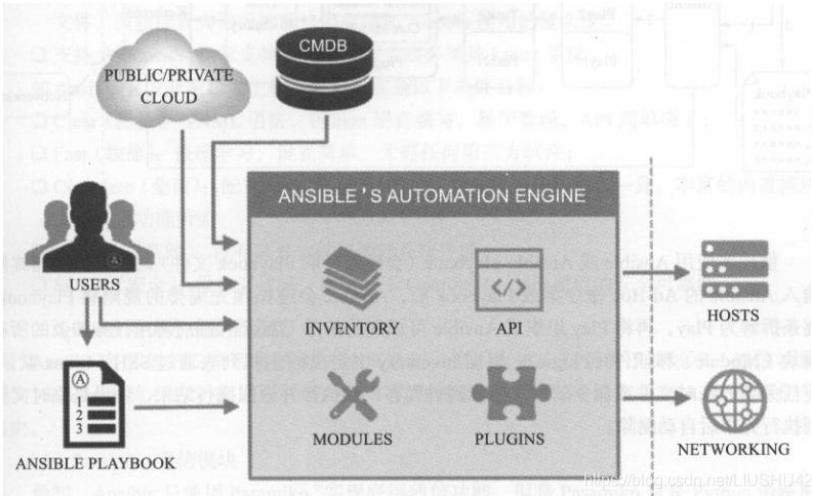
第一章 介绍 Ansible

Ansible 一款自动化运维的工具，基于 Python 开发，实现批量系统配置，批量程序部署，批量执行命令

1 用户管理 权限权限 磁盘管理 定时任务
2 丰富的内置模块 ansible-doc -l
3 agentless 无客户端 ，只需要控制节点安装 Ansible 引擎
4 安全，基于 OpenSSH
5 支持自定义模块

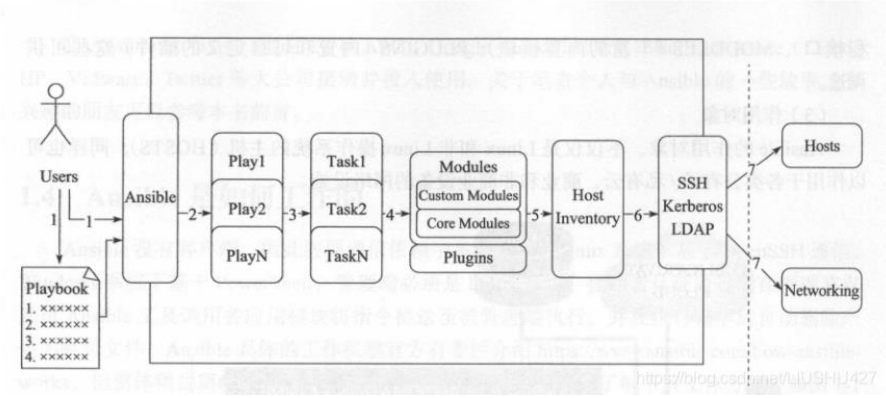
ansible的工作机制



1 CMDB 配置管理数据库 管理着企业 IT 架构的各项配置信息 运维人员结合 CMDB 和 Ansible，通过 CMDB 直接下发指令调用 Ansible
2 Public/Private 提供了丰富 API 接口
3 Users 运维人员
4 Ansible Playbook 任务剧本，格式 YML 文件，例如：安装配置 Apache
5 核心 Ansible 引擎
6 Inventory 清单 主机清单 /etc/ansible/hosts 静态清单 动态清单
7 Core Modules Ansible 引擎自带模块
8 Plugins 辅助模块，模块补充 连接类型插件 变量插件
9 API：供第三方程序调用的接口
10 Host Windows Linux Networking

利用 Ansible 来实现自动化有两种方式:
1 AD-HOC 指 Ansible 命令 主要用于一次性任务
2 ansible-playbook 大型项目

工作原理：
一台控制节点(workstaiton)需要在控制节点上安装 Ansbile 引擎，根据定义 Inventory host 或 Playbook,调用特定模块通过 openssh 协议在远程主机执行相关命令并返回结果




```
[student@workstation ~]$ ansible --version
ansible 2.8.0
  config file = /etc/ansible/ansible.cfg    主配置文件
  configured module search path =
['/home/student/.ansible/plugins/modules',
'/usr/share/ansible/plugins/modules']
  ansible python module location =
/usr/lib/python3.6/site-packages/ansible
  executable location = /usr/bin/ansible
  python version = 3.6.8 (default, Apr  3 2019, 17:26:03) [GCC 8.2.1
20180905 (Red Hat 8.2.1-3)]

[student@workstation ~]$ ansible-doc  -l |wc  -l   查看模块帮助
[student@workstation ~]$ ansible-playbook   读取 playbook 文件
[student@workstation ~]$ ansible-galaxy   安装和创建角色
[student@workstation ~]$ ansible-vault     文本加密

[student@workstation ~]$ ll /etc/ansible/hosts    默认清单配置文件
-rw-r--r--. 1 root root 1016 May 17  2019 /etc/ansible/hosts
```

