第 2 章 Chapter 2

Ansible 安装与配置

上一章已经介绍过 Ansible 配置管理系统由控制主机和被管节点组成。Ansible 对控制主机没有太多要求，当前控制主机的操作系统可以选用 Linux 或 OS X。被管节点的要求更少，支持 Linux、OS X、类 UNIX、Windows 等类型的主机节点，甚至支持 Cisco、Juniper 等网络设备、负载均衡器。

本章将着手准备 Ansible 的安装环境、安装 Ansible、了解 Ansible 基本配置、运行第一个测试应用。

2.1 Ansible 环境准备

安装前的准备工作包括思考如下一些内容：

- 从哪里获取安装软件包？
- 需要安装哪些支撑软件？
- 安装哪个版本的软件？
- 控制主机需要做哪些准备工作？
- 被管节点需要准备什么条件？

下面就分头讲讲。

1. 从 GitHub 获取 Ansible

Ansible 项目源码放在 GitHub 上管理，可以从 https://github.com/ansible/ansible 获取代码。也可以从这里了解到 Ansible 项目的最新进展，跟踪 Ansible 最新的思想和 bug 的处理情况。

2. 不需要安装支撑软件

Ansible 默认是基于 SSH 协议进行通信的。安装 Ansible 之后，控制主机不需要启动或运行任何 Ansible 的后台进程，也不需要有数据库。只要在一台控制主机上安装好，就可以通过这台主机管理一组远程节点。在远程被管节点上，同样也不需要安装、运行任何 Ansible 特有的软件。这样如果 Ansible 版本需要升级，只需升级控制主机，不涉及被管节点。

3. 选择开发版本

由于 Ansible 基于简单的源码运行，不必在被管节点上安装特有软件，因此很多用户会使用 Ansible 的开发版本。

Ansible 一般每两个月出一个发行版本。小 bug 一般在下一个发行版本中修复，并在稳定分支中用 backport（将一个软件的补丁应用到比此补丁所对应的版本更老的版本的行为）方式增加补丁。大 bug 将会在必要时出更新维护版本，这种情况很少出现。

希望使用 Ansible 最新的版本，并且使用的操作系统是 RHEL、CentOS、Fedora、Debian、Ubuntu 等，我们建议使用系统软件包管理工具安装。

另有一种选择是通过 pip 工具安装，pip 是一个安装和管理 Python 包的工具。

如果希望使用开发版本，需要使用、测试最新的功能特性，可以直接下载源码、运行，源码程序不需要安装过程，直接可以使用。

4. 准备控制主机

只要主机上安装了 Python 2.6 或以上版本，就可以运行 Ansible 配置工具，Windows 环境系统当前还不能作为控制主机。控制主机需要有下面这些组件：

- Python 2.6 或以上
- paramiko 模块

- PyYAML
- Jinja2
- httplib2

控制主机的系统可以是各种类 UNIX 操作系统，如 Red Hat、Debian、CentOS、OS X、BSD 等各种版本。

5. 查看被管节点

被管节点如果是类 UNIX 系统，则需要安装 Python 2.4 或以上版本。但如果版本低于 Python 2.5，则需要额外安装一个模块：python-simplejson 模块。Ansible 的 raw 模块和 script 模块是不需要 python-simplejson 模块支持的。从技术上讲，可以通过 Ansible 的 raw 模块安装 python-simplejson，之后就可以使用 Ansible 的所有功能了。

被管节点如果是 Windows，则需要有 PowerShell 3.0 并授权远程管理。

管理开启 SELinux 环境

如果被管节点上启用了 SElinux，需要安装 libselinux-python，这样才可使用 Ansible 中与 copy/file/template 相关的函数。可以通过 Ansible 的 yum 模块在需要的托管节点上安装 libselinux-python。

