automation day 01

================================================

**ansible**

**是一款自动化运维工具，基于python开发**

**可以对系统进行批量配置、批量执行任务等操作**

**环境准备：开启6台虚拟机**

**1，先在control主机修改hosts文件**

**装一些常用软件包：**

[root@control ~]# dnf -y install bash-completion

[root@control ~]# dnf -y install vim

[root@control ~]# dnf -y install net-tools

[root@control ~]# vim /etc/hosts //修改文件，手动添加如下内容（不要删除文件原来的内容）

192.168.4.253 control con //这里加了control和con两个域名，也就是以后敲任意一个域名都可以找到4.253，写一个control也行。

192.168.4.11 node1 n1

192.168.4.12 node2 n2

192.168.4.13 node3 n3

192.168.4.14 node4 n4

192.168.4.15 node5 n5

**2，传递秘钥，实现control主机对node主机免密操作**

[root@control ~]#ping node1 //依次测试6台主机

Ansible是基于SSH远程的原理实现远程控制，如果控制端主机无法免密登录被管理端主机，后续的所有试验都会失败。

[root@control ~]# ssh-keygen //生成秘钥

[root@control ~]# for i in node1 node2 node3 node4 node5

do

ssh-copy-id $i //拷贝密钥到远程主机，过程中要反复输入不同主机的密码

done

[root@control ~]# ssh node1 //之后使用ssh连接测试

[root@node1 ~]# exit

3，**部署Ansible软件（仅Control主机操作，软件包在ansible\_soft目录）。**

[root@control ~]# tar -xf ansible\_soft.tar.gz //拷贝tar包到家目录，然后释放

[root@control ~]# cd ansible\_soft //进入目录可以看到很多关于ansible的包

[root@control ansible\_soft]# dnf -y install \* //安装这些包

**4，修改配置文件**

**主配置文件名是ansible.cfg ( 默认位置/etc/ansible/ )**

**ansible对配置文件的查找顺序：**

**首先检测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**

**再检查家目录下的ansible.cfg文件**

**最后检查/etc/ansible/ansible.cfg文件**

[root@control ~]# mkdir ansible

[root@control ~]# cd ansible

[root@control ansible]# vim ansible.cfg

[defaults]

inventory = /root/ansible/inventory //被控主机清单文件，文件名可以自定义

#forks = 5 //ssh并发数量

#ask\_pass = True //使用密码还是秘钥远程管理被控主机

True是使用密码，False是秘钥

#host\_key\_checking = False //第一次ssh时没有yes/no的提示

**5，修改主机清单文件（清单文件名必须与主配置文件inventory定义的一致）**

[root@control ansible]# vim inventory

[test] //定义主机组（组名称任意）

node1 //定义组中的具体主机，组中包括一台主机node1

[proxy] //再定义主机组

node2

[webserver]

node3

node4

[database]

node5

[cluster:children] //嵌套组（children为关键字）

webserver //嵌套组可以在组中包含其他组

database

-------------------------------------------------------------------------

**ansible ad-hoc应用**

**ansible ad-hoc是一种通过命令批量管理的方式，基本格式如下：**

**ansible 主机 -m 模块 -a "参数"**

**1，查看主机列表并测试**

[root@control /]#cd ~/ansible

[root@control ansible]# ansible all --list-hosts //查看所有主机列表

[root@control ansible]# ansible all -m ping //调用ping模块测试被控

主机，看到SUCCESS字样则成功

[root@control ansible]# ansible node1 -m ping //测试一台主机

[root@control ansible]# ansible webserver -m ping //使用组名测试,被

测对象是该组中所有主机

[root@control ansible]# ansible node1,node2 -m ping //测试多台主机

**2，使用command模块，可以对远程主机执行普通的shell命令**

[root@control ansible]# ansible node1 -m command -a "uptime" //查看node1主机的CPU负载

[root@control ansible]# ansible node1,node2 -m command -a "uptime" //查看两台主机的CPU负载

[root@control ansible]# ansible all -m command -a "hostname" //查看所有主机的主机名

[root@control ansible]# ansible all -a "date" //command模块是默认模块，可以不写

[root@control ansible]# ansible all -a "ip addr show"

command不支持bash的特性，有些任务无法执行，比如管道、重

定向等，可以换成shell模块。

[root@control ansible]# ansible-doc -l //查看ansible模块

[root@control ansible]# ansible-doc -l | wc -l //统计模块总数

ansible node1 -m shell -a "who | wc -l"

ansible node1 -m shell -a "echo xyz > /opt/xyz"

ansible node1 -m shell -a "ls /opt"

如果想让远程主机批量执行任务，可以先把任务写成脚本

然后利用script模块传递到被控主机并执行

[root@control ansible]# vim test.sh //编写测试脚本

#!/bin/bash

dnf -y install httpd

systemctl restart httpd

[root@control ansible]# ansible all -m script -a "/root/ansible/test.sh" //在

所有主机执行test.sh脚本

[root@control ansible]# ansible node5 -m shell -a "ss -ntulp | grep :80" //检查

command

shell

script

useradd tom

幂等性，多次执行任务产生的结果与只执行1次相同

2^2 2x2 2^3 2x2x2

1^1 1^1^1

**Ansible ad-hoc应用二**

file模块可以创建文件

state=touch是创建文件，state=directory是创建目录

ansible node1 -m file -a "path=/opt/abc02 state=touch" //在node1

主机创建/opt/abc02文件

ansible node1 -m file -a "path=/opt/abc03 state=directory" //在node1

主机创建/opt/abc03目录

下课休息 16:15回