第二章 部署 Ansible
Ansible 主配置文件优先顺序：

/etc/ansible/ansible.cfg 主配置文件，优先级最低

~/.ansible.cfg 在用户家目录下创建 Ansible 配置文件，覆盖全局配置文件

./ansible.cfg 如果执行 ansible 命令目录中存在 ansible 配置文件，使用它，优先级最高 推荐方式

[defaults] 默认值
inventory = /home/student/playbooks/inventory 清单文件路径
forks = 5 并发连接数
sudo_user = root 提权到 root 用户
roles_path = /etc/ansible/roles 定义角色路径
remote_user = ansible 控制节点连接到被管理节点使用 ansible 用户
[privilege_escalation]
become=True 允许提权
become_method=sudo 提供方式
become_user=root 提供哪个用户
become_ask_pass=False 不需要输入密码

清单文件：
1 静态清单
2 动态清单
静态文件：

```
[student@workstation ~]$ mkdir playbooks
[student@workstation ~]$ cd playbooks
[student@workstation playbooks]$ vim inventory
servera.lab.example.com
serverb.lab.example.com
serverc.lab.example.com
serverd.lab.example.com
192.168.1.1
192.168.1.2
192.168.1.3

[webserver] 定义分组
servera.lab.example.com
serverb.lab.example.com

[dbserver] 定义分组
serverc.lab.example.com
```

```
serverd.lab.example.com

[server:children]  定义嵌套组
webserver
dbserver
```

默认分组：
1 all 清单文件中列出每一个主机
2 ungrouped 清单主机没有定义分组

简化主机的定义：

```
[start:end]
192.168.1.[1:100]
db[01:20].lab.example.com
[a:c].cloudshelledu.com


[student@workstation playbooks]$ cat ansible.cfg
[defaults]

inventory  =  /home/student/playbooks/inventory 定义清单文件绝对路径
remote_user = devops

[privilege_escalation]
become=True
become_method=sudo
become_user=root
become_ask_pass=False

[student@workstation playbooks]$ ansible ungrouped --list-hosts
```

ansible [清单] -list-hosts -i inventory

```
[student@workstation playbooks]$ ansible webserver --list-hosts
  hosts (2):
    servera.lab.example.com
    serverb.lab.example.com
[student@workstation playbooks]$ ansible dbserver --list-hosts
  hosts (2):
    serverc.lab.example.com
    serverd.lab.example.com
[student@workstation playbooks]$ ansible server --list-hosts
```

```
  hosts (4):
    servera.lab.example.com
    serverb.lab.example.com
    serverc.lab.example.com
    serverd.lab.example.com
[student@workstation playbooks]$ ansible all --list-hosts
  hosts (14):
    192.168.1.1
    192.168.1.2
    192.168.1.3
    192.168.1.4
    192.168.1.5
    192.168.1.6
    192.168.1.7
    192.168.1.8
    192.168.1.9
    192.168.1.10
    servera.lab.example.com
    serverb.lab.example.com
    serverc.lab.example.com
    serverd.lab.example.com
[student@workstation playbooks]$ ansible ungrouped --list-hosts
  hosts (10):
    192.168.1.1
    192.168.1.2
    192.168.1.3
    192.168.1.4
    192.168.1.5
    192.168.1.6
    192.168.1.7
    192.168.1.8
    192.168.1.9
    192.168.1.10
```

运行临时命令：

```
$ ansible host-pattern  -m module [-a  'module arguments' ] [-i
inventory]
```

