~~Edit~~

运维工具及 Ansible 入门

- role 基于 when 条件判断

1.1 1.运维工具介绍及 ansible 入门 ~73 分钟

运维自动化是什么（机器，配置，发布，灰度）
从 pxe 批量安装系统自动化开始引入
配置管理
puppet(ruby)
salastck(python)
chef
cfengine

Command and Control:
fabric
func

- 程序发布：

  人工智能（手动发布）
  脚本
  发布程序（运维程序）
  不能影响用户体验
  系统不能停机
  不能导致系统故障或造成系统不可用

灰度模型：
主机
用户

1.1.1发布过程

上次分享的 ansible ppt 中有图可以用

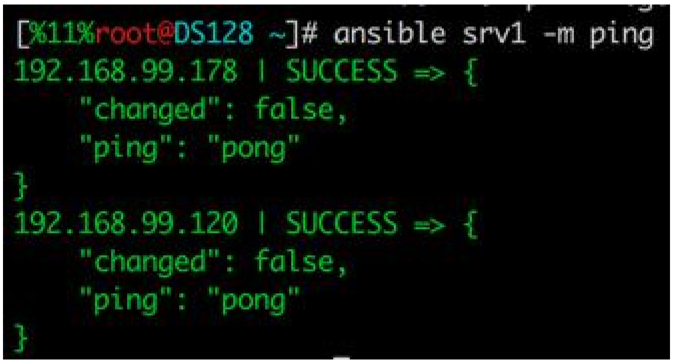
1.1.2运维工具分类

agent
agentless

```
  ssh-keygen -t rsa -P ''
ssh-copy-id -i .ssh/id_rsa.pub root@192.168.99.120
ansible srv1 -m ping
```

ansible 简单使用格式：

```
  ansible <host-pattern> [options]
```



```
[%11%root@DS128 ~]# ansible srv1 -m ping
192.168.99.178 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
192.168.99.120 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
```

