp ansible-2.9.9/examples/\* /etc/ansible/ #把ansible配置文件复制到ansible目录下

Ansible的apt安装

pip install paramiko PyYAML Jinja2 httplib2 six #安装ansible的依赖

apt install -y ansible #安装ansible

**6、ansible配置文件**

配置文件: /etc/ansible/

执行文件目录: /usr/bin/

ib依赖库: /usr/lib/python2.7/site-packages/ansible/

help文件: /usr/lib/python2.7/site-packages/ansible

ansible配置文件查找顺序

检查环境变量ANSIBLE\_CONFIG指向的路径文件(export ANSIBLE\_CONFIG=/etc/ansible.cfg)；

./ansible.cfg，检查当前目录下的ansible.cfg配置文件；

~/ansible.cfg 检查当前用户家目录下

/etc/ansible.cfg检查etc目录的配置文件。

ansible配置文件 cat /etc/ansible/ansible.cfg  
ansible 有许多参数，下面我们列出一些常见的参数：

inventory = /etc/ansible/hosts #这个参数表示资源清单inventory文件的位置

library = /usr/share/ansible #指向存放Ansible模块的目录，支持多个目录方式，只要用冒号（：）隔开就可以

forks = 5 #并发连接数，默认为5

sudo\_user = root #设置默认执行命令的用户

remote\_port = 22 #指定连接被管节点的管理端口，默认为22端口，建议修改，能够更加安全

host\_key\_checking = False #设置是否检查SSH主机的密钥，值为True/False。关闭后第一次连接不会提示配置实例

timeout = 60 #设置SSH连接的超时时间，单位为秒

log\_path = /var/log/ansible.log #指定一个存储ansible日志的文件（默认不记录日志）

(3)定义Inventory(主机列表)

ansible的主要功用在于批量主机操作，为了便捷地使用其中的部分主机，可以在inventory file中将其分组命名。默认的inventory file为/etc/ansible/hosts。  
inventory file可以有多个，且也可以通过Dynamic Inventory来动态生成。

Inventory文件格式:

inventory文件遵循INI文件风格，中括号中的字符为组名。可以将同一个主机同时归并到多个不同的组中；此外，当如若目标主机使用了非默认的SSH端口，还可以在主机名称之后使用冒号加端口号来标明。

ntp.com

[webservers]

www1.com:2222

www2.com

[dbservers]

db1.com

db2.com

db3.com

如果主机名称遵循相似的命名模式，还可以使用列表的方式标识各主机，例如：

[webservers]

www[01:50].example.com

[databases]

db-[a:f].example.com

主机变量: 可以在inventory中定义主机时为其添加主机变量以便于在playbook中使用。例如：

[webservers]

www1.com http\_port=80 maxRequestsPerChild=808

www2.com http\_port=8080 maxRequestsPerChild=909

组变量

[webservers]

www1.com

www2.com

[webservers:vars]