-m setup

文件管理:copy file lineinfile 软件包：yum dnf 网络：nmcli get_url

-a 参数 通常情况下模块都是都是需要参数

-i 指定清单文件

```
[student@workstation playbooks]$ ansible -m user -a
"name=cloudshelluser uid=2000 shell=/sbin/nologin" server
```

```
[student@workstation playbooks]$ ansible -m command  -a 'id
cloudshelluser' server
serverc.lab.example.com | CHANGED | rc=0 >>
uid=2000(cloudshelluser) gid=2000(cloudshelluser)
groups=2000(cloudshelluser)
servera.lab.example.com | CHANGED | rc=0 >>
uid=2000(cloudshelluser) gid=2000(cloudshelluser)
groups=2000(cloudshelluser)
serverd.lab.example.com | CHANGED | rc=0 >>
uid=2000(cloudshelluser) gid=2000(cloudshelluser)
groups=2000(cloudshelluser)
serverb.lab.example.com | CHANGED | rc=0 >>
uid=2000(cloudshelluser) gid=2000(cloudshelluser)
groups=2000(cloudshelluser)
```

LAB:配置 YUM 客户端

Create a shell script called /home/student/ansible/adhoc.sh that runs ans Ansible ad-hoc command to create a yum repository on servera.lab.example.com The name of the repository is CLOUDSHELL The Description is BaseOS Software The bash URL is http://content.example.com/rhel8.0/x86_64/dvd/BaseOS The GPG key URL is http://content.example.com/rhel/RPM-GPG-KEY-redhat-release The repository is enabled

第三章 实施 PLAYBOOK

Playbook 文本文件 YAML 格式 扩展名 xxx.yml

方法 1：

```
[student@workstation playbooks]$ ansible -m user -a
"name=cloudshelluser uid=2000 shell=/sbin/nologin" server
```

方法 2 ：

```
[student@workstation playbooks]$ vim createuser.yml
---    以破折号开始
- name: First Play 第一个 Play 开始以破折号加空格一个 Play 包含多个任务
  hosts:  server
  tasks:
      -  name: newuser cloudshelluser
         user:
               name: cloudshelluser
               uid:  2000
               shell: /sbin/nologin
```

剧本包含一个 play,这个 play 包含一个任务

```
- name: Second Play
  hosts: server
  tasks:
    - name: Install Apache Package
      yum:
          name: httpd
          state: latest
```

Play: 三个键值组合
name hosts tasks 三个键值对齐

排错方法:
方法 1
[student@workstation playbooks]$ ansible-playbook --syntax-check installweb.yml

playbook: installweb.yml

方法 2:
[student@workstation playbooks]$ ansible-playbook -C installweb.yml
-C 空运行 不会实际修改被管理主机

方法 3:
-v 显示任务结果
-vv 显示任务结果和任务配置
-vvv 显示连接信息
-vvvv 执行过程

方法 4:
定义 Logfile

安装软件包 提示:ansible-doc yum
- 创建/home/student/ansible/install_packages.yml 的 playbook:
  - 在 servera 中安装 php 和 mariadb 软件包
  - 在 serverb 中安装 Development Tools 包组
  - 在 servera 主机升级所有的软件包

第四章 管理大项目

清单文件：

```
[student@workstation ansible]$ cat inventory
servera.lab.example.com
serverb.lab.example.com
serverc.lab.example.com
serverd.lab.example.com

[webserver]
servera.lab.example.com
serverb.lab.example.com

[dbserver]
serverc.lab.example.com
serverd.lab.example.com

---
- hosts: servera.lab.example.com
  become: yes
  tasks:
          - name: install packages
            yum:
                    name: "{{ item }}"
                    state: present


- hosts: servera.lab.example.com

- webserver

- dbserver

- hosts: all

- hosts:  '*'   通配符

- hosts:  '! servera.lab.example.com,webserver'

- hosts:  '*.lab.example.com"

- hosts:  '192.168.1.*'

- hosts:  'webser*'
```

```
- hosts:  'webserver,dbserver,test'

- hosts:  'webserver:dbserver:test'
```