1.2 2. Ansible 常用模块及 playbook 基础 ~63 分钟

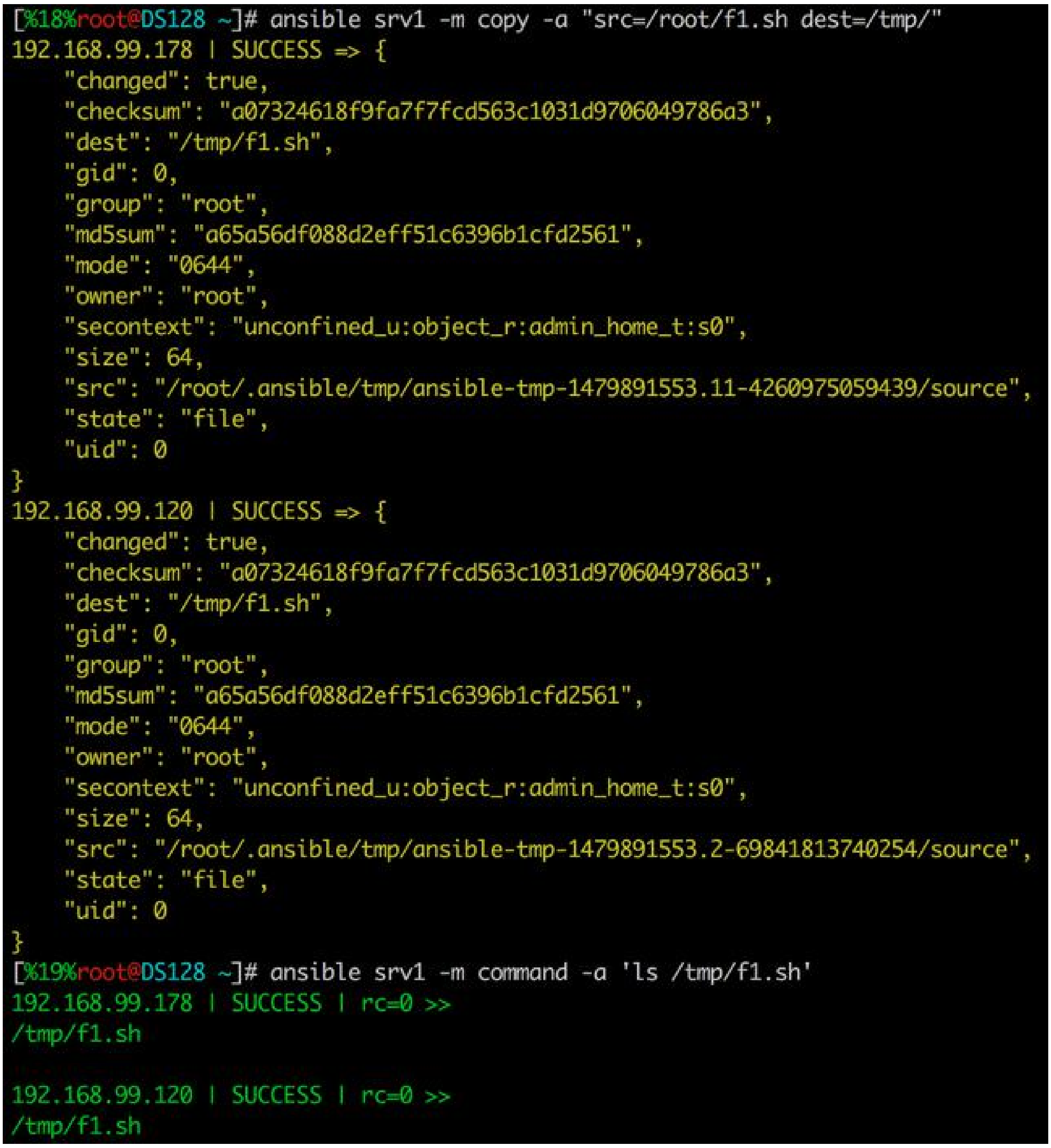
1.2.1 命令使用

ansible
ansible-doc -l

1.2.2 常用模块列表

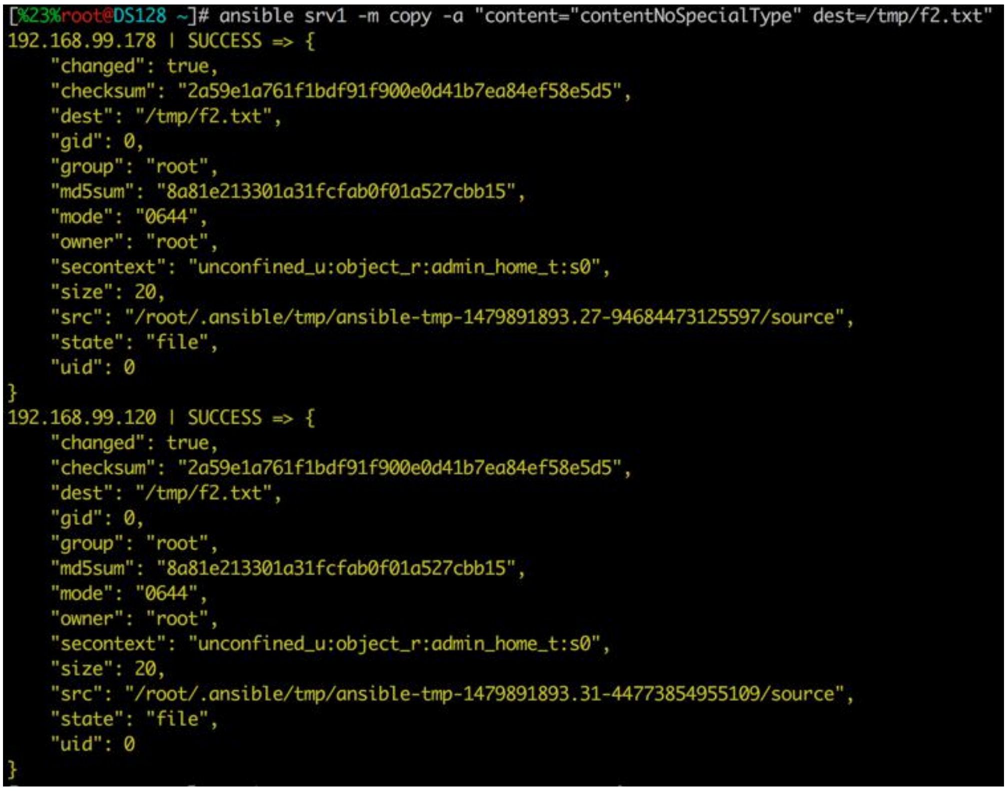
- command
    -
        - ansible srv1 -m ping
        - ansible srv1 -m command -a ‘echo magedu | passwd –stdin user1’ 这条命令执行结果不会如预期，因为不支持 | 管道
- shell
    -
        - ansible srv1 -m shell -a ‘echo magedu | passwd –stdin user1’ shell 命令支持管道