关于 Python 版本

Python 3 与 Python 2 是稍有不同的语言，而大多数 Python 程序还不能在 Python 3 中正确运行。而一些 Linux 发行版（Gentoo、Arch）默认没有安装 Python 2.X。在这些系统上，需要专门安装 Python 2.X 解释器，并把资源清单（inventory）中的 ansible_python_interpreter 变量设置为该 Python 2.X。当然，也可以使用 raw 模块在被管节点上远程安装 Python 2.X。

RHEL、CentOS、Fedora、Ubuntu 等发行版都默认安装了 2.X 的解释器，几乎所有的 UNIX 系统也预安装了。

为了方便读者理解，笔者通过虚拟化环境部署了两组业务功能服务器来进行演示。笔者的操作系统版本为 CentOS 7.0，自带 Python 2.7.5，将采用下一节介绍的 yum 方式安装。相关服务器信息如表 2-1 所示（CPU 核数及 Web 根目录的差异化便于演示生成动态配置）。

表 2-1 业务环境表

角色	主机名	IP 地址	组名	CPU（核数）	Web 根目录
控制主机	ansiblecontrol	192.168.1.100	---	---	---
被管节点	web1	192.168.1.111	webservers	2	/website
被管节点	web2	192.168.1.112	webservers	2	/website

本章后续的安装过程，也根据这个环境来进行。

2.2 安装 Ansible

Ansible 的安装方式非常灵活，满足各种环境部署的需求。一般可以直接用源码进行安装，也可用操作系统软件包管理工具进行安装，下面分别介绍。

2.2.1 直接用源码安装

1. 从 GitHub 源码库安装方式

很容易从 Ansible 项目的 GitHub 源码库提取出来安装，运行 Ansible 不需要 root 权限，也不依赖于其他软件，没有后台进程运行，不需要数据库支撑。不少社区用户直接使用 Ansible 的开发版本，这样可以利用最新的功能特性，也方便对项目进行测试。由于被管节点不需要安装任何软件，及时跟进、更新 Ansible 开发版相对于其他开源项目要容易得多。

从源码安装的过程如下：

1）提取 Absible 源代码：

```
$ git clone git://github.com/ansible/ansible.git --recursive
$ cd ./ansible
$ source ./hacking/env-setup
```

如果想要安装过程中减少告警 / 错误信息输出，可以在安装时加上 -q 参数：

```
$ source ./hacking/env-setup -q
```

2）如果系统没有安装过 pip，先安装对应 Python 版本的 pip：

```
$ sudo easy_install pip
```

3）安装 Ansible 控制主机需要的 Python 模块：

```
$ sudo pip install paramiko PyYAML Jinja2 httplib2 six
```

4）当更新 Ansible 版本时，不但要更新 git 的源码树，还要更新 git 中指向 Ansible 自身的模块，称为 submodules：

```
$ git pull --rebase
$ git submodule update --init --recursive
```

5）一旦运行 env-setup 脚本，就意味着 Ansible 从源码中运行起来了。默认的资源清单 inventory 文件是 /etc/ansible/hosts，清单文件 inventory 可以指定其他位置：

```
.. code-block:: bash
$ echo "127.0.0.1" > ~/ansible_hosts
$ export ANSIBLE_HOSTS=~/ansible_hosts
```

ANSIBLE_HOSTS 是 1.9 版本之后开始使用的，代替之前使用的 ANSIBLE_HOSTS。

可以在 3.1 节找到详细讲解 inventory 文件及使用的内容。

这样 Ansible 系统就安装完成了，后面就可以编写一些脚本开始对 Ansible 进行测试。

2. Tar 包安装方式

不想通过 GitHub 提取方式获得 Ansible 的软件包，可以直接下载 Ansible 的 Tar 包，下载地址是 http://releases.ansible.com/ansible。这里面存放着从 Ansible 最初发行的 1.1 版本开始，直到最新 1.9.2 版本，到现在共发行过的 42 个软件版本，可以根据需要下载。

