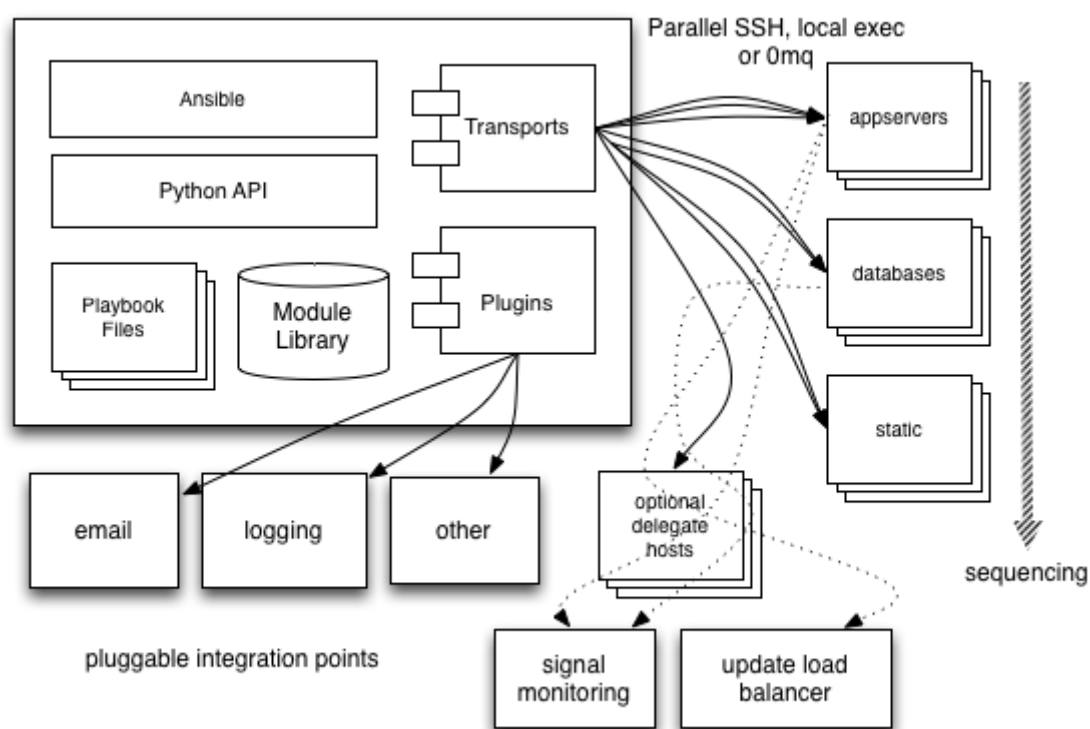


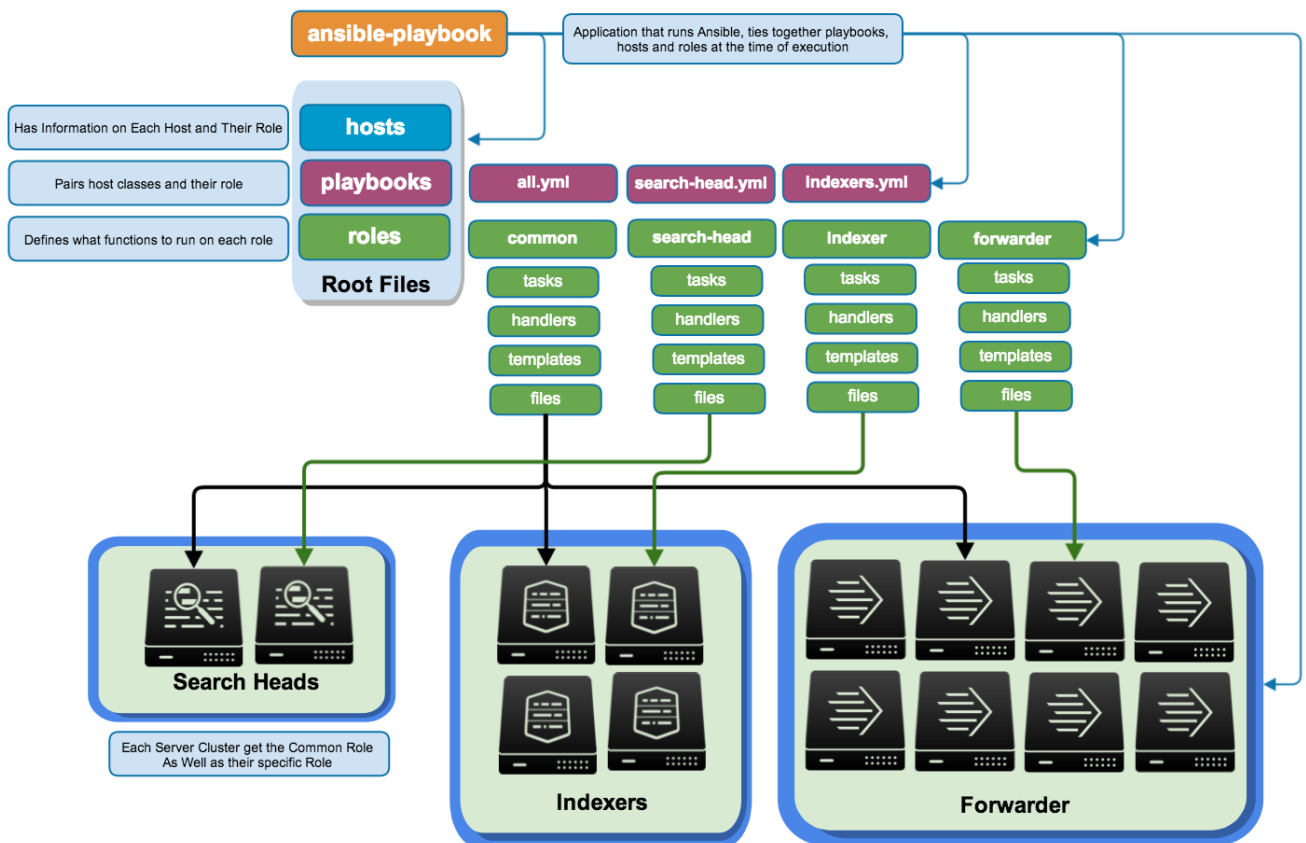
## 什么是Ansible?

HGG • 2017-09-24 23:34 • Linux干货

### 自动化工具——ansible



## ansibleArchitecture



## ansibleProject

### 1. 什么是ansible

ansible是个什么东西呢？官方的title是“Ansible is Simple IT Automation”——简单的自动化IT工具。这个工具的目标有这么几项：让我们自动化部署APP；自动化管理配置项；自动化的持续交付；自动化的（AWS）云服务管理。

所有的这几个目标本质上来说都是在一个台或者几台服务器上，执行一系列的命令而已。就像我之前有介绍过的Fabric，以及我们基于Fabric开发的自动化应用部署的工具：Essay。都是做了这么个事——**批量的在远程服务器上执行命令**。

那么fabric和ansible有什么差别呢？简单来说fabric像是一个工具箱，提供了很多好用的工具，用来在Remote执行命令，而Ansible则是提供了一套简单的流程，你要按照它的流程来做，就能轻松完成任务。这就像是库和框架的关系一样。

当然，它们之间也是有共同点的——都是基于 [paramiko](#) 开发的。这个paramiko是什么呢？它是一个纯Python实现的ssh协议库。因此fabric和ansible还有一个共同点就是不需要在远程主机上安装client/agents，因为它们是基于ssh来和远程主机通讯的。

## 2. 快速安装

上面简单介绍了下这是个什么东西，怎么安装呢？也很简单，因为ansible是python开发的，因此可以这么安装：

```
yum install ansible
# 或者
yum install ansible
```

