# 第 1 章 Ansible概述

## 1. IT自动化的好处

### 团队影响

* 节省时间，提高工作效率
* 消除重复任务
* 更少的错误风险
* 改善协作和工作满意度

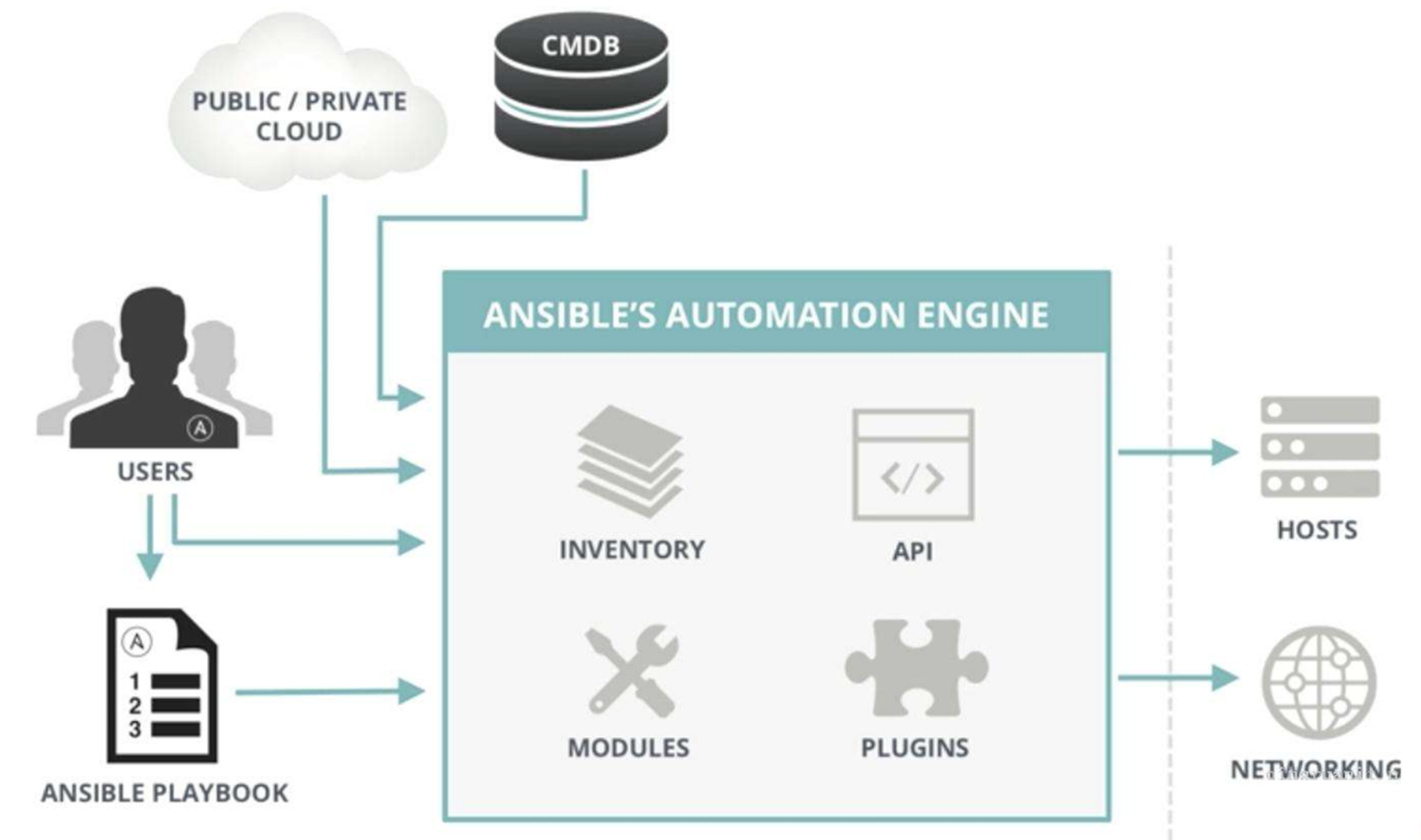
### 企业影响

* 克服复杂性
* 更多创新资源
* 加强问责制和合规性

## 2. Ansible是什么

Ansible是一种IT自动化工具。它可以配置系统，部署软件以及协调更高级的IT任务，例如持续部署，滚动更新。Ansible适用于管理企业IT基础设施，从具有少数主机的小规模到数千个实例的企业环境。Ansible也是一种简单的自动化语言，可以完美地描述IT应用程序基础结构。