Tar 包的安装过程与上述源码安装方式一样，只是源代码获取方式不同而已。

3. 制作 rpm 包安装方式

有时需要制作成 rpm 软件包再进行安装。在 GitHub 上 Ansible 项目中提取软件，或直接下载一个 Tar 包，然后使用 make rpm 命令创建 RPM 软件包，最后可分发这个软件包，或者使用它来安装 Ansible。在创建之前，先确定已安装了 rpm-build、make、python2-devel 组件。操作过程如下：

```
$ git clone git://github.com/ansible/ansible.git
$ cd ./ansible
$ make rpm
$ sudo rpm -Uvh ~/rpmbuild/ansible-*.noarch.rpm
```

在 Debian/Ubuntu 环境中，可以采用类似的方法制作安装包，制作的命令是 $ make deb。

2.2.2 用包管理工具安装

1. yum 安装方式

要使用 yum 方式安装，需要有合适的 yum 源。Fedora 用户只要连接着因特网，就可以直接使用官方的 yum 源安装。但对于 RHEL、CentOS 的官方 yum 源中没有 Ansible 安装包，这就需要先安装支持第三方的 yum 仓库组件，最常用的有 EPEL、Remi、RPMForge 等。在国内速度较快的高质量 yum 源有中国科技大学（http://mirrors.ustc.edu.cn）、浙江大学（http://mirrors.zju.edu.cn/epel/）、上海交通大学（http://ftp.sjtu.edu.cn/fedora/epel/）、网易 163（http://mirrors.163.com）、sohu 镜像源（http://mirrors.sohu.com/fedora-epel/）等。

下面安装 EPEL 作为部署 Ansible 的默认 yum 源。

- RHEL（CentOS）5 版本：

```
rpm -Uvh http://mirrors.zju.edu.cn/epel/6/i386/epel-release-5-4.noarch.
    rpm
rpm -Uvh http:// mirrors.zju.edu.cn/epel/5/x86_64/epel-release-5-4.
    noarch.rpm
```

- RHEL（CentOS）6 版本：

```
rpm -Uvh http:// mirrors.zju.edu.cn/epel/6/x86_64/epel-release-6-8.
    noarch.rpm
rpm -Uvh http:// mirrors.zju.edu.cn/epel/6/i386/epel-release-6-8.noarch.
    rpm
```

- RHEL（CentOS）7 版本：

```
rpm -Uvh http:// mirrors.zju.edu.cn/epel/7/x86_64/e/epel-release-7-5.
    noarch.rpm
```

准备好 yum 源之后，Ansible 就可直接用 yum 命令安装了，命令如下：

```
$ sudo yum install ansible
```

2. Apt (Ubuntu) 安装方式

Ubuntu 编译版可在如下地址中获得：https://launchpad.net/ ～ ansible/+archive/ansible。

通过执行如下命令直接安装：

```
$ sudo apt-get install software-properties-common
$ sudo apt-add-repository ppa:ansible/ansible
$ sudo apt-get update
$ sudo apt-get install ansible
```

3. Homebrew (Mac OSX) 安装方式

在 Mac 系统中，确定已经安装了 Homebrew 之后，直接执行下面命令安装 Ansible：

```
$ brew update
$ brew install Ansible
```

4. pip 方式安装

Ansible 也支持可通过 pip 方式安装。pip 是 Python 软件包的安装和管理工具，执行如下命令先安装 pip：

```
$ sudo easy_install pip
```

然后再安装 Ansible：

```
$ sudo pip install ansible
```

如果你是在 OS X 系统上安装，编译器可能会有警告或出错，需要设置 CFLAGS、CPPFLAGS 环境变量：

```
$ sudo CFLAGS=-Qunused-arguments CPPFLAGS=-Qunused-arguments pip install
    ansible
```