ntp\_server=ntp.com

nfs\_server=nfs.com

inventory其他的参数

ansible基于ssh连接inventory中指定的远程主机时，还可以通过参数指定其交互方式；这些参数如下所示：

ansible\_ssh\_host # 远程主机

ansible\_ssh\_port # 指定远程主机ssh端口

ansible\_ssh\_user # ssh连接远程主机的用户,默认root

ansible\_ssh\_pass # 连接远程主机使用的密码,在文件中明文,建议使用--ask-pass或者使用SSH keys

ansible\_sudo\_pass # sudo密码, 建议使用--ask-sudo-pass

ansible\_connection # 指定连接类型: local, ssh, paramiko

ansible\_ssh\_private\_key\_file # ssh 连接使用的私钥

ansible\_shell\_type # 指定连接对端的shell类型, 默认sh,支持csh,fish

ansible\_python\_interpreter # 指定对端使用的python编译器的路径

**第三章、ansible环境准备**

**1、准备虚拟机**

4台虚拟机：

（ansible 192.168.200.10）

（node1 192.168.200.11）

（node2 192.168.200.12）

（node3 192.168.200.13）

**2、配置免密登陆和hosts解析**

ssh-keygen 一直回车 #生成密钥

ssh-copy-id node1 #拷贝私钥

所有机器都必须是免密登陆

4台主机上面配置/etc/hosts，以ansible主机为例

vim /etc/hosts

192.168.200.10 ansible

192.168.200.11 node1

192.168.200.12 node2

192.168.200.13 node3

for i in {10..13}; do scp /etc/hosts 192.168.200.$i:/etc/hosts; done

**3、安装ansible**

apt install -y ansible

**4、配置主机清单**

vim /etc/ansible/hosts

[web]

node1

node2

[db]

node3

**5、结果测试**

ansible all -m ping

node2 | SUCCESS => {

"changed": false,

"ping": "pong"

}

node1 | SUCCESS => {

"changed": false,

"ping": "pong"

}

node3 | SUCCESS => {

"changed": false,

"ping": "pong"

}

ansible web -m ping

node2 | SUCCESS => {

"changed": false,

"ping": "pong"

}

node1 | SUCCESS => {

"changed": false,

"ping": "pong"

}

**第四章 ansible的模块**

**ad-hoc模式(点对点模块)**

使用单个模块,支持批量执行单条命令,相当与在bash中执行一句shell命令

ansible-doc -l列出所有可用的模块

ansible -s <module-name>可以查看指定module的用法,或者参看官方帮助文档:

语法格式

ansible 主机模式 -m 模块 [-a '参数'] [-i 主机清单文件]

**Ansible常用的模**

文件模块 copy:把文件从本机拷贝到远程

file:设置文件的权限等属性

lineinfile:确保文件中存在或不存在某些行

syncchronize:能过 rsync同步内容

软件包模块 package:通过自动检测，使用操作系统原生的包管理器管理软件

apt:通过apt进行软件包管理

mount:文件系统挂载

dnf:通过dnf进行软件包管理

gem:管理 Ruby gems

pip:从PyPI进行 python包管理

系统模块 firewall:通过 firewall管理服务和端口

reboot:重启主机

service:管理服务

user:添加、删除、管理用户

网络工具模块 get\_url:通过htp/htts/ftp下载文件

nmcli:管理网络

uri:与Web服务交互

**1、ping模块**

ansible all -m ping #查看ping客户端组情况

**2、command命令模块**

ansible all -m command -a "uptime" #查看客户端负载信息

ansible all -m command -a "useradd uostest" #在客户端添加用户

ansible all -m command -a "touch /uostest" #在客户端建立文件

**3、shell模块**

让远程主机在shell进程下执行命令，从而支持shell的特性，如管道等

ansible all -m shell -a "echo "uostest:uostest" | chpasswd"

chdir：在运行命令之前，切换到此目录。

executable：更改用于执行命令的shell（bash，sh）。 应该是可执行文件的绝对路径。

**4、script模块**

将uos1上的脚本在uos2、uos3和uos4上运行

vim /tmp/uosscript.sh

#!/bin/bash

touch /tmp/uosscript

useradd uosscript

echo "uosscript:uosscript" | chpasswd

ansible all -m script -a "/tmp/uosscript.sh" #在所有主机上批量执行uosscript脚本

**5、copy模块**

ansible all -m copy -a "src=/etc/fstab dest=/tmp/ owner=uostest group=root mode=0600"

ansible all -m copy -a "src=/var/log/apt/ dest=/tmp/ owner=uostest group=root mode=0750"