## 3. Ansible架构



## 4. 先来认识一下Ansible

# 第 2 章 Ansible安装与配置

## 1.Ansible使用要求

### 服务端要求

* Python2.6/2.7/3.x
* RedHat，Debian，CentOS，OS X等。不支持Windows

### 被管理端要求

* OpenSSH
* Python2.6/2.7/3.x

## 2.安装Ansible

* yum install ansible （推荐）
* pip install ansible
* https://releases.ansible.com/ansible or https://github.com/ansible/ansible.git

## 3.配置文件

# vi /etc/ansible/ansible.cfg   
[defaults]  
  
inventory = /etc/ansible/hosts  
  
forks = 5  
  
become = root  
  
remote\_port = 22  
  
host\_key\_checking = False  
  
timeout = 10  
  
log\_path = /var/log/ansible.log  
  
private\_key\_file = /root/.ssh/id\_rsa

## 4.Inventory（主机清单）

# 示例1：未分组的主机  
green.example.com  
blue.example.com  
192.168.100.1  
192.168.100.10  
   
# 示例2：属于webservers组主机集合  
[webservers]  
alpha.example.org  
beta.example.org  
192.168.1.100  
192.168.1.110  
www[001:006].example.com   
   
示例3：属于dbservers组主机集合  
[dbservers]  
db01.intranet.mydomain.net  
db02.intranet.mydomain.net  
10.25.1.56  
10.25.1.57  
db-[99:101]-node.example.com

主机和主机组变量：  
  
[webservers]  
192.168.1.10 ansible\_ssh\_user=root ansible\_ssh\_pass='123456’ http\_port=80  
192.168.1.11 ansible\_ssh\_user=root ansible\_ssh\_pass='123456’ http\_port=80  
  
[webservers:vars]  
http\_port=8080  
server\_name=www.ctnrs.com  
  
组变量分解到单个文件：  
  
# cat /etc/ansible/group\_vars/webservers.yml   
http\_port: 8080  
server\_name: www.ctnrs.com

# 第 3 章 ad-hoc命令

## 1.命令行工具常用选项

格式：ansible <host-pattern> [ options ]  
  
选项：  
-a MODULE\_ARGS, --args=MODULE\_ARGS 模块参数  
-C, --check 运行检查，不执行任何操作  
-e EXTRA\_VARS, --extra-vars=EXTRA\_VARS 设置附加变量 key=value  
-f FORKS, --forks=FORKS 指定并行进程数量，默认5  
-i INVENTORY, --inventory=INVENTORY 指定主机清单文件路径  
--list-hosts 输出匹配的主机列表，不执行任何操作  
-m MODULE\_NAME, --module-name=MODULE\_NAME 执行的模块名，默认command  
--syntax-check 语法检查playbook文件，不执行任何操作  
-t TREE, --tree=TREE 将日志输出到此目录  
-v, --verbose 详细信息，-vvv更多, -vvvv debug  
--version 查看程序版本  
  
连接选项：控制谁连接主机和如何连接  
-k, --ask-pass 请求连接密码  
--private-key=PRIVATE\_KEY\_FILE, --key-file=PRIVATE\_KEY\_FILE 私钥文件  
-u REMOTE\_USER, --user=REMOTE\_USER 连接用户，默认None  
-T TIMEOUT, --timeout=TIMEOUT 覆盖连接超时时间，默认10秒  
  
提权选项：控制在目标主机以什么用户身份运行  
-b, --become 以另一个用户身份操作  
--become-method=BECOME\_METHOD 提权方法，默认sudo  
--become-user=BECOME\_USER 提权后的用户身份，默认root  
-K, --ask-become-pass 提权密码

## 2.SSH密码认证

[webservers]  
192.168.1.10:22 ansible\_ssh\_user=root ansible\_ssh\_pass=’123456’  
192.168.1.11:22 ansible\_ssh\_user=root ansible\_ssh\_pass=’123456’

## 3.SSH秘钥对认证

[webservers]  
192.168.1.10:22 ansible\_ssh\_user=root ansible\_ssh\_key=/root/.ssh/id\_rsa  
192.168.1.11:22 ansible\_ssh\_user=root

# 第 4 章 Ansible常用模块

* 执行shell命令（command和shell）
* 文件传输（copy和file）
* 管理软件包（yum）
* 用户和组（user）
* 从源代码管理系统部署（git）
* 管理服务（service）
* 收集目标主机信息（setup）