使用 virtualenv 的读者可通过 virtualenv 安装 Ansible，然而不建议这样做，直接在全局安装 Ansible。不要使用 easy_install 直接安装 Ansible。

如果把 Ansible 安装在其他相对少见的 Linux 操作系统（如 Gentoo、FreeBSD 等）上，详见 Ansible 官网 http://docs.ansible.com。

2.3 配置运行环境

2.3.1 配置 Ansible 环境

Ansible 配置文件是以 ini 格式存储配置数据的，在 Ansible 中，几乎所有的配置项都可以通过 Ansible 的 playbook 或环境变量来重新赋值。在运行 Ansible 命令时，命令将会按照预先设定的顺序查找配置文件，如下所示：

1）ANSIBLE_CONFIG：首先，Ansible 命令会检查环境变量，及这个环境变量将指向的配置文件。

2）./ansible.cfg：其次，将会检查当前目录下的 ansible.cfg 配置文件。

3）～ /.ansible.cfg：再次，将会检查当前用户 home 目录下的 .ansible.cfg 配置文件。

4）/etc/ansible/ansible.cfg：最后，将会检查在用软件包管理工具安装 Ansible 时自动产生的配置文件。

注意 如果你通过操作系统软件包管理工具或 pip 安装，那么你在 /etc/ansible 目录下应该已经有了 ansible.cfg 配置文件；如果你是通过 GitHub 仓库安装的，在你复制的仓库中 examples 目录下可以找到 ansible.cfg，你可以把它拷贝到 /etc/ansible 目录下。

1. 使用环境变量方式来配置

大多数的 Ansible 参数可以通过设置带有 ANSIBLE_ 开头的环境变量进行配置，参数名称必须都是大写字母，如下配置项：

```
export ANSIBLE_SUDO_USER=root
```

设置了环境变量之后，ANSIBLE_SUDO_USER 就可以在 playbook 中直接引用。

2. 设置 ansible.cfg 配置参数

Ansible 有很多配置参数，你也许不会都使用到。下面列出常用的配置参数：

- inventory——这个参数表示资源清单 inventory 文件的位置，资源清单就是一些 Ansible 需要连接管理的主机列表。在 1.9 版本之前有个类似功能的参数 hostfile，但 1.9 版本之后就不建议再使用了。下一章将对资源清单做详细的讲

解。这个参数的配置实例如下：

```
inventory = /etc/ansible/hosts
```

- library——Ansible 的操作动作，无论是本地或远程，都使用一小段代码来执行，这小段代码称为模块，这个 library 参数就是指向存放 Ansible 模块的目录。配置实例如下：

```
library = /usr/share/ansible
```

Ansible 支持多个目录方式，只要用冒号（：）隔开就可以，同时也会检查当前执行 playbook 位置下的 ./library 目录。

- forks——设置默认情况下 Ansible 最多能有多少个进程同时工作，默认设置最多 5 个进程并行处理。具体需要设置多少个，可以根据控制主机的性能和被管节点的数量来确定，可能是 50 或 100，默认值 5 是非常保守的设置。配置实例如下：

```
forks = 5
```

- sudo_user——这是设置默认执行命令的用户，也可以在 playbook 中重新设置这个参数。配置实例如下：

```
sudo_user = root
```

- remote_port——这是指定连接被管节点的管理端口，默认是 22。除非设置了特殊的 SSH 端口，不然这个参数一般是不需要修改的。配置实例如下：

```
remote_port = 22
```

- host_key_checking——这是设置是否检查 SSH 主机的密钥。可以设置为 True 或 False，下一节将详细介绍。配置实例如下：

```
host_key_checking = False
```

- timeout——这是设置 SSH 连接的超时间隔，单位是秒。配置实例如下：

```
timeout = 60
```

- log_path——Ansible 系统默认是不记录日志的，如果想把 Ansible 系统的输出记录到日志文件中，需要设置 log_path 来指定一个存储 Ansible 日志的文件。配置实例如下：

```
log_path = /var/log/ansible.log
```

