**Ansible管理软件**  
**计算机的真理：简单，高效**

**选择一款配置管理软件总的来说,无外乎从以下几点**

**来权衡利弊**

**– 活跃度(社区活跃度)**

**– 学习成本**

**– 使用成本**

**– 编码语言**

**– 性能**

**– 使用是否广泛**

**配置基本环境：**

启动6台虚拟机

192.168.1.10 ansible

192.168.1.11 web1

192.168.1.12 web2

192.168.1.21 db1

192.168.1.22 db2

192.168.1.33 cache

**对ansible进行免密钥登陆其他机器。**

**使用脚本安装**

**hostnamectl set-hostname ansible**

**vim /etc/sysconfig/network-scripts/ifcfg-eth0**

**---------------------------------------------------------------------------**

**在ansible中**

**Ssh keygen -t rsa**

**for i in $( awk 'NR<=4{print $1}' /etc/hosts ); do ssh-copy-id $i; done**

**----------------------------------------------------------------------------------**

**初学者----学习社区流行软件（资源较多，性能稳定）----->功能简单高效**

**大牛 -----学习许多不流行的（资源较少）**

**考虑：，学习成本，维护成本，使用成本，编码语言（自己熟悉的语言，没有熟悉的语言尽量选择原生的）**

**-----------------------------------------------------------------------------**

**ansible在一组相同的机器中执行一组相同的命令**

**安装方式：源码安装/yum安装**

**Pythpn环境下的源码安装**

**python setup.py build = make**

**python setup.py install =make install**

**在系统所在 的官网下载：**

**https://mirrors.aliyun.com/centos/7/extras/x86\_64/Packages/**

**/7中是7版本通用的，找到相关的包（依赖关系）**

**Ansible安装：（yum方式）**

**yum install epel-release**

**yum install git python2-pip**

**将：**

**将以下rpm包作制作为yum---createrepo .**

**2018-07-26 19-01-03屏幕截图**

**yum install ansible**

**检验：**

**Ansible --version**

**#################################**

**安装好了 Ansible后就可以开始一些简单的任务了**

**• Ansible配置文件查找顺序**

**环境变量– 首先检测 ANSIBLE\_CONFIG 变量定义的配置文件**

/etc/ansible/

[root@ansible ansible]# ls

ansible.cfg hosts roles

**配置文件– 其次检查当前目彔下的 ./ansible.cfg 文件**

Vim ansible.cfg #inventory = /etc/ansible/hosts //默认管理机群目录

**家目录– 再次检查当前用户家目彔下 ~/ansible.cfg 文件**

Vim hosts

**/etc– 最后检查 /etc/ansible/ansible.cfg 文件**

• /etc/ansible/ansible.cfg 默认配置文件路径

**################################**

**增加末尾：**

[web]

web[1:2]

[db]

db[1:2]

[web:vars] ---------(对web的分组配置)

ansible\_ssh\_user="root"

ansible\_ssh\_password="1" （在没有设置密钥登陆时时用登陆对应主机）

[other]

192.168.1.33

[app:children] -------(大的组引用小的组)