# 第 5 章 Playbook基本使用

## 1.使用Playbook的好处

特点

* 易读的编排语言
* 适合配置管理和应用部署
* 非常适合部署复杂的工作

## 2.先来认识一下Playbook

自动部署Nginx

# main.yml  
---  
- hosts: webservers  
 vars:  
 hello: Ansible  
   
 tasks:  
 - name: Add repo   
 yum\_repository:  
 name: nginx  
 description: nginx repo  
 baseurl: http://nginx.org/packages/centos/7/$basearch/  
 gpgcheck: no  
 enabled: 1  
 - name: Install nginx  
 yum:  
 name: nginx  
 state: latest  
 - name: Copy nginx configuration file  
 copy:  
 src: ./site.conf  
 dest: /etc/nginx/conf.d/site.conf  
 - name: Start nginx  
 service:  
 name: nginx  
 state: started  
 - name: Create wwwroot directory  
 file:  
 dest: /var/www/html  
 state: directory  
 - name: Create test page index.html  
 shell: echo "hello {{hello}}" > /var/www/html/index.html  
  
  
# site.conf  
server {  
 listen 80;  
 server\_name www.ctnrs.com;  
 location / {  
 root /var/www/html;  
 index index.html;  
 }  
}

## 3.YAML语法

* 缩进表示层级关系
* 不支持制表符“tab”缩进，使用空格缩进
* 通常开头缩进 2 个空格
* 字符后缩进 1 个空格，如冒号、逗号等
* “---” 表示YAML格式，一个文件的开始
* “#”注释

## 4.Playbook文件结构

---  
- name: play1  
 hosts: webservers  
 remote\_user: root  
 vars:  
 var\_name: value  
 tasks:  
 - name: echo  
 shell: "echo {{var\_name}}"  
   
- name: play2  
 hosts: webservers  
 remote\_user: root  
 vars:  
 var\_name: value  
 tasks:  
 - name: echo  
 shell: "echo {{var\_name}}"  
   
   
 ---  
- hosts: webservers  
 remote\_user: root  
 vars:  
 var\_name: value  
 tasks:  
 - name: echo  
 shell: "echo {{var\_name}}"

## 5.在变更时执行操作（handlers）

notify：在任务结束时触发

handlers：由特定条件触发Tasks

---  
hosts: webservers  
 gather\_facts: no  
  
 tasks:  
 - name: Copy nginx configuration file  
 copy:  
 src: ./site.conf  
 dest: /etc/nginx/conf.d/site.conf  
 notify:  
 - restart nginx  
  
 handlers:  
 - name: restart nginx  
 service: name=nginx state=reloaded

## 6.任务控制（tags）

---  
- hosts: webservers  
 gather\_facts: no  
  
 tasks:  
 - name: Install redis  
 yum: name=redis state=present  
 tags: install  
   
 - name: Copy redis configuration file  
 copy: src=redis.conf dest=/etc/redis/redis.conf  
 tags: configuration  
  
 - name: Restart redis  
 service: name=redis state=restarted  
 tags: restart

## 7.Playbook文件调试

语法检查：ansible-playbook main.yml --syntax-check  
  
打印语句：  
---  
- hosts: webservers  
 tasks:  
 - debug:  
 msg: {{group\_names}}  
 - debug:   
 msg: {{inventory\_hostname}}  
 - debug:  
 msg: {{ansible\_hostname}}

## 8.案例：自动部署Tomcat

---  
hosts: webservers   
 gather\_facts: no  
 vars:  
 tomcat\_version: 8.5.34  
 tomcat\_install\_dir: /usr/local  
   
 tasks:  
 - name: Install jdk1.8  
 yum: name=java-1.8.0-openjdk state=present  
   
 - name: Download tomcat  
 get\_url: url=http://mirrors.hust.edu.cn/apache/tomcat/tomcat-8/v{{ tomcat\_version }}/bin/  
 apache-tomcat-{{ tomcat\_version }}.tar.gz dest=/tmp  
   
 - name: Unarchive tomcat-{{ tomcat\_version }}.tar.gz  
 unarchive:  
 src: /tmp/apache-tomcat-{{ tomcat\_version }}.tar.gz   
 dest: "{{ tomcat\_install\_dir }}"  
 copy: no   
   
 - name: Start tomcat   
 shell: cd {{ tomcat\_install\_dir }} &&  
 mv apache-tomcat-{{ tomcat\_version }} tomcat8 &&  
 cd tomcat8/bin && nohup ./startup.sh &

