## ****Ansible****

前言ansilbe 优适用于集群化的系统运维，配合k8s实现从系统到服务的一键式部署与管理。

# 安装

* 1. **Yum安装**

|  |
| --- |
| * yum install http://mirrors.163.com/centos/7.4.1708/extras/x86\_64/Packages/epel-release-7-9.noarch.rpm * yum install ansible -y |

* 1. **配置**

**vim /etc/hosts #增加**

|  |
| --- |
| 192.168.193.63 k8s-master  192.168.193.65 k8s-node1  192.168.193.66 k8s-node2 |

**vim /etc/ansible/ansible.cfg #打开注释**

|  |
| --- |
| host\_key\_checking = False |

**vim /etc/ansible/hosts #增加**

|  |
| --- |
| [k8s]  k8s-master ansible\_ssh\_user="root" ansible\_ssh\_pass="eccom"  k8s-node1 ansible\_ssh\_user="root" ansible\_ssh\_pass="eccom"  k8s-node2 ansible\_ssh\_user="root" ansible\_ssh\_pass="eccom" |

* 1. **验证**

**ansible k8s -m ping**

|  |
| --- |
| [root@localhost etc]# ansible k8s -m ping  k8s-master | SUCCESS => {  "changed": false,  "ping": "pong"  }  k8s-node1 | SUCCESS => {  "changed": false,  "ping": "pong"  }  k8s-node2 | SUCCESS => {  "changed": false,  "ping": "pong"  } |

# 功能

* 1. **特点**
* 部署简单，只需在主控端部署 Ansible 环境，被控端无需做任何操作。
* 默认使用 SSH（Secure Shell）协议对设备进行管理。
* 主从集中化管理。
* 配置简单、功能强大、扩展性强。
* 支持 API 及自定义模块，可通过 Python 轻松扩展。
* 通过 Playbooks 来定制强大的配置、状态管理。
* 对云计算平台、大数据都有很好的支持。
* 提供一个功能强大、操作性强的 Web 管理界面和 REST API 接口 —- AWX 平台。
  1. **常用命令**
     1. **ansible-doc**

ansible-doc -l 列出支持模块

ansible-doc ping 查看ping模块用法

**PS ：以下Client 是配置文件定义的组**

**学习定义组可以参照/etc/ansible/hosts 中的例子，相当于集合，便于批量操作**

* + 1. **script**

script 的功能是在远程主机执行主控端存储的 shell 脚本文件，相当于 scp + shell 组合。

shell > ansible Client -m script -a "/home/test.sh 12 34" # 远程执行本地脚本

* + 1. **shell**

shell模块基本和command相同，但是shell支持管道符

shell > ansible Client -m shell -a "/home/test.sh" # 执行远程脚本

* + 1. **copy**

实现主控端向目标主机拷贝文件，类似于 scp 功能

shell > ansible Client -m copy -a "src=/home/test.sh dest=/tmp/ owner=root group=root mode=0755" # 向 Client 组中主机拷贝 test.sh 到 /tmp 下，属主、组为 root ，权限为 0755

* + 1. **stat**

获取远程文件状态信息，atime/ctime/mtime/md5/uid/gid 等信息

shell > ansible Client -m stat -a "path=/etc/syctl.conf"

* + 1. **get\_url**

实现在远程主机下载指定 URL 到本地，支持 sha256sum 文件校验

shell > ansible Client -m get\_url -a "url=http://www.baidu.com dest=/tmp/index.html mode=0440 force=yes"

* + 1. **yum**

软件包管理

shell > ansible Client -m yum -a "name=curl state=latest"

* + 1. **corn**

远程主机 crontab 配置

shell > ansible Client -m cron -a "name='check dirs' hour='5,2' job='ls -alh > /dev/null'"

* + 1. **mount**

远程主机分区挂载

shell > ansible Client -m mount -a "name=/mnt/data src=/dev/sd0 fstype=ext4 opts=ro state=present"

* + 1. **service**

远程主机系统服务管理

shell > ansible Client -m service -a "name=nginx state=stoped"

shell > ansible Client -m service -a "name=nginx state=restarted"

shell > ansible Client -m service -a "name=nginx state=reloaded"