web

db

##################################

在末尾设置用户与密码，在/etc/hosts中的域名无法免密钥登陆

如果db1，db2域名远程第一次无法登陆（yes/no）------>因此会失败

**验证：**

**清除连接缓存：/etc/.ansible/cp下**

**自定义主机分组：**

**在不同的路径下都可以创建一个新的主机分组文件：**

**按照ansible的查找顺序进行创建**

**在新的路径下面vim ansible.cfg(新做一个.cfg)中间加：**

**[default]**

**Inventry=/var/ooxx**

**Ansible.cfg 更改inventroy=后面写路径**

**在路径下创建 hosts（新建一个hosts内部与前面的完全相同）**

**动态主机**

**• 无限可能**

– Ansible Inventory实际上是包含静态Inventory和劢

态Inventory两部分,静态Inventory指的是在文件

/etc/ansible/hosts中指定的主机和组,Dynamic

Inventory指通过外部脚本获取主机列表,并按照

ansible 所要求的格式返回给ansilbe命令的。

**• json**

– JSON的全称是”JavaScript Object Notation”,意

思是JavaScript对象表示法,它是一种基亍文本,独立

于语言的轻量级**数据交换格式。**

**常用的格式：**

1. **hash对2.列表**

**编写一个脚本文件 aaa.sh**

**Echo ‘{**

**"aa" : {**

**"hosts" : ["192.168.1.11", "192.168.1.12"],**

**"vars" : {**

**"ansible\_ssh\_user" : "root",**

**"ansible\_ssh\_pass" : "pwd"**

**}**

**},**

**"bb" : ["192.168.1.15", "192.168.1.16"],**

**"192.168.1.13": { "ansible\_ssh\_user" : "root",**

**"ansible\_ssh\_pass" : "pwd"}**

**}’**

**将ansible------> shell脚本可以进行分组**

**Ansible操作**

列出要执行的主机,不执行任何操作

– ansible all --list-hosts

– ansible all -m ping • 批量检测主机

– ansible all -m command -a 'id' -k • 批量执行命令

e.g

ansible all -m command -a 'free' -k ----->可以执行批量命令（创建，查看，删除）

**• 给所有主机部署公钥**

**在/root/.ssh/下**

– ansible all -m authorized\_key -a "user=root exclusive=true（强行覆盖） manage\_dir=true（若文件夹目录不存在则部署一个）key='$(< /root/.ssh/authorized\_keys)'"（将key完整的读取出来） -k（需要密钥时询问） -v

注意：在第一次输入的时候会要求ssh输入yes---->报错"msg": "Using a SSH password

解决办法：

– 修改 ansible.cfg

host\_key\_checking = False

**Ansible -doc 模块手册**

Ansible -doc -l 列出所有模块------1378个

学会查找模块以及用法：

Ping 模块：测试ssh是否能够登陆

**Command模块：**

Bash 才能实现> | &等命令，而在command 中直接进行命令而没开启bash进程 command 执行的是硬盘上可以执行的文件 与bash相关的就不能执行

如：

[root@ansible ~]# ansible db1 -m command -a 'set'

db1 | FAILED | rc=2 >>

[Errno 2] 没有那个文件或目录

##################################

– 该模块通过-a跟上要执行的命令可以直接执行,丌过

命令里如果有带有如下字符部分则执行丌成功

– "<", ">", "|", "&"

– 该模块丌吭劢 shell 直接在 ssh 迚程中执行,所有使用

到 shell 特性的命令执行都会失败

**Shell/row模块**

– shell 模块用法基本和command一样,区别是 shell模

块是通过/bin/sh迚行执行命令,可以执行任意命令

– raw模块,用法和shell 模块一样 ,可以执行任意命令

– 区别是 raw 没有chdir、creates、removes参数

总结：shell最强，command默认，row最兼容

补充：chage改变用户密码有效期的信息

chage -d 0 zhang3

**script模块**

– 复杂命令怎么办?

– ansible 要上天

– 直接在本地写脚本,然后使用 script 模块批量执行

– ansible t1 -m script -a 'urscript'

**远程执行文件**

**可以写脚本，给各个主机批量执行**

如：vim user.sh (判断是否有zhang3无则创建li4)

#!/bin/bash

id zhang3

if [ $? != 0 ];then

useradd li4

echo 123 | passwd --stdin li4

fi

执行语句：ansible app -m script -a 'a.sh'

**copy模块**

ansible web -m copy -a 'src=/etc/dns.conf dest=/etc/dns.conf'

Src=源 dest=目标地址

适合用于长的一样的配置文件

**Lineinfile| replace模块**

类似于sed的替换方式：用于有个性的配置文件，正则用法与sed完全一致

ansible db1 -m lineinfile -a 'path=/etc/sysconfig/network-scripts/ifcfg-eth0 regexp="^ONBOOT" line="ONBOOT=\"no\""'

Regexp匹配到了这一行的内容，line将整行删掉重新写

**Replace模块：**将匹配到的内容重新写入

ansible db1 -m replace -a 'path=/etc/sysconfig/network-scripts/ifcfg-eth0 regexp="^(ONBOOT=).\*" replace="\1\"yes\""'

**Yum模块**

– 使用yum包管理器来管理软件包

– config\_file:yum的配置文件

– disable\_gpg\_check:关闭gpg\_check

– disablerepo:不用某个源

– enablerepo:启用某个源

– name:要进行操作的软件包的名字,也可以传递一个url或者一个本地的rpm包的路径

– state:状态(present,absent,latest)

– 删除软件包

ansible t1 -m yum -a 'name="lrzsz" state=absent'

– 删除多个软件包

ansible t1 -m yum -a 'name="lrzsz,lftp" state=absent'

– 安装软件包

ansible t1 -m yum -a 'name="lrzsz"'

– 安装多个软件包

ansible t1 -m yum -a 'name="lrzsz,lftp"'

setup模块 set uname -a 批量统计信息 加筛选

– 主要用于获取主机信息,在playbooks里经常会用到的

一个参数gather\_facts就不该模块相关。setup模块下

经常使用的一个参数是filter参数

– filter 可以过滤到我们需要的信息

ansible t1 -m setup -a 'filter=ansible\_distribution'