- copy
    -
        - ansible srv1 -m copy -a “src=/root/f1.sh dest=/tmp/”
        - ansible srv1 -m copy -a “content=”contentNoSpecialType” dest=/tmp/f2.txt”



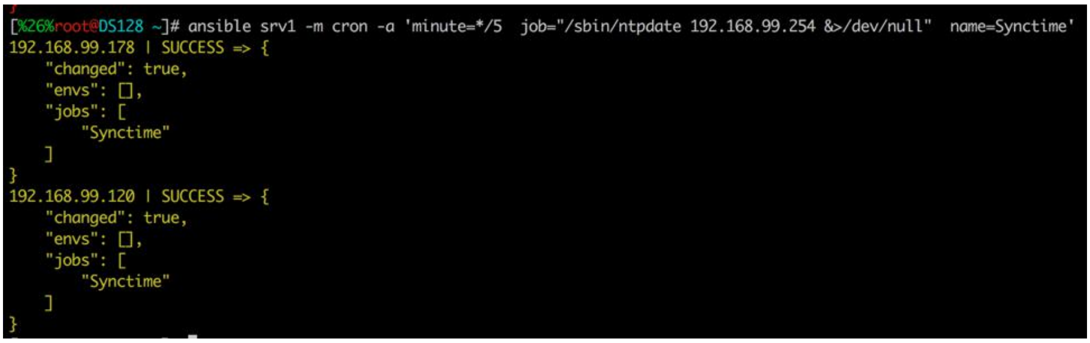
```
[%18%root@DS128 ~]# ansible srv1 -m copy -a "src=/root/f1.sh dest=/tmp/"
192.168.99.178 | SUCCESS => {
    "changed": true,
    "checksum": "a07324618f9fa7f7fcd563c1031d9706049786a3",
    "dest": "/tmp/f1.sh",
    "gid": 0,
    "group": "root",
    "md5sum": "a65a56df088d2eff51c6396b1cfd2561",
    "mode": "0644",
    "owner": "root",
    "secontext": "unconfined_u:object_r:admin_home_t:s0",
    "size": 64,
    "src": "/root/.ansible/tmp/ansible-tmp-1479891553.11-4260975059439/source",
    "state": "file",
    "uid": 0
}
192.168.99.120 | SUCCESS => {
    "changed": true,
    "checksum": "a07324618f9fa7f7fcd563c1031d9706049786a3",
    "dest": "/tmp/f1.sh",
    "gid": 0,
    "group": "root",
    "md5sum": "a65a56df088d2eff51c6396b1cfd2561",
    "mode": "0644",
    "owner": "root",
    "secontext": "unconfined_u:object_r:admin_home_t:s0",
    "size": 64,
    "src": "/root/.ansible/tmp/ansible-tmp-1479891553.2-69841813740254/source",
    "state": "file",
    "uid": 0
}
[%19%root@DS128 ~]# ansible srv1 -m command -a 'ls /tmp/f1.sh'
192.168.99.178 | SUCCESS | rc=0 >>
/tmp/f1.sh

192.168.99.120 | SUCCESS | rc=0 >>
/tmp/f1.sh
```

