**15Architecture\_01ansible基础+ad-hoc+批量配置管理**

**一 ansible基础-ansible简介**

**1.1 什么是ansible**

ansible是2013年退出的一款IT自动化和devops软件,2015年被redhat收购.基于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:

自动化部署APP\自动化管理配置项\自动化持续交付\自动化AWS云服务管理

**1.2 为什么选择ansible**

选择一款配置管理软件,无外乎从以下几点来权衡利弊:

活跃度(社区)\学习成本\使用成本\编码语言\性能\使用是否广泛

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\基于Python开发,做二次开发更容易\使用公司比较多,社区活跃

**1.3 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(半结构化语言)\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

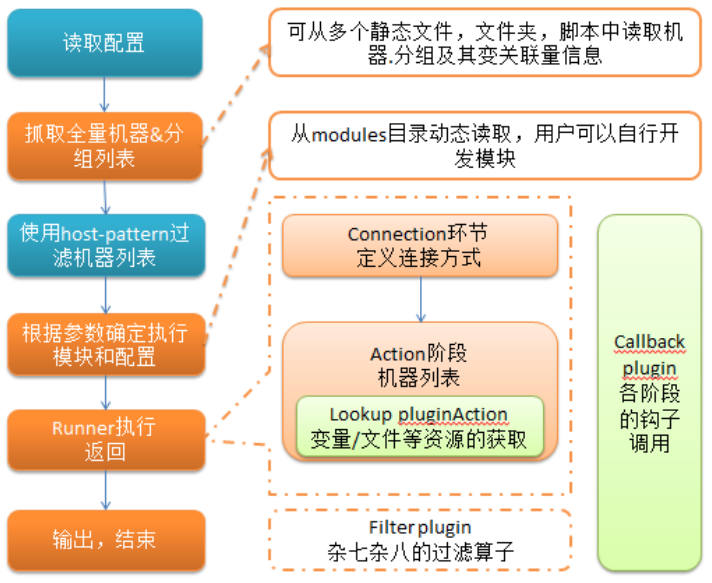
部署简单\主从模式工作\支持自定义模块\支持playbook\易于使用\支持多层部署\支持异构IT环境

命令集合

主机集合

**1.4 ansible工作流程**

ansible大体执行过程



**二 ansible基础-ansible安装**

**2.1 软件依赖关系**

**2.1.1 对管理主机**

要求Python2.6或Python2.7

ansible使用以下模块,都需要安装

paramiko\PyYAML\jinja2\httplib2\six

**2.1.2 对于被托管主机**

ansible默认通过SSH协议管理机器

被管理主机要开启ssh服务,允许ansible主机登录

在托管节点上也需要安装Python2.5或以上的版本

如果托管节点上开启了SELinux,需要安装libselinux-python

**2.2 安装ansible**

2.2.1 yum安装

把软件拷贝到真机ftp共享目录

room9pc01 ~]# cp -r /linux-soft/04/ansible /var/ftp/

更新索引文件

room9pc01 ~]# createrepo /var/ftp/ansible/

**2.2.2 安装及验证**

在ansible托管主机上配置yum配置文件

ansible40 ~]# vim /etc/yum.repos.d/local.repo

[ansiable]

name=ansible

baseurl="ftp://192.168.1.254/ansible"

enabled=1

gpgcheck=0

安装yum install ansible

ansible40 ~]# yum -y install ansible

验证ansible --version

ansible40 ~]# ansible --version

ansible 2.4.2.0

config file = /etc/ansible/ansible.cfg

executable location = /usr/bin/ansible

**2.3 6台主机上配置/etc/hosts**

ansible40 ~]# vim /etc/hosts #添加以下6行

192.168.1.40 ansible40

192.168.1.41 web141

192.168.1.42 web242

192.168.1.43 db143

192.168.1.44 db244

192.168.1.45 cache45

ansible40 ~]# for i in {40..45};

> do scp /etc/hosts root@192.168.1.$i:/etc/hosts;