# 第 6 章 Playbook定义变量与使用

## 1.命令行

## 2.在Inventory中定义

## 3.在Playbook中定义

# 在Playbook中定义变量  
---  
hosts: webservers  
 gather\_facts: no  
 vars:  
 var\_name: value  
 var\_name: value  
 tasks:  
 - name: hello  
 shell: "echo {{var\_name}}"

## 4.在Role中定义

## 5.注册变量（register）

# 注册变量  
---  
hosts: webservers   
 gather\_facts: no  
 tasks:  
 - name: Get date   
 command: date +"%F\_%T"  
 register: date\_output  
 - name: Echo date\_output  
 command: touch /tmp/{{date\_output.stdout}}

## 6.系统信息变量（facts）

# 系统变量  
---  
hosts: webservers   
 tasks:  
 - name: Get hostname  
 debug: msg={{ansible\_hostname}}

# 第 7 章 Playbook文件复用

## 1.include & import 区别

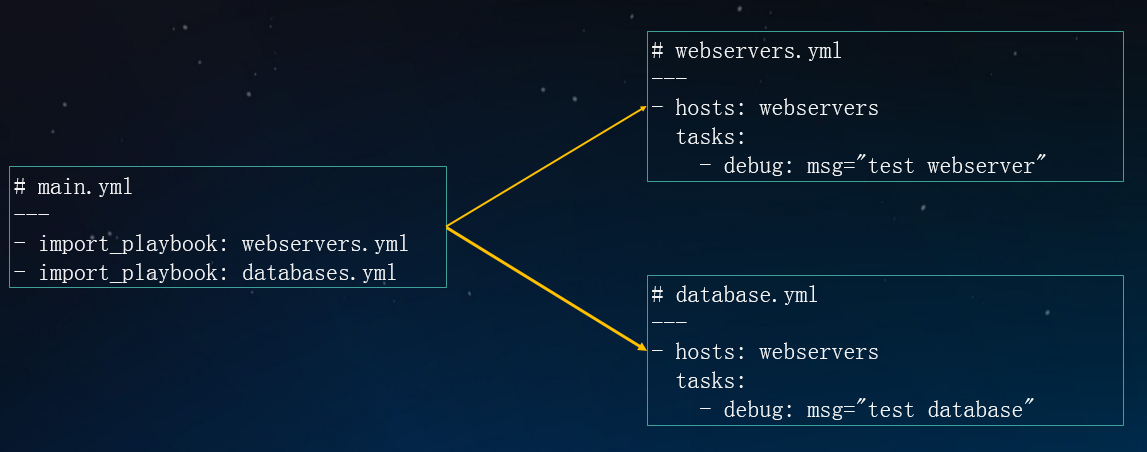
include\*（动态）：在运行时导入

* --list-tags，--list-tasks不会显示到输出
* 不能使用notify触发来自include\*内处理程序名称（handlers）

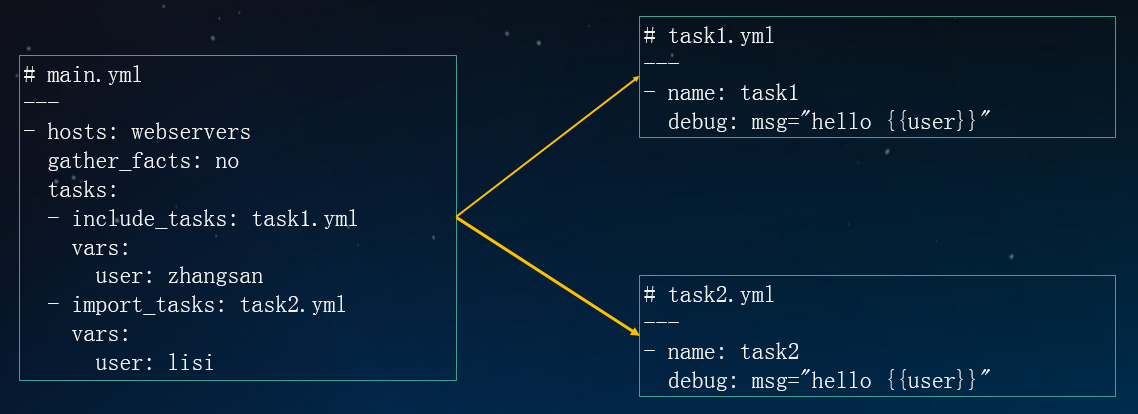
import\*（静态）：在Playbook解析时预先导入

* 不能与循环一起使用
* 将变量用于目标文件或角色名称时，不能使用inventory（主机/主机组等）中的变量

## 2.import\_playbook



## 3.include*tasks & import*tasks



# 第 8 章 Playbook流程控制

## 1.条件

- hosts: webservers  
   
 tasks:  
 - name: Host 192.168.1.12 run this task  
 debug: msg="{{ansible\_default\_ipv4.address}}"  
 when: ansible\_default\_ipv4.address == '192.168.1.12 '

