Ansible

是一款IT自动化和Devops软件,基于Python开发,实现了批量操作,批量程序部署,批量运行命令等

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

选择理由:只需要SSH和Python即可使用

无客户端,模块丰富

上手容易,门槛低

基于Python开发,做二次开发更容易,使用率比较高

Ansible特性模块花设计,调用特定的模块完成特定的任务

基于Python语言实现

Paramiko

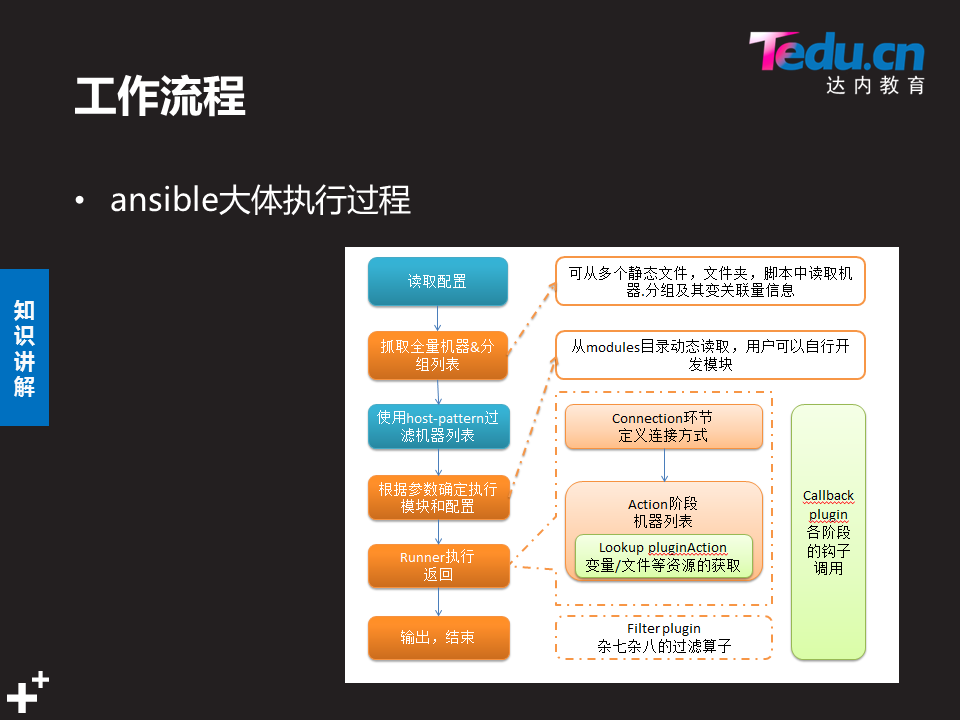
pyYAML (半结构花语言)

Jinja2

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



Ansible安装

软件依赖关系

要求Python2.6或Python2.7版本

安装yum源:

在ftp下创建一个共享目录(ansible),把软件包考入

Createrepo --update /var/ftp/ansible //更新索引文件

创建虚拟机ansible,配置yum源

[ansible]

name=ansible

baseurl="ftp://192.168.1.254/ansible"

enabled=1

gpgcheck=0

~]# yum repolist //多出六个包

~]# ansible --version //在进行验证

在准备其他主机(托管主机)

首先检查huanjing变量定义的配置文件

检查当前目录下的~/ansible.cfg文件

在检查当前目录~/ansible.cfg文件

最后检查/etc/ansible/ansible.cfg主配置文件

/etc/ansible/hosts //配置文件

14 inventory = /etc/ansible/hosts //指定解析文件的路径

61 host\_key\_checking = False

如果为False,不需要输入 yes

如果为True,则等待输入yes

[web]

web1

web2

[db]

db1

db2

[other]

cache

Ansible 主机集合 -m (默认command模块)模块名 -a(or --args) 模块参数

查看多个,可以使用逗号分割

-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 显示显示详细信息

~]# ansible web --list-hosts //显示web集合里的主机

~]# ansible all --list //显示定义的所有主机

~]# ansible web -m ping -k //批量检测主机,部署密码会报错

SSH password:

web1 | SUCCESS => {

"changed": false,

"ping": "pong"

}

web2 | SUCCESS => {

"changed": false,

"ping": "pong"

部署证书文件:

创建一对密钥对

~]# cd /root/.ssh/

]# ssh-keygen -t rsa -b 2048 -N "" -f key

给所有主机部署秘钥

]# ssh-copy-id -i key.pud 主机名(ansible web(1-2) db(1-2))

]# ansible web -m ping //会报错,只支持id\_rsa

1.

