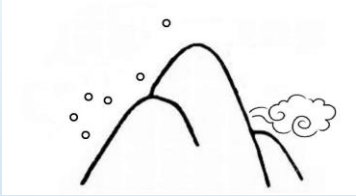
Ansible Playbooks



千易雲(北京)教育科技有限公司

简介

Playbooks是Ansible的配置，部署，编排语言。他们可以被描述为一个需要希望远程主机执行命令的方案，或者一组IT程序运行的命令集合。

如果 Ansible 模块你是工作室中的工具，那么 playbooks 就是你设置的方案计划。

在基础层面， playbooks 可以被用来管理用于部署到远程主机的配置文件.在更高的层面上，playbooks可以依次对多层式架构上的服务器执行上线包括滚动更新在内的操作并可以将操作委托给其他主机包括在此过程中发生的与监视服务器，负载均衡服务器的交互操作在内。

虽然这里讲发很多，但是不需要立刻一次性全部学完。你可以从小功能开始，当你需要的时候再来这里找对应的功能即可。

Playbooks被设计的非常简单易懂和基于text language二次开发。有多种办法来组织playbooks和其附属的文件，同时我们也会提供一些关于学习Ansible的建议。

Playbook基础

- 主机与用户

  你可以为playbook中的每一个play，个别地选择操作的目标机器是哪些。以哪个用户身份去完成要执行的步骤（called tasks）。
  hosts 行的内容是一个或多个组或主机，以逗号为分隔符。
  *remote_user*就是账户名：

```
---
- hosts: webservers
  remote_user: root
```

Playbook基础

- 主机与用户

再者，在每一个task中，可以定义自己的远程用户：

```
---
- hosts: webservers
  remote_user: root
  tasks:
     - name: test connection
       ping:
       remote_user: yourname
```

Playbook基础

- 主机与用户

  也支持从sudo执行命令：

```
---
- hosts: webservers
  remote_user: yourname
  sudo: yes
```

Playbook基础

- 主机与用户

同样的，你可以仅在一个 task 中，使用 sudo 执行命令，而不是在整个play中使用sudo：

```
---
- hosts: webservers
  remote_user: yourname
  tasks:
     - service: name=nginx state=started
       sudo: yes
```

Playbook基础

- 主机与用户

  你也可以登陆后，sudo 到不同的用户身份，而不是使用root：

```
---
- hosts: webservers
  remote_user: yourname
  sudo: yes
  sudo_user: postgres
```

Playbook基础

- **Tasks列表**

每一个 play 包含了一个task列表（任务列表）。一个task在其所对应的所有主机上（通过host pattern匹配的所有主机）执行完毕之后，下一个task才会执行。有一点需要明白的是（很重要），在一个play之中，所有 hosts 会获取相同的任务指令，这是play的一个目的所在，也就是将一组选出的 hosts 映射到 task。（注：此处翻译未必准确，暂时保留原文）在运行playbook时（从上到下执行），如果一个 host 执行 task 失败，这个 host 将会从整个playbook的rotation中移除。如果发生执行失败的情况。请修正 playbook 中的错误，然后重新执行即可。

每个task的目标在于执行一个moudule，通常是带有特定的参数来执行。在参数中可以使用变量（variables）。

modules 具有“幂等”性，意思是如果你再一次地执行moudule（译者注:比如遇到远端系统被意外改动，需要恢复原状），moudle只会执行必要的改动，只会改变需要改变的地方。所以重复多次执行 playbook 也很安全。

对于*command* module和*shell* module，重复执行playbook，实际上是重复运行同样的命令。如果执行的命令类似于 ‘chmod’ 或者 ‘setsebool’ 这种命令，这没有任何问题。也可以使用一个叫做 ‘creates’ 的flag使得这两个modules变得具有“幂等”特性（不是必要的）。

每一个task必须有一个名称*name*，这样在运行playbook时，从其输出的任务执行信息中可以很好的辨别出是属于哪一个task的，如果没有定义*name*,‘action’ 的值将会用作输出信息中标记特定的 task。如果要声明一个task，以前有一种格式：“action: module options”（可能在一些老的 playbooks 中还能见到）。现在推荐使用更常见的格式：”module:options”，本文档使用的就是这种格式。

Playbook基础

- **Tasks列表**

下面是一种基本的task的定义，service moudle 使用key=value格式的参数，这也是大多数module使用的参数格式：

```
tasks:
- name: make sure apache is running
  service: name=httpd state=running
```

Playbook基础

- **Tasks列表**

  比较特别的两个modudle是*command*和*shell*，它们不使用key=value格式的参数，而是这样：

  ```
  tasks:
  - name: disable selinux
    command: /sbin/setenforce 0
  ```

Playbook基础

- **Tasks列表**

使用command module和shell module时，我们需要关心返回码信息，如果有一条命令，它的成功执行的返回码不是0，你或许希望这样做：

```
tasks:
  - name: run this command and ignore the result
    shell: /usr/bin/somecommand || /bin/true
```

或者是这样：

```
tasks:
  - name: run this command and ignore the result
    shell: /usr/bin/somecommand
    ignore_errors: True
```

Playbook基础

- **Tasks列表**

如果action行看起来太长，你可以使用space（空格）或者indent（缩进） 隔开连续的一行：

```
tasks:
  - name: Copy ansible inventory file to client
    copy: src=/etc/ansible/hosts dest=/etc/ansible/hosts
          owner=root group=root mode=0644
```

YAML语法

- 基本的 YAML

对于 Ansible，每一个YAML文件都是从一个列表开始。列表中的每一项都是一个键值对， 通常它们被称为一个 “哈希” 或 “字典”。 所以，我们需要知道如何在YAML中编写列表和字典。

YAML还有一个小的怪癖。所有的YAML文件（无论和 Ansible 有没有关系）开始行都应该是 ---。这是YAML格式的一部分，表明一个文件的开始。

YAML语法

- 基本的 YAML

列表中的所有成员都开始于相同的缩进级别，并且使用一个 “- ” 作为开头（一个横杠和一个空格）：

```
---
# 一个美味水果的列表
- Apple
- Orange
- Strawberry
- Mango
```

YAML语法

- **基本的 YAML**

一个字典是由一个简单的“键: 值”的形式组成（这个冒号后面必须是一个空格）：

```
---
# 一位职工的记录
name: Example Developer
job: Developer
skill: Elite
```

YAML语法

- **基本的 YAML**

  字典也可以使用缩进形式来表示，如果你喜欢这样的话：

  ```
  ---
  # 一位职工的记录
  {name: Example Developer, job: Developer, skill: Elite}
  ```

YAML语法

- 基本的 YAML

Ansible并不是太多的使用这种格式，但是你可以通过以下格式来指定一个布尔值（true/fase）：

```
---
create_key: yes
needs_agent: no
knows_oop: True
likes_emacs: TRUE
uses_cvs: false
```

YAML语法

- 基本的 YAML

让我们把目前学到的YAML例子组合起来。这些在Ansible中什么也干不了，但这些格式将会给你感觉：

```
---
# 一位职工记录
name: Example Developer
job: Developer
skill: Elite
employed: True
foods:
    - Apple
    - Orange
    - Strawberry
    - Mango
languages:
    ruby: Elite
    python: Elite
    dotnet: Lame
```

这就是你开始编写*Ansible* playbooks所需要知道的所有YAML语法。

执行一个Playbook

既然现在你已经学习了playbook的语法，那要如何运行一个playbook呢？这很简单，这里的示例是并行的运行playbook，并行的级别是10（译者注：是10个并发的进程）：

```
ansible-playbook playbook.yml -f 10
```

谢谢