- hosts: webservers  
   
 tasks:  
 - name: Update apache version - yum  
 yum: name=httpd state=present  
 when: ansible\_pkg\_mgr == 'yum'  
 notify: restart httpd  
   
 - name: Update apache version - apt  
 apt: name=apache2 state=present update\_cache=yes  
 when: ansible\_pkg\_mgr == 'apt'  
 notify: restart apache2  
   
 handlers:  
 - name: restart httpd  
 service: name=httpd state=restared  
 handlers:  
 - name: restart apache2  
 service: name=apache2 state=restared

tasks:  
 - name: "shut down CentOS 6 and Debian 7 systems"  
 command: /sbin/shutdown -t now  
 when: (ansible\_distribution == "CentOS" and ansible\_distribution\_major\_version == "6") or  
 (ansible\_distribution == "Debian" and ansible\_distribution\_major\_version == "7")

tasks:  
 - name: "shut down CentOS 6 systems"  
 command: /sbin/shutdown -t now  
 when:  
 - ansible\_distribution == "CentOS"  
 - ansible\_distribution\_major\_version == "6"

## 2.循环

- name: with\_list  
 debug:  
 msg: "{{ item }}"  
 with\_list:  
 - one  
 - two  
  
- name: with\_list -> loop  
 debug:  
 msg: "{{ item }}"  
 loop:  
 - one  
 - two

- name: with\_items  
 debug:  
 msg: "{{ item }}"  
 with\_items: "{{ items }}"  
  
- name: with\_items -> loop  
 debug:  
 msg: "{{ item }}"  
 loop: "{{ items|flatten(levels=1) }}"

# 第 9 章 Playbook模板（jinja2）

## 1.条件和循环

# test.yml   
---  
- hosts: webservers  
 vars:  
 hello: Ansible  
   
 tasks:  
 - template: src=f.j2 dest=/tmp/f.j2  
  
  
# f.j2   
{% set list=['one', 'two', 'three'] %}  
   
{% for i in list %}  
 {% if i == 'two' %}  
 -> two  
 {% elif loop.index == 3 %}  
 -> 3  
 {% else %}  
 {{i}}  
 {% endif %}  
{% endfor %}   
   
{{ hello }}

{% set dict={'zhangsan': '26', 'lisi': '25'} %}  
{% for key, value in dict.iteritems() %}  
 {{key}} -> {{value}}  
{% endfor %}

## 2.案例：管理Nginx配置文件

# main.yml   
---  
- hosts: webservers  
 gather\_facts: no  
 vars:  
 http\_port: 80  
 server\_name: www.ctnrs.com  
   
 tasks:  
 - name: Copy nginx configuration file   
 template: src=site.conf.j2 dest=/etc/nginx/conf.d/www.ctnrs.com.conf  
 notify: reload nginx  
   
 handlers:  
 - name: reload nginx  
 service: name=nginx state=reloaded

# site.conf.j2   
{% set list=[10, 12, 13, 25, 31] %}  
upstream {{server\_name}} {  
 {% for i in list %}  
 server 192.168.1.{{i}}:80;  
 {% endfor %}  
}  
server {  
 listen {{ http\_port }};  
 server\_name {{ server\_name }};  
   
 location / {  
 proxy\_pass http://{{server\_name}};  
 }   
}

# 第 10 章 角色（roles）

## 1.Roles目录结构

site.yml   
webservers.yml   
fooservers.yml   
roles/   
 common/   
 tasks/   
 handlers/   
 files/   
 templates/   
 vars/   
 defaults/   
 meta/   
 webservers/   
 tasks/   
 defaults/   
 meta/

* tasks - 包含角色要执行的主要任务列表
* handlers - 包含角色使用的处理程序
* defaults - 角色默认的变量
* vars - 角色其他的变量
* files - 角色部署时用到的文件
* templates - 角色部署时用到的模板
* meta - 角色定义的一些元数据