> done

**2.4 批量部署SSH证书文件**

ansible40 ~]# ssh-keygen -t rsa -b 2048 -N '' -f /root/.ssh/id\_rsa

ansible40 ~]# for i in {41..45};

> do ssh-copy-id -i /root/.ssh/id\_rsa.pub root@192.168.1.$i;

> done

**三 ad-hoc 主机管理**

**3.1 主机定义与分组**

**3.1.1 安装ansible之后可以做一些简单的任务**

**3.1.2 安装ansible配置文件查找顺序**

首先检查ANSIBLE\_CONFIG变量定义的配置文件

其次检查当前目录下的./ansible.cfg文件

再次检查当前用户家目录下~/ansible.cfg文件

最后检查/etc/ansible/ansible.cfg文件

**3.1.3 /etc/ansible/ansible.cfg是ansible的默认配置文件路径**

**3.2 ansible.cfg配置文件**

inventory定义托管主机地址配置文件路径名

inventory指定的配置文件,写如远程主机的地址

ssh主机key验证配置参数

host\_key\_checking = False

如果为False,不需要输入yes

如果为True,等待输入yes

ansible40 ~]# vim /etc/ansible/ansible.cfg #解除3行注释

14 inventory　= /etc/ansible/hosts 　#定义托管主机地址配置文件路径

61 host\_key\_checking = False #设置不输入ssh主机key

97 remote\_user = root #设置远程连接用户为root

**3.3 分组定义\范围定义 /etc/ansible/hosts[托管主机配置文件]**

**格式**

#表示注释

[组名称]

主机名称或IP地址 其他参数

ansible40 ~]# vim /etc/ansible/hosts #43行以下添加以下内容

[web]

web141

web242

[db]

db143

db244

[other]

cache45

**3.4 ansible命令基础**

**3.4.1 命令格式: ansible主机集合 -m 模块名称 -a 模块参数**

主机集合:主机名或分组名,多个使用逗号分隔

-m 模块名称,默认command模块

-a or --args 模块参数

**3.4.2 其他参数:**

**-i inventory文件路径,或可执行脚本**

**-k 使用交互式登录密码**

**-e 定义变量**

**-v 显示详细信息**

**ansible all --list-hosts 列出要执行的主机**

**ansible all -m ping -k 批量ping检测主机**

ansible40 ~]# ansible web --list-hosts

hosts (2):

web141

web242

ansible40 ~]# ansible web,db,other --list-hosts

hosts (5):

web141

web242

db143

db244

cache45

ansible40 ~]# ansible web141,db244 --list-hosts

hosts (2):

web141

db244

ansible40 ~]# ansible all -m ping -k

SSH password: #输入密码a

**3.4.3 部署证书文件**(2.4已部署完成)

ansible是通过SSH在远程执行命令的

SSH远程执行命令必须要通过认证才行

密码写入配置文件安全行很差

使用key方式认证

给所有主机部署公钥

没有秘钥命令执行会出错

ansible web -a ‘uptime’

**3.5 inventory[ˈɪnvəntɔːri]扩展参数**

3.5.1 ansible\_ssh\_port 指定ssh端口号

3.5.2 ansible\_ssh\_user 指定ssh用户名

3.5.3 ansilbe\_ssh\_pass 指定ssh密码

3.5.4 ansible\_ssh\_private\_key\_file 指定ssh使用的公钥文件

3.5.5 vars变量定义,用于组名后面,例如[all:vars]表示所有主机

ansible40 ~]# vim /etc/ansible/hosts

[other] #指定ssh到cache45主机使用的私钥文件

cache45 ansible\_ssh\_private\_key\_file=”/root/.ssh/id\_rsa”

[all:vars] #给本机指定ssh其他主机使用的私钥文件

ansible\_ssh\_private\_key\_file=”/root/.ssh/id\_rsa”