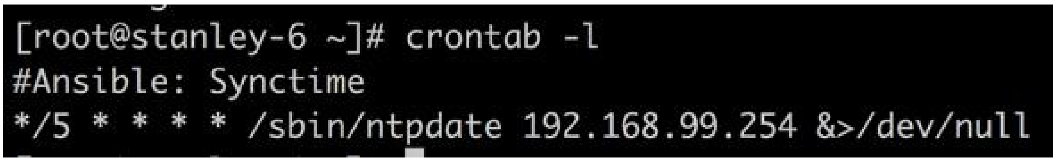


```
[%23%root@DS128 ~]# ansible srv1 -m copy -a "content="contentNoSpecialType" dest=/tmp/f2.txt"
192.168.99.178 | SUCCESS => {
    "changed": true,
    "checksum": "2a59e1a761f1bdf91f900e0d41b7ea84ef58e5d5",
    "dest": "/tmp/f2.txt",
    "gid": 0,
    "group": "root",
    "md5sum": "8a81e213301a31fcfab0f01a527cbb15",
    "mode": "0644",
    "owner": "root",
    "secontext": "unconfined_u:object_r:admin_home_t:s0",
    "size": 20,
    "src": "/root/.ansible/tmp/ansible-tmp-1479891893.27-94684473125597/source",
    "state": "file",
    "uid": 0
}
192.168.99.120 | SUCCESS => {
    "changed": true,
    "checksum": "2a59e1a761f1bdf91f900e0d41b7ea84ef58e5d5",
    "dest": "/tmp/f2.txt",
    "gid": 0,
    "group": "root",
    "md5sum": "8a81e213301a31fcfab0f01a527cbb15",
    "mode": "0644",
    "owner": "root",
    "secontext": "unconfined_u:object_r:admin_home_t:s0",
    "size": 20,
    "src": "/root/.ansible/tmp/ansible-tmp-1479891893.31-44773854955109/source",
    "state": "file",
    "uid": 0
}
```

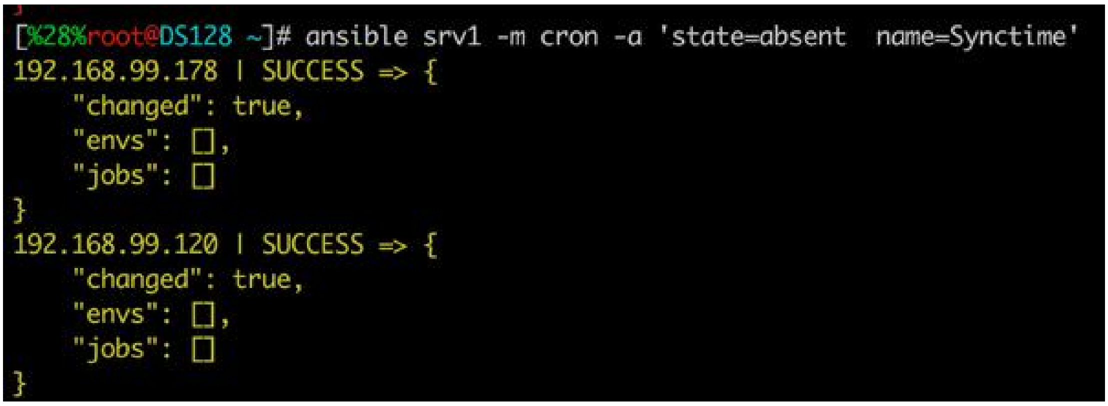
- cron
    -
        - ansible srv1 -m cron -a ‘minute=*/5 job=”/sbin/ntpdate 192.168.99.254 &>/dev/null” name=Synctime’
        - ansible srv1 -m cron -a ‘state=absent name=Synctime’



```
[%26%root@DS128 ~]# ansible srv1 -m cron -a 'minute=*/5  job="/sbin/ntpdate 192.168.99.254 &>/dev/null"  name=Synctime'
192.168.99.178 | SUCCESS => {
    "changed": true,
    "envs": [],
    "jobs": [
        "Synctime"
    ]
}
192.168.99.120 | SUCCESS => {
    "changed": true,
    "envs": [],
    "jobs": [
        "Synctime"
    ]
}
```



```
[root@stanley-6 ~]# crontab -l
#Ansible: Synctime
*/5 * * * * /sbin/ntpdate 192.168.99.254 &>/dev/null
```



```
[%28%root@DS128 ~]# ansible srv1 -m cron -a 'state=absent  name=Synctime'
192.168.99.178 | SUCCESS => {
    "changed": true,
    "envs": [],
    "jobs": []
}
192.168.99.120 | SUCCESS => {
    "changed": true,
    "envs": [],
    "jobs": []
}
```