另外需要注意，执行 Ansible 的用户需要有写入日志的权限，模块将会调用被管节点的 syslog 来记录，口令是不会出现在日志中的。

2.3.2 使用公钥认证

现在运维对安全要求都是比较重视的，一般会采用密钥验证方式来登录，Ansible 1.2.1 之后的版本都默认启用公钥认证。

如果有台被管节点重新安装系统并在 known_hosts 中有了与之前不同的密钥信息，就会提示一个密钥不匹配的错误信息，直到被纠正为止。在使用 Ansible 时，如果有台被管节点没有在 known_hosts 中被初始化，将会在使用 Ansible 或定时执行 Ansible 时提示对 key 信息的确认。

如果你不想出现这种情况，并且你明白禁用此项行为的含义，只要修改 home 目录下～ /.ansible.cfg 或 /etc/ansible/ansible.cfg 的配置项：

```
[defaults]
host_key_checking = False
```

或者直接在控制主机的操作系统中设置环境变量，如下所示：

```
$export ANSIBLE_HOST_KEY_CHECKING=False
```

需要说明的是，在早期使用 paramiko 模式时，公钥认证速度相当慢，因此当使用密钥认证方式时建议采用 SSH 方式连接，这也是现在默认的连接方式。

2.3.3 配置 Linux 主机 SSH 无密码访问

为了避免 Ansible 下发指令时输入目标主机密码，通过证书签名达到 SSH 无密码是一个好的方案。推荐使用 ssh-keygen 与 ssh-copy-id 来实现快速证书的生成及公钥下发，其中 ssh-keygen 生产一对密钥，使用 ssh-copy-id 来下发生成的公钥。具体操作如下。

在控制主机（ansiblecontrol）上创建密钥，执行：ssh-keygen –t rsa，有询问直接按回车键即可，将在 /root/.ssh/ 下生成一对密钥，其中 id_rsa 为私钥，id_rsa.pub 为公钥，代码如下：

```
[root@ansiblecontrol]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):< 回车 >
Enter passphrase (empty for no passphrase): < 回车 >
Enter same passphrase again: < 回车 >
```

```
    Your identification has been saved in /root/.ssh/id_rsa.
    Your public key has been saved in /root/.ssh/id_rsa.pub.
    The key fingerprint is:
    ec:f2:89:2a:62:c5:1a:53:32:04:04:fd:40:e9:ad:dd root@ansiblecontrol.
ansible.cn
    The key's randomart image is:
    +--[ RSA 2048]-----+
    |*+..              |
    | .+               |
    |.. +              |
    | o..o  .          |
    |  =o .  S         |
    | o.o. E.          |
    |  =   . .         |
    |.o.    + .        |
    |.. .... o         |
    +------------------+
```

下发密钥就是控制主机把公钥 id_rsa.pub 下发到被管节点上用户下的 .ssh 目录，并重命名成 authorized_keys，且权限值为 400。接下来推荐常用的密钥拷贝工具 ssh-copy-id，把公钥文件 id_rsa.pub 公钥拷贝到被管节点，命令格式如下：

```
    ssh-copy-id [-h|-?|-n] [-i [identity_file]] [-p port] [[-o <ssh -o
options>] ...] [user@]hostname
```

输入以下命令同步公钥到被管节点 web1（192.168.1.111）：

```
    [root@ansiblecontrol]# ssh-copy-id -i /root/.ssh/id_rsa.pub
        root@192.168.1.111
    /usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s),
         to filter out any that are already installed
    /usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you
         are prompted now it is to install the new keys
    root@192.168.1.111's password:< 输入口令 >

    Number of key(s) added: 1

    Now try logging into the machine, with:   "ssh 'root@192.168.1.111'"
    and check to make sure that only the key(s) you wanted were added.

    [root@ansiblecontrol]# ssh root@192.168.1.111
    Last login: Sat Aug 29 12:07:45 2015 from ansiblecontrol.ansible.cn
```

```
[root@web1 ~]#
```

