简单说明

在通常情况下，所有的配置信息都会被作为ansible的变量保存了，而且可以保存在ansible允许定义变量的各种地方，诸如vars区段，`vars_files`加载的文件中，以及host_vars和group_vars目录中。

但在有些时候，我们希望从诸如文本文件或者.csv文件中收集数据作为ansible的变量，或者直接获取某些命令的输出作为ansible的变量，甚至从redis或者etcd这样的键值存储中取得相应的值作为ansible的变量。这个时候，我们就需要通过ansible的lookup插件来从这些数据源中读取配置数据，传递给ansbile变量，并在playbook或者模板中使用这些数据。

ansible支持一套从不同数据源获取数据的lookup，包括file, password, pipe, env, template, csvfile, dnstxt, redis_kv, etcd等

1. file

使用file lookup可以从文本文件中获取数据，并在这些数据传递给ansible变量，在task或者jinja2模板中进行引用。下面是一个从文本文件中获取ssh公钥并复制到远程主机的示例：

```
- name: copy authorized_host file
  template:
    src: authorized_keys.j2
    dest: /home/deploy/.ssh/authrized_keys
    owner: deploy
    group: deploy
    mode: 0600
```

authorized_keys.j2模板文件示例如下：

```
{{ lookup('file', '/users/breeze/.ssh/id_rsa.pub')}}
```

2. pipe

使用pipe lookup可以直接调用外部命令，并将命令执行的结果打印到标准输出，作为ansible变量。下面的例子通过pipe调用date指令拿到一个以时间数字组成的字串

```
- name: Flamingo | Get release version
  set_fact:
    flamingo_release_version: "{{ lookup('pipe', 'date +%Y%m%d%H%M%SZ') }}"
```

3. url

```
{{ lookup('url', 'http://xxx')}}
```

3. env

env lookup实际就是获取在控制主机上的某个环境变量的值。下面是一个读取控制机上 `$JAVA_HOME` 变量值的示例：

```
- name: get JAVA_HOME
  debug: msg="{{ lookup('env', 'JAVA_HOME')}}"
```

4. template

template lookup可以指定一个jinja2模板，然后返回这个模板中的变量被替换以后的结果。

假设我们有一个message.j2模板，内容如下：

```
This host runs {{ ansible_distribution }}
```

定义一个如下的task：

```
- name: print message from template
  debug: msg="{{ lookup('template', 'message.j2')}}"
```

输出的msg的结果如下：

```
This host runs CentOS
```

5. csvfile

csvfile可以从.csv文件中读取一个条目。假设我们有如下示例的名为users.csv的文件：

```
username,email
lorin,lorin@test.com
john,john@example.com
sue,sue@exmaple.com
```

下面是一个使用csvfile lookkup提取sue的电子邮件地址的task示例：

```
- name: get sue's email
  debug: msg="{{ lookup('csvfile','sue file=users.csv delimiter=, col=1')}}"
```

可以看到，一共向插件传递了四个参数：sue, file=users.csv, delimiter=,以及col=1。说明如下：

- 第一个参数指定一个名字，该名字必须出现在其所在行的第0列，需要说明的是，如果指定的第一个参数名字在文件中出现多次，则匹配第一次出现的结果
- 第二个参数指定csv文件的文件名
- 第三个参数指定csv文件的中条目的分隔符，
- 第四个参数指定要取得哪一列的值，这一列正是第一个参数所在行的那一列的值

如果我们想要查找的用户存储在名为username的变量中，则可以使用"+"符号来连接username字串和其他的参数字串，来构建完整的参数字符串：

```
lookup('csvfile', username+'file=users.csv' delimiter=, col=1)
```

6. redis_kv

`redis_kv lookup`可以直接从redis存储中来获取一个key的value，key必须是一个字符串，如同Redis GET指令一样。需要注意的是，要使用`redis_kv lookup`，需要在主控端安装python的redis客户端，在centos上，软件包为python-redis。

下面是一个在playbook中调用redis lookup的task，从本地的redis中取中一个key为weather的值：

```
- name: lookup value in redis
  debug: msg="{{ lookup('redis_kv', 'redis://localhost:6379,weather')}}"
```

其中URL部分如果不指定，该模块会默认连接到 `redis://localhost:6379`，所以实际上在上面的实例中，调用可以直接写成如下：

```
{{ lookup('redis_kv', 'weather')}}
```

7. etcd

etcd是一个分布式的key-value存储，通常被用于保存配置信息或者被用于实现服务发现。可以使用etcd lookup来从etcd中获取指定key的value。

我们通过如下方法往一个etcd中写入一个key：

```
curl -L http://127.0.0.1:2379/v2/keys/weather -XPUT -d value=sunny
```

定义一个调用etcd插件的task：

```
- name: look up value in etcd
  debug: msg="{{ lookup('etcd','http://127.0.0.1:2379,weather')}}"
```

默认情况下，etcd lookup会在http://127.0.0.1:2379上查找etcd服务器。但我们在执行playbook之前可以通过设置 `ANSIBLE_ETCD_URL` 环境变量来修改这个设置。

8. password

password lookup会随机生成一个密码，并将这个密码写入到参数指定的文件中。如下示例，创建一个名为bob的mysql用户，并随机生成该用户的密码，并将密码写入到主控端的bob-password.txt中：

```
- name: create deploy mysql user
  mysql_user: name=bob password={{ lookup('password', 'bob-password.txt')}} priv=*.*:ALL
state=present
```

9. dnstxt

dnstxt lookup用于获取指定域名的TXT记录。需要在主控端安装python-dns。

使用方法如下：

```
- name: lookup TXT record
  debug: msg="{{ lookup('dnstxt', "aliyun.com") }}"
```

如果某一个主机有多个相关联的TXT记录，那么模块会把他们连在一起，并且每次调用时的连接顺序可能不同