- fetch(拉)
- file 设置文件属性
- hostname
- pip
- ping
- yum
  -
    - ansible srv1 -m yum -a ‘name=httpd state=latest’
- service
  -
    - arguments
    - name
    - enabled
    - pattern
    - runlevel
    - sleep
    - state
      -
        - started
        - ansible srv1 -m service -a ‘name=httpd——————’
        - stopped
        - restarted
    - user

1.2.3 YAML

1.3 3.Ansible playbook 应用详解 ~70 分钟

1.3.1 Playbook 核心元素

Hosts:
Tasks:
Varniables
Templates:
Handlers:
监控资源改变时才会触发改变
Roles:

tags: 打标签，分别执行

Hosts: 运行指定任务的目标主机
remote_user: 在远程主机上执行任务的用户
tasks: 任务列表
一个任务在所有主机运行完后再在其它主机上运行下一个任务

测试：

```
ansible-playbook --check

ansible 192.168.99.120 -m setup
```

playbook

```
---

- hosts: srv1
  remote_user: root
  tasks:
    - name: Install httpd
      yum: name=httpd state=present

    - name: Install configure file
      copy: src=files/httpd.conf dest=/etc/httpd/conf/
```

```
     tags: conf
     notify: restart httpd

   - name: start httpd service
     tags: process
     service: name=httpd state=started
```

1.3.2变量：

variable:

facts: 可直接调用

ansible-playbook 命令调用的自定义变量

-e VARS, - extra-vars=VARS

在/etc/ansible/hosts 添加变量

```
  [srv1]
192.168.99.120 http_port=8080 hname=www120
192.168.99.178 http_port=80 hname=www178
```

组变量

```
  [svr1:vars]
http_port=808

[srv1]
192.168.99.120 http_port=8080 hname=www120
192.168.99.178 http_port=80 hname=www178
```

Inventory 参数：

用于定义 ansible 连接目标主机时使用的参数（这里好像有问题，再看看官网再）：

```
  ansible_ssh_host
ansible_ssh_port
ansible_ssh_user
ansible_ssh_pass
ansible_sudo_pass
```

1.4 4.ansible playbook 用法详解 01 ~70 分钟

模块：

group 模块

scripts 脚本 #将本地的脚本复制到远程主机并在远程主机执行

1.4.1 template

worker_processes {{ ansible_processor_vcpus }};

worker_processes {{ ansible_processor_vcpus+1 }}; #支持算术运算

ansible-playbook tem.yml

cat tem.yml

```
   ---

- hosts: srv120
  remote_user: root
  tasks:
  - name:
    template: src=nginx.conf.j2 dest=/etc/nginx/nginx.conf
```

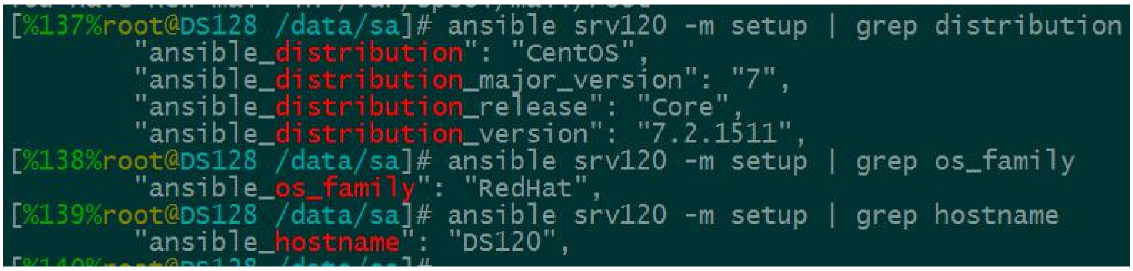
1.5 条件判断

```
   ansible srv120 -m setup | grep distribution
ansible srv120 -m setup | grep os_family
ansible srv120 -m setup | grep hostname
```

cat tem.yml

```
   ---

- hosts: srv120
  remote_user: root
  tasks:
  - name:
    template: src=nginx.conf.j2 dest=/etc/nginx/nginx.conf
    when: ansible_distribution_major_version == "7"
```



```
You have new mail in /var/spool/mail/root
[%137%root@DS128 /data/sa]# ansible srv120 -m setup | grep distribution
        "ansible_distribution": "CentOS",
        "ansible_distribution_major_version": "7",
        "ansible_distribution_release": "Core",
        "ansible_distribution_version": "7.2.1511",
[%138%root@DS128 /data/sa]# ansible srv120 -m setup | grep os_family
        "ansible_os_family": "RedHat",
[%139%root@DS128 /data/sa]# ansible srv120 -m setup | grep hostname
        "ansible_hostname": "DS120",
[%140%root@DS128 /data/sa]#
```