* + 1. **user**

远程主机用户管理

shell > ansible Client -m user -a "name=wang comment='user wang'"

shell > ansible Client -m user -a "name=wang state=absent remove=yes" # 添加删除用户

* 1. **单命令模式ansible Ad-Hoc**

用法 ansible <hostname pattern> <options>

|  |  |
| --- | --- |
| 常用Option | describe |
| -f NUM | 并发线程数 默认5 |
| -m NAME | 模块 如 ping yum |
| -a ‘args’ | ansible apps -a ‘df -lh’’ |
| -K | 用户密码sudo时使用 |
| -t dir | 输出信息到dir下 |
| -T NUM | 远程最大超时间 |
| -B NUM | 超过NUM后终止正在执行任务 |
| -P NUM | --poll 定期返回后台执行进度 |
| -u username | 用户名 |
| --list | 列出符合条件的主机列表 |
| -l ~REGEX | 正则运行指定主机 |
| -l SUBSET | 运行指定主机 |

更多见：<https://docs.ansible.com/ansible/latest/user_guide/intro_adhoc.html>

* 1. **批量命令模式-剧本 ansible playbook**
* 基于yaml格式 .yml ,
* 剧本执行出错时自动停止
* 系统现有状态与剧本内容一致时默认跳过
* 限制执行剧本对象
* 自定义组下的管理变量替代hosts 中的group var
* ansible-vault 加密解密机制
* Block块编程，增强脚本的逻辑与异常处理
* Include封装执行过程，以供重复调用
  1. **项目级命令模式Ansible roles** 
     1. **简介**

roles作为一个部署任务，其下定义多个子任务role，role是task文件、变量文件、handlers文件的集合体，特点是：具有较好的可移植性可重复执行性。主playbook文件通过调用roles目录下的role，来实现各种灵活多变的部署需，即将运维过程项目化，模块化。

* + 1. **创建部署任务及结构介绍**

ansible-galaxy init roles\_demo

用ansible-galaxy init 构建部署任务目录roles\_demo ,目录如下

* tasks : 用于存放role\_A的主要任务，也可以添加其他task文件，供main.yaml调用，从而实现更加复杂的部署功能。
* handlers : 用于存放触发执行（ hanlders ）的任务。
* defaults : 用于存放默认变量，优先级最低。
* vars : 用于存放变量文件，role\_A中任务和模版里用到的变量可以在这里定义。
* files ：用于存放需要拷贝到目的主机的文件，例如，作为「copy」模块src参数的默认根目录。
* template : 用于存放模版文件，格式为.j2，文件内容要符合Jinja2语法规则，通常使用「template」模块部署服务的配置文件。
* meta : 用于存放role依赖列表，这个知识点后面会详细阐述。
* tests : 用于存放测试role本身功能的playbook和主机定义文件，在开发测试阶段比较常用。
  + 1. **Role的编写**

比较常用的方法，我们可以使用「roles:」语句引用role ：

|  |
| --- |
| ---  - hosts: node1  roles:  - role\_A |

或者

|  |
| --- |
| ---  - hosts: node1  roles:  - name: role\_A  - name: role\_A |

或者

|  |
| --- |
| ---  - hosts: node1  roles:  - role: role\_A  - role: role\_A |

或者使用绝对路径：

|  |
| --- |
| ---  # playbooks/test.yaml- hosts: node1  roles:  - role: /root/lab-ansible/roles/role\_A |

引入的同时添加变量参数：

|  |
| --- |
| ---  # playbooks/test.yaml  - hosts: node1  roles:  - role: role\_A  vars:  name: Maurice  age: 100 |

引入的同时添加tag参数：

|  |
| --- |
| ---  # playbooks/test.yaml- hosts: node1  roles:  - role: role\_B  tags:  - tag\_one  - tag\_two  # 等价于上面  - { role: role\_B, tags:['tag\_one','tag\_two'] } |

**检索路径**

在不使用绝对路径的情况下，ansible检索role的默认路径有：

* 执行ansible-playbook命令时所在的当前目录
* playbook文件所在的目录及playbook文件所在目录的roles目录
* 当前系统用户下的～/.ansible/roles目录
* /usr/share/ansible/roles目录
* ansible.cfg 中「roles\_path」指定的目录，默认值为/etc/ansible/roles目录