密钥分发后，需要验证一下 SSH 无密码配置是否成功，只需运行 ssh root@192.168.1.111，如果直接进入被管节点 root 账号提示符，即出现 [root@web1 ~]#，则说明配置成功。

同样的配置方式对 web2（192.168.1.112）也做下密钥分发。

2.4 Ansible 小试身手

Ansible 安装完成之后，我们也了解了基本配置方式，是不是已经急切要动手实践一下 Ansible？下面我们将介绍主机连通性测试和远程执行命令两个小场景，体会一下 Ansible 的便捷、强大。

首先可以先查看一下安装的 Ansible 软件版本信息：

```
$ansible --version
ansible 1.9.2
configured module search path = None
```

2.4.1 主机连通性测试

为了使用 Ansible，第一步需要修改主机与组配置，默认的文件在 /etc/ansible/hosts 了，格式为 ini，添加两台主机的 IP 地址，同时定义一个 webservers 组包含这两个 IP 地址，内容如下：

```
【/etc/Ansible/hosts】
#web1.ansible.cn
#web2.ansible.cn
192.168.1.101
192.168.1.102

[webservers]
#web1.ansible.cn
#web2.ansible.cn
192.168.1.101
192.168.1.102
```

然后，用 ping 模块对单台主机进行 ping 操作，结果如下：

```
$ ansible 192.168.1.111 -m ping
192.168.1.111 | success >> {
    "changed": false,
    "ping": "pong"
}
```

对 webservers 组进行 ping 操作，结果如下：

```
$ ansible webservers -m ping
192.168.1.111 | success >> {
    "changed": false,
    "ping": "pong"
}

192.168.1.112 | success >> {
    "changed": false,
    "ping": "pong"
}
```

出现上述结果表示 Ansible 正确工作，主机的连通性测试成功，没有对被管节点做任何变更。

这里测试时在控制主机与被管节点之间配置了 SSH 证书信任。如果没有用证书认证，则需要在执行 Ansible 命令时添加 -k 参数，在提示“SSH password:”时输入 root（默认）账号密码。实际生产环境中，大多数更倾向于使用 Linux 普通用户账户进行连接并通过 sudo 命令实现 root 权限，格式为：

```
ansible webservers -m ping -u ansible -sudo
```

2.4.2 在被管节点上批量执行命令

Ansible 很方便进行运维自动化，是运维与研发合作的重要工具。我们借鉴学习开发语言的经典示范程序，也用 Hello World 程序作为自动化运维环境的测试、校验手段。

在用户 home 目录下创建一个资源清单文件 inventory.cfg，内容如下：

```
$cat inventory.cfg
[webservers]
192.168.1.101
192.168.1.102
```

用 Ansible 的 shell 模块在 webservers 组的各服务器上显示“hello ansible！”，命令如下：

```
$ ansible webservers -m shell -a '/bin/echo hello ansible!' -i
    inventory.cfg
192.168.1.111 | success | rc=0 >>
hello ansible!

192.168.1.112 | success | rc=0 >>
hello ansible!
$
```

也可以用 command 模块，得到类似的结果：

```
$ ansible webservers -m command -a '/bin/echo hello ansible!' -i
    inventory.cfg
192.168.1.111 | success | rc=0 >>
hello ansible!
192.168.1.112 | success | rc=0 >>
hello ansible!
```

这样简单的命令就可以完成批量服务器的管理。现在，你已经来到通往 Ansible 自动化运维的门口，但是真正巧妙、灵活、功能强大的 Ansible 等待着你深入学习、探索。