## 3. 配置

安装完成之后，先来配置下配置项——.ansible.cfg。ansible执行的时候会按照以下顺序查找配置项：

```
* ANSIBLE_CONFIG (环境变量)
* ansible.cfg (当前目录下)
* .ansible.cfg (用户家目录下)
* /etc/ansible/ansible.cfg
```

还有一个重要的配置是hosts的配置，所有的远程主机需要在hosts中配置，可以分组。当然hosts也可以执行是指定。先来一个简单的例子，在家目录下新建一个hosts文件：

```
# hosts
[local]
127.0.0.1
```

然后在终端执行：

```
$ ansible -i ~/hosts all -a 'who'

# 结果如下：
127.0.0.1 | success | rc=0 >>
Guest      console  Feb  1 16:29
the5fire   console   Jan 20 19:50
the5fire   ttys018   Feb 22 15:35 (localhost)
```

这是一条ad-hoc命令——临时执行命令，ad-hoc是ansible里的一个概念, 在上面命令中就是 -a，具体稍后再说。命令中的all是值hoss中的所有服务器，当然也可以通过 `ansible -i ~/hosts local -a 'who'` 这样根据组名指定服务器。

再说到ansible.cfg的配置，默认ansible执行时会从该配置中加载hosts配置，因此可以通过修改ansible.cfg来指定默认的hosts文件地址：

```
# .ansible.cfg
[defaults]
hostfile=/Users/the5fire/hosts
```

这样下次执行，就不需要 -i 参数了。

## 4. Ad-Hoc

ad hoc——临时的，在ansible中是指需要快速执行，并且不需要保存的命令。说白了就是执行简单的命令——一条命令。对于复杂的命令后面会说playbook。

那么这个Ad-Hoc命令怎么用呢？上面已经简单的示范了下。在ansible中还有一个Module（模块）的概念，这个模块可以理解为一个库，所有的命令都需要通过模块来执行，比如上面的那个命令：`ansible -i ~/hosts all -a 'who'`，其实是调用了默认的command模块：`ansible -i ~/hosts all -m command -a 'who'`，除了command模块还有其他很多模块，比如你就想ping下这个服务器是不是还存在可以通过ping模块：`ansible -i ~/hosts all -m ping`。

还有几个参数需要记录下：

```
-u username # 指定ssh连接的用户名
-f 10       # 指定并发数
--sudo [-K] # 如果需要root权限执行的话，-K参数是用来输入root密码的
```

你可以通过各种模块来批量完成某个包的安装，或者其他什么需要的操作。更多模块可以看官网文档: [modules](#)

关于Ad-Hoc的更多内容参考这里: [intro\\_adhoc](#)

## 5. 简单Playbook

上面的ad hoc是指执行一条临时的不需要保存的命令，那么复杂的命令怎么执行呢？因此也就有了playbook这个命令: `ansible-playbook`。

playbook（剧本），顾名思义，就是需要定义一个脚本或者说配置文件，然后定义好做什么。一个简单的playbook是这样的，把当前用户名输出到whoami.rst文件中：

```
# playbook.yml
---
- hosts: local # hosts中指定
  remote_user: the5fire # 如果和当前用户一样，则无需指定
  tasks:
    - name: whoami
      shell: 'whoami > whoami.rst'
```

执行完这个命令后，你可以在local所代表的服务器的用户目录下发现这么个文件。说到这里，要停一下。这个配置文件是yaml格式的，因此你可能需要去了解YAML：[wiki YAML](#)。

简单解释下上面的playbook，hosts后面根据local是从hosts中读取的，tasks是关键词，指明了要执行哪些任务；下面的name是任务的名称，shell是前面提到的module(模块)，单引号中是命令。