#src主控端文件位置,dest被控端目标位置,owner文件复制过去后的所有者,group文件复制过去后的所属组,mode文件的权限设定

**6、user模块**

ansible all -m user -a "name=uosuser state=present groups=uostest uid=2000 createhome=yes home=/tmp/uosuser password=uosuser shell=/bin/bash comment='The user for uos' append=yes"

ansible all -m user -a "name=uosuser state=absent remove=yes"

name:用户名

password:为用户设置登陆密码，此密码是明文密码加密后的密码

update\_password:always/on\_create

always:只有当密码不相同时才会更新密码(默认)

on\_create:只为新用户设置密码

shell:用户的shell设定

groups:用户组设定

home:指定用户的家目录

state:present/absent

append:yes/no

yes:增量添加group

no:全量变更group,只设置groups指定的group组(默认)

remove：配合state=absent使用，删除用户的家目录->remove=yes

expires：设置用户的过期时间，值是一个时间戳

**7、file模块**

ansible all -m file -a "src=/etc/fstab dest=/tmp/uosfstab state=link"

ansible all -m file -a "path=/tmp/uosfile state=touch owner=uostest group=root mode=0600"

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

**8、mount模块**

ansible all -m mount -a "src=/dev/sr0 path=/mnt/cdrom fstype=iso9660 opts=ro state=mounted"

ansible all -m mount -a "src=/dev/sr0 path=/mnt/cdrom fstype=iso9660 opts=ro state=absent"

fstype：必选项，挂载文件的类型

path：必选项，挂载点

opts：传递给mount命令的参数

src：必选项，要挂载的文件

state： present 添加到fstab中，不挂载

absent 删除fstab，并卸载资源

mounted 添加到fstab中，并自动挂载

umounted 不删除fstab信息，只卸载资源

**9、service模块**

ansible all -m service -a "name=sshd state=started enabled=yes"

arguments：给命令行提供一些选项

enabled：是否开机启动 yes|no, 要求状态（state）和启用（enabled）中至少有一个。

name：必选项，服务名称

runlevel：运行级别

sleep：如果执行了restarted，在则stop和start之间沉睡几秒钟

state：对当前服务执行启动，停止、重启、重新加载等操作（started,stopped,restarted,reloaded）

**10、setup模块**

ansible all -m setup #查看所有主机信息

ansible all -m setup -a "filter=ansible\_hostname" #查看主机名

ansible all -m setup -a "filter=ansible\_kernel" #查看内核信息

ansible all -m setup -a "filter=ansible\_memory\_mb" #查看内存信息

ansible all -m setup -a "filter=ansible\_default\_ipv4" #查看网卡ipv4信息

ansible all -m setup -a "filter=\*user\*" #查看user相关信息

**11、debug模块**

ansible all -m debug -a "msg=hahahahahaha"

msg：调试输出的消息

var：将某个任务执行的输出作为变量传递给debug模块，debug会直接将其打印输出

verbosity：debug的级别（默认是0级，全部显示）

**12、cron计划任务模块**

ansible all -m cron -a "name='uoscron' minute='30' hour='12' job='touch /tmp/uoscron' user=root"

ansible all -m cron -a 'minute="\*/1" job="/usr/bin/date >> /date.txt" name="date job"'

minute= /hour= /day= /month= /weekday= 某个值不写，默认就是\*

name:必选项，任务描述信息

job:执行的任务，要加引号

state:present(创建)/absent(删除)

**13、get\_url模块**

ansible all -m get\_url -a 'url=http://192.168.200.10/favicon.ico dest=/tmp' #配置好url

sha256sum：下载完成后进行sha256 check；

timeout：下载超时时间，默认10s

url：下载的URL

url\_password、url\_username：主要用于需要用户名密码进行验证的情况

dest：将文件下载到哪里的绝对路径。如果dest是目录，则使用服务器提供的文件名，或者如果没有提供，将使用远程服务器上的URL的基本名称。

headers：以格式“key：value，key：value”为请求添加自定义HTTP标头。

**14、synchronize模块**

delete=yes 使两边的内容一样（即以推送方为主）

compress=yes 开启压缩，默认为开启

--exclude=.Git 忽略同步.git结尾的文件

mode=pull 更改推送模式为拉取模式

在uos1上

mkdir /uos

touch /uos/file{1..99}

ansible all -m synchronize -a 'src=/uos dest=/tmp/ compress=yes'