2.5 获取帮助信息

在 Ansible 1.9.2 版本中有 8 个主要的 Ansible 管理工具，每个管理工具都是一系列的模块、参数支持。随时可获取的帮助信息对了解掌握 Ansible 系统非常重要。对于 Ansible 每个工具，都可以简单地在命令后面加上 -h 或 --help 直接获取帮助。

例如，列出 ansible-doc 工具的支持参数。最主要的参数 -l 列出可使用的模块，-s 列出某个模块支持的动作，如下所示：

```
[root@ansiblecontrol]# ansible-doc -h
Usage: ansible-doc [options] [module...]

Show Ansible module documentation

Options:
  --version             show program's version number and exit
```

```
  -h, --help            show this help message and exit
  -M MODULE_PATH, --module-path=MODULE_PATH
                        Ansible modules/ directory
  -l, --list            List available modules
  -s, --snippet         Show playbook snippet for specified module(s)
  -v                    Show version number and exit
```

用 ansible-doc –l 列出 Ansible 系统支持的模块，Ansible 安装后能够列出 259 个模块，如下所示

```
$ ansible-doc -l
less 458 (POSIX regular expressions)
Copyright (C) 1984-2012 Mark Nudelman
less comes with NO WARRANTY, to the extent permitted by law.
For information about the terms of redistribution,
see the file named README in the less distribution.
Homepage: http://www.greenwoodsoftware.com/less
a10_server            Manage A10 Networks AX/SoftAX/Thunder/vThunder...
a10_service_group     Manage A10 Networks AX/SoftAX/Thunder/vThunder...
a10_virtual_server    Manage A10 Networks AX/SoftAX/Thunder/vThunder...
acl                   Sets and retrieves file ACL information.
add_host              add a host (and alternatively a group) to the ...
airbrake_deployment   Notify airbrake about app deployments
alternatives          Manages alternative programs for common comman...
apache2_module        enables/disables a module of the Apache2 webse...
apt                   Manages apt-packages
apt_key               Add or remove an apt key
.........
```

ansible-doc 直接加模块名称，将显示该模块的描述和使用示例，如 ansible-doc ping。每个模块都有一系列的动作，可以用 ansible-doc –s+ 模块名称列出，如下面列出 yum 模块的动作：

```
$ ansible-doc -s yum
less 458 (POSIX regular expressions)
Copyright (C) 1984-2012 Mark Nudelman
less comes with NO WARRANTY, to the extent permitted by law.
For information about the terms of redistribution,
see the file named README in the less distribution.
Homepage: http://www.greenwoodsoftware.com/less
- name: Manages pack age swith the I (yum) pac
```

```
action: yum
    conf_file         # The remote yum configuration file to use
                        for the
    disable_gpg_check # Whether to disable the GPG checking of
                        signatures
    disablerepo       # 'Repoid' of repositories to disable for the
                        insta
    enablerepo        # 'Repoid' of repositories to enable for the
                        instal
    list              # Various (non-idempotent) commands for usage
                        with
    name=             # Package name, or package specifier with
                        version,
    state             # Whether to install (`present', `latest'),
                        or remo
    update_cache      # Force updating the cache. Has an effect
                        only if s
```

另外，在 Ansible 调试自动化脚本时候经常需要获取执行过程的详细信息，可以在命令后面添加 -v 或 -vvv 得到详细的输出结果。如：

```
$ansible webservers -i inventory.cfg -m ping -vvv
```

最后别忘了经常浏览官方网站 http://docs.ansible.com/，其中有详细的使用说明。

2.6 本章小结

本章讲解 Ansible 在不同环境下的各种安装方式，以及安装完成之后对系统进行必要的参数配置，对安装后 Ansible 的基本测试，最后介绍如何获取 Ansible 帮助的工具。到这里你应该已经搭建好了基本的 Ansible 测试环境，为后续深入学习 Ansible 提供了必要的基本环境。

下一章将详细介绍 Ansible 涉及的一些基础概念，如何对资源清单进行管理，引入变量、匹配模式等内容。