除了tasks之外，还有一个handlers的命令，handlers是在执行tasks之后服务器发生变化之后可供调用的handler，使用起来如下：

```
# playbook.yml
---
- hosts: local # hosts中指定
  remote_user: the5fire # 如果和当前用户一样，则无需指定
  tasks:
    - name: whoami
      copy: src=~/.hosts dest=~/.hosts.dest # 本地拷贝到远端
      notify: # 如果copy执行完之后~/.hosts.dest文件发送了变化，则执行
        - clear copy # 调用handler
  handlers:
    - name: clear copy
      shell: 'mv ~/.hosts.dest hosts.del' # 假装删除
```

上面只是一个演示，再来一个真实的功能——在local服务器上，从git上clone下来我的blog源码，然后创建虚拟环境，创建数据库，最后运行：

```
# deploy-blog-simple.yml
---
- hosts: local # hosts中指定
```

```
remote_user: the5fire # 如果和当前用户一样,则无需指定
tasks:
  - name: check out django_blog
    git: dest=~/.demos/django_selfblog repo=https://github.com/the5fire/django_se
      update=yes
  - name: make virtualenv
    shell: 'virtualenv ~/.demos/'
  - name: install requirements
    pip: requirements=~/.demos/django_selfblog/requirements.txt
      virtualenv=~/.demos/
  - name: init database
    shell: . ./bin/activate && cd django_selfblog/selfblog && ./init_database.sh
  - name: run manage.py
    shell: . ./bin/activate && cd django_selfblog/selfblog && ./run.sh chdir=~/
```

如果你已经配置好ssh账户免密码登录之后,直接执行: `ansible-playbook deploy-blog-simple.yml` 就可以在你指定的服务器上部署,并启动blog了。

本文来自投稿,不代表Linux运维部落立场,如若转载,请注明出处: <http://www.178linux.com/87557>

# ansible

# 自动化运维

赞 (11)

MYSQL高级运用-MHA(提供主从复制高可用,主节点故障时,进行故障转移)

上一篇

2017-09-24 21:20

数据结构知识点 (list,tuple,冒泡法)

2017-09-25 09:41

下一篇

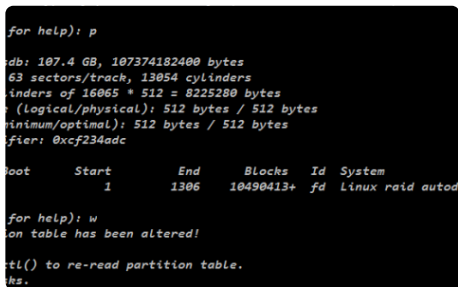


# Python全能自动化开发实战课程

学后薪资平均提升30%，最高年薪30万+



## 相关推荐



http和apache服务器

2017-12-05

## 文件管理权限

命令和笔记

Linux干货 2017-12-03

## N25-第二周作业

1、linux上的文件管理类命令都有哪些，其常用的使用方法及其相关示例演示 linux上处理目录的常用命令：  
ls：列出目录 cd：切换目录 pwd：显示当前工作路径 &nbs...

Linux干货 2016-12-13



数据库的历史及常见基本功能

2018-01-29

## 第八周作业

第八周作业 1、写一个脚本，使用ping命令推测172.16.250.1—172.16.250.254之间的所有主机的在线状态； 在线的主机使用绿色显示； 不在线的主机使用红色显示；  
#!/bin/bash for i in {1..254};do if ping -c 1 -w 1 192.168.1.\$i

Linux干货 2017-11-13

## Nginx+Keepalived实现站点高可用

Nginx+Keepalived实现站点高可用 vrrp 虚拟路由冗余协议(Virtual Router Redundancy Protocol，简称VRRP)是由 IETF提出的解决局域网中配置静态网关出现单点失效现象的路由协议，1998年已推出正式的RFC2338协议标准。

Linux干货 2016-11-07

## 评论列表（1条）



非常柠檬

2017-11-16 12:32

通过博主介绍 充分了解了!