更多参考见http://www.ansible.com.cn/docs/playbooks\_roles.html

* + 1. **案例-搭建zookeeper集群**
* 搭建zookeeper集群 附带安装tree与jdkdevel工具
* 创建roles目录 ,图下

|  |
| --- |
| zookeeper  ├── files  │   └── zookeeper-3.4.5.tar.gz  ├── templates  │   └── zoo.cfg.j2  ├── vars  │   └── main.yml  └── zookeeper.yml |

* zookeeper.yml

|  |
| --- |
| - hosts: zookeeper  remote\_user: root  roles:  - zookeeper  tasks:  - name: jps tool install  yum: name={{item}} state=present  with\_items:  - java-1.8.0-openjdk-devel.x86\_64  - tree  - name: mkdir directory for bigdata data  file: dest={{BigdataDir}} mode=0755 state=directory  - name: install zookeeper  unarchive: src={{AnsibleDir}}/zookeeper/files/{{zookeeper\_file\_name}}.tar.gz dest={{BigdataDir}}  - name: install configuration file for zookeeper  template: src={{AnsibleDir}}/zookeeper/templates/zoo.cfg.j2 dest={{BigdataDir}}/{{zookeeper\_file\_name}}/conf/zoo.cfg  - name: create log\data directory  file: dest={{BigdataDir}}/zookeeper/dataLogDir mode=0755 state=directory  file: dest={{BigdataDir}}/zookeeper/data mode=0755 state=directory  - name: add myid file  shell: echo {{ myid }} > {{BigdataDir}}/zookeeper/data/myid  - name: start zookeeper  shell: sh {{BigdataDir}}/{{zookeeper\_file\_name}}/bin/zkServer.sh start  tags:  - start zookeeper |

* Var main.yml

|  |
| --- |
| zk1: 192.168.193.57  zk2: 192.168.193.58  zk3: 192.168.193.59  BigdataDir: /opt/hzgc  AnsibleDir: /etc/ansible/roles  zookeeper\_file\_name: zookeeper-3.4.5 |

* templates zoo.cfg.j2

|  |
| --- |
| tickTime=2000  initLimit=10  syncLimit=5  clientPort=2181  dataLogDir={{BigdataDir}}/zookeeper/dataLogDir  dataDir={{BigdataDir}}/zookeeper/data  clientPort=2181  quorumListenOnAllIPs=true  server.1={{zk1}}:2888:3888  server.2={{zk2}}:2888:3888  server.3={{zk3}}:2888:3888 |

执行任务 ansible-playbook -i /etc/ansible/hosts zookeeper.yml

执行输出

|  |
| --- |
| PLAY [zookeeper] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ok: [node2]  ok: [node3]  ok: [node1]  TASK [jps tool install] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ok: [node3] => (item=[u'java-1.8.0-openjdk-devel.x86\_64', u'tree'])  ok: [node1] => (item=[u'java-1.8.0-openjdk-devel.x86\_64', u'tree'])  ok: [node2] => (item=[u'java-1.8.0-openjdk-devel.x86\_64', u'tree'])  TASK [mkdir directory for bigdata data] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ok: [node1]  ok: [node3]  ok: [node2]  TASK [install zookeeper] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ok: [node1]  ok: [node2]  ok: [node3]  TASK [install configuration file for zookeeper] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ok: [node1]  ok: [node3]  ok: [node2]  TASK [create log\data directory] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  changed: [node1]  changed: [node2]  changed: [node3]  TASK [add myid file] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  changed: [node2]  changed: [node1]  changed: [node3]  TASK [start zookeeper] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  changed: [node1]  changed: [node2]  changed: [node3]  PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  node1 : ok=8 changed=3 unreachable=0 failed=0  node2 : ok=8 changed=3 unreachable=0 failed=0  node3 : ok=8 changed=3 unreachable=0 failed=0 |

验证：目标服务器 jps 验证即可。

可参于https://github.com/nhz94259/Ansible/tree/master/zookeeper直接下载