]# mv id\_rsa key //修改名字就行

1. 修改扩展参数

对全部的主机定义:

[all:vars] ///表示定义的是变量

ansible\_ssh\_private\_key\_file=/root/.ssh/key(key的绝对路径)

对单个主机定义:

若ssh端口不是默认的

在定义的变量主机名后面加上

ansible\_ssh\_port=23

就行了

Ansible web -m ping

就不会出错了

子主定义:

[app:children] //宣布web,db,other为变量名,而不是主机名

web

db

Other

Ping一下能不能同

Ansible app -m ping

自定义ansible

创建自定文件

mkdir ooxx

cd ooxx/

vim ansible.cfg

[defaults]

inventory = /root/ooxx/myhost

host\_key\_checking = False

vim myhost

[web]

db1

web2

[db]

db2

web1

[xx]

cache

ansible web --list

若是在当前路径下,生效的是当前路径下的文件

退出后则生效的是默认的文件

批量配置管理:

模块:

~]# ansible-doc -l //列出所有的模块

~]# ansible-doc modulename //查看帮助

~]# ansible-doc 模块名 //查看哪一个模块的用法

Command:

默认模块,远程执行命令:

~]# ansible all -m command -a 'uptime' //查看所有机器的负载

要通过-a跟上要执行的命令可以直接执行,若有 < > | & 字符则不能执行,

该模块不能解析系统变量 ,不启动shell直接在ssh中执行,所有用到shell命令就会失败

Shell模块

用法和command一样,但是不能执行交互式的命令如 vim top等

~]# ansible all -m shell -a 'uptime' //查看所有机器的负载

~]# ansible web -m shell -a "echo ${HOSTNAME}" (显示的是目标主机的主机名)

web2 | SUCCESS | rc=0 >>

ansible

web1 | SUCCESS | rc=0 >>

ansible

~]# ansible web -m shell -a 'echo ${HOSTNAME}' (显示的是本地的主机名)

web1 | SUCCESS | rc=0 >>

web1

web2 | SUCCESS | rc=0 >>

web2

Sctpt模块:

命令太复杂,在本地写脚本,使用sctipt模块执行

列子:

给主机添加用户hh

若有nb用户则没有hh用户

设置hh密码为456

先写脚本:

#!/bin/bash

id nb

if [ $? != 0 ];then

useradd hh

echo 456 | passwd --stdin hh

fi

ansible all -m script -a './hh.sh' //就相当于script模块把脚本带到每一台主机去运行

Yum模块:

Name : 要进行操作的软件包名字

State : 动作用逗号分割(installed(安装), removed(卸载))

列子:

~]# ansible web1 -m yum -a 'name=httpd state=installed' //安装httpd服务

Service模块

Name:服务名称

Enabled:是否开机启动 yes| on

Sleep : 执行restarted,会在stop和start之间谁几秒

State : 对当前服务执行(restarted , stopped . Started , reloaded)等操作(要加ed)

列子:

Copy模块:

复制文件到远程主机

Src : 本地路径(可以绝对路径和相对路径)

Dest : 目标主机 (必须是绝对路径)

Backup : 覆盖覆盖原来的源文件 yes|no

Force : 若目标主机有这个文件,内容不同,如果是yes,则强制覆盖,若为no则只有目标主机文件不存在时才复制,默认为yes

列子:

Lineinfile模块:

Path 目标文件

Regexp :正则表达式,要修改的行

Line :最终修改的结果

Replace模块:

Path 目标文件

Regexp :正则表达式,要修改的行

Replace :替换后的结果

(区别在于lineinfile匹配到了一个字就整行干掉

Replace是匹配到几个字就替换几个字)

Setup模块:

主要用于获取主机信息,playbook是里经常会用到另一个参数gather\_facts与该模块相关,

这个模式下用的是filter参数