

- 简介
- 定义主机和组
  - 1. 简单的主机和组
  - 2. 指定主机范围
  - 3. 定义主机组嵌套
- 选择主机与组
  - 1. 匹配所有主机
  - 2. 匹配指定的主机或主机组
  - 3. 通配符匹配
  - 4. 通配符组合匹配
  - 4. 正则表达式匹配
  - 5. 通过 `--limit` 明确指定主机或组

## 简介

在使用Ansible来批量管理主机的时候，通常我们需要先定义要管理哪些主机或者主机组，而这个用于管理主机与主机组的文件就叫做Inventory，也叫主机清单。该文件默认位于/etc/ansible/hosts。当然我们也可以通过修改ansible配置文件的hostfile配置项来修改默认inventory的位置。

## 定义主机和组

对于/etc/ansible/hosts最简单的定义格式像下面：

### 1. 简单的主机和组

```
# 中括号中的名字代表组名，可以根据自己的需求将庞大的主机分成具有标识的组，如上面分了两个组
# webserver和dbserver组；
# 主机(hosts)部分可以使用域名、主机名、IP地址表示；当然使用前两者时，也需要主机能反解析到相应的
# IP地址，一般此类配置中多使用IP地址；
mail.yanruogu.com
[webserver]
web1.yanruogu.com
web2.yanruogu.com
[dbserver]
db1.yanruogu.com
db2.yanruogu.com
```

### 2. 指定主机范围

```
# 下面指定了从web01到web50, webservers组共计50台主机; databases组有db-a到db-f共6台主机
[webservers]
www[01:50].yanruogu.com
[databases]
db-[a:f].yanruogu.com
```

### 3. 定义主机组嵌套

一个主机组可以包含若干个子主机组：

```
# 如下示例中, production组包含两个子组, 分别为webservers和dbservers, webservers和dbservers组
分别包含若干主机
[webservers]
web1.lab.example.com
web2.lab.example.com

[dbservers]
db1.lab.example.com
db2.lab.example.com

[production:children]
webservers
dbservers
```

## 选择主机与组

在前面定义Inventory的时候, 我们会把所有被管理主机通过主机组的方式定义到Inventory当中, 但是当我们实际使用的时候, 可能只需要对某一主机或主机组进行操作, 这个时候就需要通过匹配的方式指定某一特定主机或主机组。

在此之前, 我们先定义一个主机清单示例：

```
srv1.example.com
srv2.example.com
s1.lab.example.com
s2.lab.example.com

[web]
jupiter.lab.example.com
saturn.example.com

[db]
db1.example.com
db2.example.com
db3.example.com

[1b]
```

```
lb1.lab.example.com
lb2.lab.example.com

[boston]
db1.example.com
jupiter.lab.example.com
lb2.lab.example.com

[london]
db2.example.com
db3.example.com
file1.lab.example.com
lb1.lab.example.com

[dev]
web1.lab.example.com
db3.example.com

[stage]
file2.example.com
db2.example.com

[prod]
lb2.lab.example.com
db1.example.com
jupiter.lab.example.com

[function:children]
web
db
lb
city

[city:children]
boston
london
environments

[environments:children]
dev
stage
prod
new

[new]
172.25.252.23
172.25.252.44
```

## 1. 匹配所有主机

可以通过 `all` 或者 `*` 来指定匹配所有主机，通过如下指令查看 `all` 匹配到的主机：

```
# ansible all --list-hosts
hosts (16):
  srv1.example.com
  srv2.example.com
  s1.lab.example.com
  s2.lab.example.com
  jupiter.lab.example.com
  saturn.example.com
  db1.example.com
  db2.example.com
  db3.example.com
  lb1.lab.example.com
  lb2.lab.example.com
  file1.lab.example.com
  web1.lab.example.com
  file2.example.com
  172.25.252.23
  172.25.252.44
  172.25.252.32
```

## 2. 匹配指定的主机或主机组

### 1. 匹配单个组

```
# ansible prod --list-hosts
hosts (3):
  lb2.lab.example.com
  db1.example.com
  jupiter.lab.example.com
```