3.5.6 children子组定义,用于引用其他组名称,例如

[app:children] #app组下有web组和db组

web

db

3.5.7 自定义配置文件

创建文件夹myansible

ansible40 ~]# mkdir myansible

ansible40 ~]# cd myansible/

创建配置文件ansible.cfg

ansible40 myansible]# vim ansible.cfg

[defaults]

inventory = myhost

host\_key\_checking = False

创建主机文件

ansible40 myansible]# vim myhost

[app1]

web141

db143

验证

ansible40 myansible]# ansible all -m ping

**自定义的配置文件及主机文件,只有在自定义的目录内才生效**

**3.4 动态主机**

无限可能

ansible Inventory包含静态和动态的inventory,静态inventory指在文件/etc/ansible/host中指定的主机和组,动态inventory指通过外部脚本获取主机列表,按照其要求格式返回给ansible命令

JSON(javascript object notation[nəʊˈteɪʃn]符号,记号

,JavaScript对象表示法),一种基于文本独立于语言的轻量级数据交换格式

**四 批量配置管理-模块**

**4.1 ansible-doc和ping模块**

ansible-doc模块的手册,相当于shell的man,很重要

ansible-doc -l 列出所有模块

ansible-doc modulename 查看模块帮助

**ping模块**

测试网络连通性,ping模块没有参数

ansible host-pattern -m ping

**4.2 command模块**

默认模块,远程执行命令

**用法: ansible host-pattern -m connmnd -a ‘[args]’**

查看所有主机的日期和时间

ansible40 ~]# ansible all -m command -a 'date +%F\%T'

查看web主机的负载

ansible40 ~]# ansible web -m command -a 'uptime'

command模块注意事项:

命令里不能使用管道,不能解析系统变量

该模块通过-a跟上要执行的命令可以直接执行,若命令有如下字符则执行不成功

< > | &该模块不启动shell直接在ssh进程中执行,所有使用到shell的命令执行都会失败

**4.3 shell模块**

shell模块用法基本和command一样,区别是shell模块是通过/bin/sh进行执行命令,可以执行任意命令

不能执行交互式的命令,例如vim top等

ansible40 ~]# ansible web -m shell -a "echo ${HOSTNAME}"

web141 | SUCCESS | rc=0 >>

ansible40

web242 | SUCCESS | rc=0 >>

ansible40

ansible40 ~]# ansible web -m shell -a 'echo ${HOSTNAME}'

web141 | SUCCESS | rc=0 >>

web141

web242 | SUCCESS | rc=0 >>

web242

**变量解析**

ansilbe执行命令是二次解析

第一次在本机解析,第二次在执行机器解析

执行第二次解析的变量要转义(\)

ansible 是使用ssh多次连接执行

ssh连接退出以后,之前ssh连接时的状态全部失效

解决方法:使用chdir代替cd命令

ansible cache -m shell -a ‘chdir=/tmp touch testfile’

案例:添加用户

ansible40 ~]# ansible web141,db244 -m shell -a 'useradd nb'

ansible40 ~]# ansible web141,db244 -m shell -a 'echo "123" | passwd --stdin nb'

4.4 script模块

,命令太复杂,可以在本地写脚本,然后使用script模块批量执行

ansible web -m script -a /路径/脚本名称

脚本包含但不限于shell脚本,只要指定Sha-bang解释器的脚本都可以执行

思考:添加用户

给所有web主机添加用户wk

要求nb用户与wk用户不能出现在同一台主机上

设置wk用户的密码是456

ansible40 ~]# vim wk.sh

#!/bin/bash

id nb &> /dev/null

if [ $? -ne 0 ];then

useradd wk

echo 456 | passwd --stdin wk

fi

ansible40 ~]# ansible web -m script -a /root/wk.sh

**4.5 copy模块**

复制文件到远程主机

src: 复制本地文件到远程主机,绝对路径和相对路径都可以.路径为目录时会递归复制.若路径以”/”结尾,只复制目录里面的内容;若路径不以”/”结尾,则复制包含目录在内的整个目录内容,类似于rsync

dest: 必选项.远程主机的绝对路径,如果源文件是一个目录,那该路径必须是目录

backup: 覆盖前先备份原文件,备份文件包含时间信息.有两个选项:yes|no

force: 若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,设置为no,则只有但目标主机的目标位置不存在该文件时才复制.默认为yes