1.6 循环 ~with_itmes，字典

1. whti_items

迭代，重复腿毛的任务，对迭代项的引用，固定变量名为 item，而后在 task 中使用 with_items 给定迭代的元素列表；

```
   ---

- name: Create rsyncd config
  copy: src={{ item }} dest=/etc/{{ item }}
  with_items:
    - rsyncd.secrets
    - rsyncd.conf
```

2. 字典

```
   - name: 使用 ufw 模块来管理哪些端口需要开启
  ufw:
    rule: "{{ item.rule }}"
    port: "{{ item.port }}"
    proto: "{{ item.proto }}"
  with_items:
    - { rule: 'allow', port: 22, proto: 'tcp' }
    - { rule: 'allow', port: 80, proto: 'tcp' }
    - { rule: 'allow', port: 123, proto: 'udp' }

- name: 配置网络进出方向的默认规则
```

```
  ufw:
    direction: "{{ item.direction }}"
    policy: "{{ item.policy }}"
    state: enabled
  with_items:
    - { direction: outgoing, policy: allow }
    - { direction: incoming, policy: deny }
```

3. 字典 default

```
  ---
- name: Ensure MySQL users are present.
  mysql_user:
    name: "{{ item.name }}"
    host: "{{ item.host | default('localhost') }}"
    password: "{{ item.password }}"
    priv: "{{ item.priv | default('*.*:USAGE') }}"
    state: present
    append_privs: "{{ item.append_privs | default('no') }}"
  with_items: "{{ mysql_users }}"
  no_log: true
```

1.6.2template for 循环

```
  {% for vhost in nginx_vhosts %}
server {
    listen {{ vhost.listen | default('80 default_server') }};

    {% if vhost.server_name is defined %}
    server_name {{ vhost.server_name }};
    {% endif %}

    {% if vhost.root is defined %}
    root {{ vhost.root }};
    {% endif %}

    index {{ vhost.index | default('index.html index.htm') }};

    {% if vhost.error_page is defined %}
    error_page {{ vhost.error_page }};
    {% endif %}
    {% if vhost.access_log is defined %}
    access_log {{ vhost.access_log }};
    {% endif %}
    {% if vhost.error_log is defined %}
    error_log {{ vhost.error_log }} error;
    {% endif %}

    {% if vhost.return is defined %}
    return {{ vhost.return }};
    {% endif %}

    {% if vhost.extra_parameters is defined %}
    {{ vhost.extra_parameters }}
    {% endif %}
}
{% endfor %}
```

1.7 5.ansible playbook 用法详解 02 ~70 分钟

目录结构如下：

```
  roles/
    mysql/

        files/  传输文件

        templates/ 模板文件

        tasks/  任务集文件

            main.yml

        handlers/  include 包含

            main.yml

        vars/  变量文件

        meta/  当前角色的特殊设定及依赖关系

        default/:  默认变量

    httpd

    ...

    nginx
```

```
        ...
    memcached
        ...
```

在 playbook 调用角色:

```
- hosts: srv120
  remote_user: root
  roles:
    - mysql
    - httpd
    - nginx
    - memcached
```

1. role 中 tags 使用

```
---
# This playbook deploys the whole application stack in this site.

- hosts: localhost
  remote_user: root

  roles:
    - { role: git, tags: [ 'update_pre', 'pkbuild', 'git' ] }
    - { role: composer, tags: [ 'update_pre', 'pkbuild', 'composer'] }
    - { role: create_tar, tags: [ 'update_pre', 'pkbuild', 'create_tar'] }
```

2. role 中变量调用

```
- hosts: zabbix-proxy
  sudo: yes
  roles:
     - { role: geerlingguy.php-mysql }
     - { role: dj-wasabi.zabbix-proxy, zabbix_server_host: 192.168.37.167 }
```

3. role 基于 when 条件判断

```
roles:
  - {role: nginx, when: "ansible_distribution_major_version == '7' " ,username: nginx }
```

http://www.ansible.com.cn