在uos2上

ll /tmp/uos

在uos1上

mkdir /uostest

ansible node1 -m synchronize -a ' mode=pull src=/tmp/uos dest=/uostest'

ll /uostest/uos

**第五章 Ansible的playbook**

**playbook模式(剧本模式)**

ansible主要的管理方式,通过多个

task的集合完成一类功能,可以理解为多个ad-hoc的配置文件

playbook是由一个或多个“play”组成的列表。play的主要功能在于将事先归并为一组的主机装扮成事先通过ansible中的task定义好的角色。从根本上来讲，所谓task无非是调用ansible的一个module。将多个play组织在一个playbook中，即可以让它们联同起来按事先编排的机制同唱一台大戏。

**playbooks的组成部分**

Target section: 定义要运行playbook的远程主机组

hosts: hosts用于指定要执行指定任务的主机，其可以是一个或多个由冒号分隔主机组

user: 指定远程主机上的执行任务的用户,还可以指定sudo用户等

Variable section: 定义playbook运行时使用的变量

Task section: 定义要在远程主机上运行的任务列表

name: 每个任务都有name,建议描述任务执行步骤,未通过name会用执行结果作为name

'module:options': 调用的module和传入的参数args

Handler section: 定义task完成后需要调用的任务

notify: 在Task Section在每个play的最后触发,调用在hendler中定义的操作

handler: 也是task的列表

示例如：

- hosts: master

user: root

vars:

- motd\_warning: 'WARNING: Use by master ONLY'

tasks:

- name: setup a MOTD

copy: dest=/etc/motd content="{{ motd\_warning }}"

notify: say something

handlers:

- name: say something

command: echo "copy OK"

**yaml语法格式**

同级别数据元素必须有相同的缩进

子元素必须比父元素缩进更多

推荐缩进两个空格

每个文件首行是“--”，最后一行是“...”

playbook是play 的列表，列表项表示为“-”

paly本身是key:val对的字典

帮助查看

ansible-doc apt

执行playbook

ansible-playbook --syntax-check \*\*.yml //检测语法格式

ansible-playbook \*\*.yml //执行playbook

**1、实验-包含多个play的playbook**

vim tow.yml

---

- name: Execute NODE1 #一个play

hosts: node1 #执行的主机组名

tasks: #任务列表

- name: install http #描述任务

apt: #模块名

name: apache2 #安装的包名

state: present #允许安装

- name: Start service httpd, if not started #描述任务

service: #模块名

name: apache2 #启动的服务名

state: started #启动的服务

- name: Execute NODE2

hosts:

- node2

tasks:

- name: install gcc

apt:

name: gcc

state: present

ansible-playbook --syntax-check tow.yml #检查yml语法

ansible all -m shell -a "apt update" #执行更新apt源

ansible-playbook tow.yml #运维playbook的yml文件

curl 192.168.200.11 //验证可以访问到 #验证

**2、实验-循环**

同时启动ssh和apache

vim Circulates.yml

---

- name: xhlx

hosts: node1

tasks:

- name: sshd and apache2 are running

service:

name: "{{ item }}"

state: started

loop:

- sshd

- apache2

ansible-playbook --syntax-check Circulates.yml

ansible-playbook Circulates.yml

**3、实验-变量及条件判断**

变量my\_service已定义，如果定义了，安装相应的软件包，没定义则跳过、也不会报错

vim tjpd.yml

---

- name: panduan

hosts: all

vars: #定义变量名

my\_service: apache2

tasks:

- name: "{{ my\_service }} package is installed"

apt:

name: "{{ my\_service }}"

when: my\_service is defined

判断路径:

- hosts: all

remote\_user: root

gather\_facts: no

vars:

testpath1: "/etc"

testpath2: "/haha"

tasks:

- debug:

msg: "directory"

when: testpath1 is directory

- debug:

msg: "file"

when: testpath2 is file