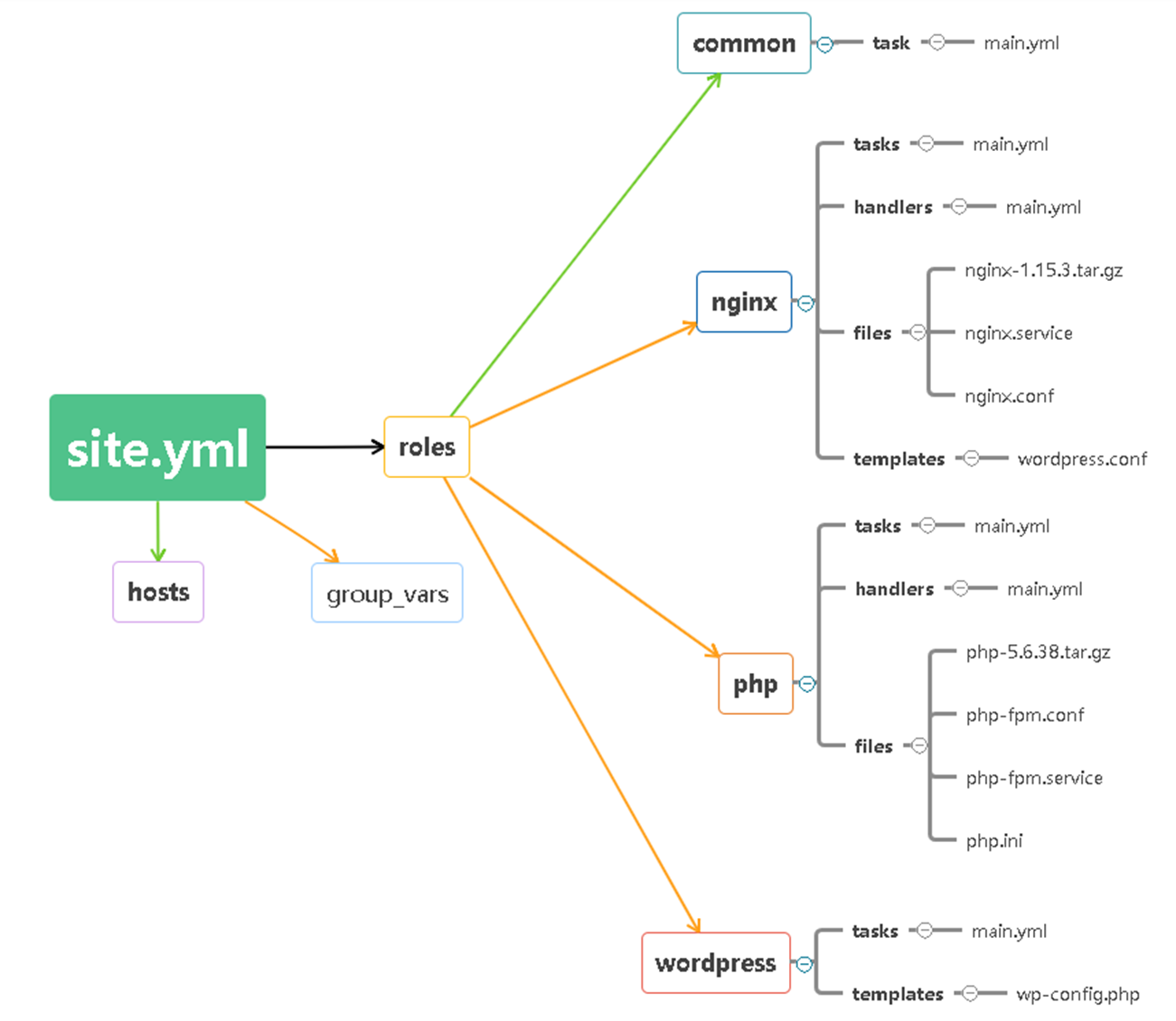
## 2.Roles基本使用

---  
- hosts: webservers  
 roles:  
 - common  
 - nginx  
 - php

---  
- hosts: webservers  
 roles:  
 - common  
 - role: nginx  
 vars:  
 dir: '/opt/a'  
 app\_port: 5000  
 - role: php  
 vars:  
 dir: '/opt/b'  
 app\_port: 5001

---  
- hosts: webservers  
 roles:  
 - role: common  
 tags: ["common"]  
 - role: nginx  
 tags: ["nginx"]  
 - role: php  
 tags: ["php"]

## 3.案例：部署Web服务器



最佳实践：https://docs.ansible.com/ansible/latest/user*guide/playbooks*best\_practices.html

示例参考：https://github.com/ansible/ansible-examples

## 学员群：545214087