https://github.com/ansible/tree/devel/contrib/inventory 获取动态清单脚本

私有云平台 OpenStack
虚拟化平台 VMware vSphere RHV
PAAS OpenShift Container Platform OCP

Ansible 控制节点 并发连接数量

```
#forks          = 5
```

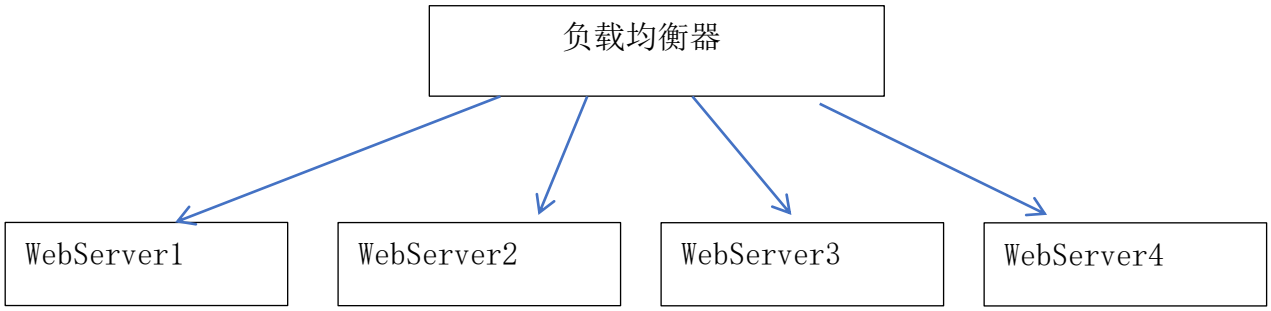
默认 forks 非常保守设置

forks = 60 ~ 100

```
---
- hosts: all    假如 all 有十台主机
  become: yes
  tasks:
          - name: install packages
            yum:
                    name: "{{ item }}"
                    state: present
            loop:
                    - php
                    - mariadb

          - name: update all pcakges
            yum:
                    name: "*"
                    state: latest

[student@workstation ansible]$ ansible-playbook  -f 10
install_packages.yml
```



如果同时更新四台 WebServer,导致服务不可用

滚动更新

```
---
- hosts: all
  become: yes
  serial: 2
  tasks:
          - name: install packages
            yum:
                    name: "{{ item }}"
                    state: present
            loop:
                    - php
                    - mariadb

          - name: update all pcakges
            yum:
                    name: "*"
                    state: latest
```

第五章 利用角色简化 PLAYBOOK

```
[student@workstation ansible]$ ansible-galaxy  list  查看角色
# /usr/share/ansible/roles
- linux-system-roles.kdump, (unknown version)
- linux-system-roles.network, (unknown version)
- linux-system-roles.postfix, (unknown version)
- linux-system-roles.selinux, (unknown version)
- linux-system-roles.timesync, (unknown version)
```

使用 Ansible Galaxy 部署角色

https://galaxy.ansible.com 公共资源库

```
[student@workstation ansible]$ mkdir roles/
[student@workstation ansible]$ cd roles/
[student@workstation roles]$ ansible-galaxy  init apache    创建角色


[student@workstation roles]$ ansible-galaxy install -r roles/xxx.yml
-p roles

[student@workstation ansible]$ cat installroles.yml
---
- src: http://materials.example.com/haproxy.tar
  name: haproxy

- src: http://materials.example.com/phpinfo.tar
  name: phpinfo
[student@workstation ansible]$ ansible-galaxy install -r
installroles.yml -p roles/
- downloading role from http://materials.example.com/haproxy.tar
- extracting haproxy to /home/student/ansible/roles/haproxy
- haproxy was installed successfully
- downloading role from http://materials.example.com/phpinfo.tar
- extracting phpinfo to /home/student/ansible/roles/phpinfo
- phpinfo was installed successfully
[student@workstation ansible]$ ll roles/
total 0
drwxrwxr-x. 10 student student 135 Sep  8 20:51 apache
drwxrwxr-x.  9 student student 122 Sep  8 21:04 haproxy
drwxrwxr-x.  9 student student 122 Sep  8 21:04 phpinfo
```

第六章 对 ANSIBLE 进行故障排除

1 Ansible 日志文件

```
[defaults]
inventory=/home/student/ansible/inventory

remote_user = devops

roles_path = /usr/share/ansible/roles:/home/student/ansible/roles

log_path = /home/student/ansible/ansible.log
```

调试模块： debug
常用参数：
msg：调试输出信息

var：将某个任务执行的输出作为变量传递给 debug 模块

管理错误：
ansible-playbook xxxx.yml －syntax-check

--step 调试 playbook 一次执行一个任务

ansible-playbook xxxx.yml
[student@workstation ansible]$ ansible-playbook installweb.yml --step

PLAY [all]
****
Perform task: TASK: Gathering Facts (N)o/(y)es/(c)ontinue: y

Perform task: TASK: Gathering Facts (N)o/(y)es/(c)ontinue:

--start-at-task 从某个特定的任务执行

[student@workstation ansible]$ ansible-playbook install_packages.yml
--start-at-task="update all pcakges"