### 2. 匹配单个主机

```
# ansible db2.example.com --list-hosts
hosts (1):
  db2.example.com
```

### 3. 匹配多个主机

```
# ansible 'lb1.lab.example.com,s1.lab.example.com,db1.example.com' --list-hosts
hosts (3):
  lb1.lab.example.com
  s1.lab.example.com
  db1.example.com
```

### 4. 匹配多个组

```
# ansible 'london,boston' --list-hosts
hosts (7):
  db2.example.com
  db3.example.com
  file1.lab.example.com
  lb1.lab.example.com
  db1.example.com
  jupiter.lab.example.com
  lb2.lab.example.com
```

### 5. 匹配不属于任何组的主机

```
# ansible ungrouped --list-hosts
hosts (4):
  srv1.example.com
  srv2.example.com
  s1.lab.example.com
  s2.lab.example.com
```

## 3. 通配符匹配

### 1. 匹配 '\*.example.com':

```
# ansible '*.example.com' --list-hosts
hosts (14):
  srv1.example.com
  srv2.example.com
  s1.lab.example.com
  s2.lab.example.com
  jupiter.lab.example.com
  saturn.example.com
  db1.example.com
  db2.example.com
  db3.example.com
  lb1.lab.example.com
  lb2.lab.example.com
  file1.lab.example.com
  web1.lab.example.com
  file2.example.com
```

### 2. 匹配 172.25.\* 的主机:

```
# ansible '172.25.*' --list-hosts
hosts (3):
  172.25.252.23
  172.25.252.44
  172.25.252.32
```

### 3. 匹配以 s 开头的主机及主机组：

```
# ansible 's*' --list-hosts
hosts (7):
  file2.example.com
  db2.example.com
  srv1.example.com
  srv2.example.com
  s1.lab.example.com
  s2.lab.example.com
  saturn.example.com
```

## 4. 通配符组合匹配

### 1. 匹配包含 \*.example.com 但不包含 \*.lab.example.com 的主机：

```
# ansible '*.example.com,!*.lab.example.com' --list-hosts
hosts (7):
  srv1.example.com
  srv2.example.com
  saturn.example.com
  db1.example.com
  db2.example.com
  db3.example.com
  file2.example.com
```

### 2. 匹配包含prod以及172开头、包含lab关键字的主机或组

```
# ansible 'prod,172*,*lab*' --list-hosts
hosts (11):
  lb2.lab.example.com
  db1.example.com
  jupiter.lab.example.com
  172.25.252.23
  172.25.252.44
  172.25.252.32
  s1.lab.example.com
  s2.lab.example.com
  lb1.lab.example.com
  file1.lab.example.com
  web1.lab.example.com
```

### 3. 匹配属于db组同时还属于london组的主机：

```
# ansible 'db,&london' --list-hosts
hosts (2):
  db2.example.com
  db3.example.com
```

4. 匹配在london组或者boston组，还必须在prod组中且必须不在lb组中的主机：

```
# ansible 'boston,london,&prod,!lb' --list-hosts
hosts (2):
  db1.example.com
  jupiter.lab.example.com
```

## 4. 正则表达式匹配

在开头的地方使用“~”，用来表示这是一个正则表达式：

```
# ansible '~(s/db).*example\.com' --list-hosts
hosts (8):
  srv1.example.com
  srv2.example.com
  s1.lab.example.com
  s2.lab.example.com
  saturn.example.com
  db1.example.com
  db2.example.com
  db3.example.com
```

## 5. 通过 --limit 明确指定主机或组

1. 通过 --limit 在选定的组中明确指定主机：

```
# ansible ungrouped --limit srv1.example.com --list-hosts
hosts (1):
  srv1.example.com
```

2. 通过 --limit 参数，还可以指定一个文件，该文件中定义明确指定的主机的列表，定义一个 retry\_hosts.txt 如下：

```
srv1.example.com
```

再次执行ansible指令如下：

```
# ansible ungrouped --limit @retry_hosts.txt --list-hosts
hosts (1):
  srv1.example.com
```

后续学习中，在ansible-playbook命令行中也可以通过复杂的主机表达式来选定主机，不过需要使用-e参数来指定：ansible-playbook -e webserver:!(excluded)&required。不过这个用法并不常用。