**批量修改配置文件**

批量修改所有机器的dns配置/etc/reslov.conf

ansible40 ~]# ansible all -m copy -a 'src="/etc/resolv.conf" dest="/etc/resolv.conf"'

**批量同步所有机器的yum配置文件**

ansible40 ~]# ansible all -m copy -a 'src="/etc/yum.repos.d/" dest="/etc/yum.repos.d/"'

ansible40 ~]# ansible all -m shell -a 'yum repolist' #验证

**删除db主机的mariadb,安装mysql,开启binlog日志**

ansible40 ~]# ansible db -m yum -a 'state="removed" name="mariadb"'

ansible40 ~]# ansible db -m yum -a 'state="installed" name="mysql"'

ansible40 ~]# vim /etc/my.cnf

[mysqld] #添加以下两行

log\_bin=mysqlsql-bin

binlog-format=mixed

ansible40 ~]# ansible db -m copy -a 'src="/etc/my.cnf" dest="/etc/my.cnf"'

ansible40 ~]# ansible db -m shell -a 'head -n 6 /etc/my.cnf'

**4.6 lineinfle模块**

类似sed的一样行编辑替换模块

path 目标文件

regexp 正则表达式,要修改的行.匹配到后,整行删除,替换上line指定的结果

line 最终修改的结果

修改db主机的my.cnf中bin-log的格式:mixed --> row

ansible40 ~]# ansible db -m lineinfile -a 'path="/etc/my.cnf" regexp="=mixed$" line="binlog-format=row"'

验证: ansible40 ~]# ansible db -m shell -a ‘grep “binlog-format” /etc/my.cnf’

**4.7** **replace模块**

类似sed的一样行编辑替换模块

path 目标文件

regexp 正则表达式,要修改的部分.匹配到后,删除,替换上replace指定的结果

replace 最终修改的结果

替换db主机中my.cnf中binlog-fortmat: row --> mixed

ansible40 ~]# ansible db -m replace -a 'path="/etc/my.cnf" regexp="=row$" replace="=mixed"'

验证:ansible40 ~]# ansible db -m shell -a 'grep "binlog-format" /etc/my.cnf'

**4.8 yum模块**

使用yum包管理器来管理软件包,前提是被控主机有完整yum源

name: 要进行操作的软件包名字

state: 动作(installed,removed)

install -> installed yum中install对应ansible中installed,下同

remove -> removed

软件包可以写多个,用逗号隔开

给所有db主机安装mariadb

ansible40 ~]# ansible db -m yum -a 'state=installed name="mariadb-server"'

ansible40 ~]# ssh db143 #ssh到db143验证mariadb已安装

db143 ~]# rpm -qa | grep mariadb

cache主机删除lrzsz

ansible40 ~]# ansible cache45 -m yum -a 'state=removed name="lrzsz"'

**4.9 service模块**

name: 必选项,服务名称

enabled: 是否开机启动 yes|no

sleep: 执行restarted,会在stop和start之间沉睡几秒钟

state: started,stopped,restarted,reloaded

给所有web主机安装mariadb,并设置开机自启动

ansible40 ~]# ansible web -m yum -a 'state=installed name="mariadb-server"'

ansible40 ~]# ansible web -m service -a 'name="mariadb" state=started enabled="yes"'

nsible40 ~]# ansible web -m shell -a 'ss -antulp | grep :3306'

**4.10 setup模块**

主要用于获取主机信息,playbooks里经常会用的另一个参数gather\_facts与该模块相关,setup模块下经常用的是fileter参数

filter过滤所需信息

ansible40 ~]# ansible cache45 -m setup -a 'filter=ansible\_distribution'