
Ansible UI 平台 用户手册

Alex li

liyang23456@163.com

版本控制

版本	日期	描述（修改原因）	作者（修改者）
V1.0	2014-08-11	初稿	Alex li

目录

1. 概览.....	4
1.1 Ansible UI.....	4
1.2 需求	4
1.3 已知问题	4
1.4 版本历史	5
2. 开始.....	6
2.1 安装	6
2.2 快速开始	7
3. 用户手册.....	13
3.1 登录	13
3.2 用户	13
3.3 项目	14
3.4 权限	15
3.5 主机管理	16
3.6 部署密钥	17
3.7 数据	18
3.8 变量	20
3.9 任务	21
3.10 任务模板	24
3.11 预约管理	25
3.12 日志	25
3.13 执行脚本	25
3.14 传输文件	26
3.15 持续集成	27
4. 附录.....	28
4.1 使用 LDAP 认证	28
4.2 后台管理	29

1. 概览

1.1 Ansible UI

Ansible UI 是集程序部署、配置管理和持续集成于一体的配置管理平台。该平台基于 ansible 系统，这是一个开源的 IT 配置管理引擎，提供了强大的工具来实现 Linux 系统的自动化操作。

- 一键部署

通过简单的几个点击，实现项目的快速部署。你可以自由的组织项目、主机、任务、变量，能够查看任务的执行结果。

- 配置管理

实现 Linux 系统的环境配置，保障程序运行所需要的统一环境。

- 持续集成

与 jenkins 系统集成，可以自动从 jenkins 平台获取程序包。

- 角色权限控制

对不同的用户提供分级的角色，赋予查看、配置、执行、管理四级权限，保障程序和系统的安全。

- 预约执行

提供预约执行和邮件通知功能，可以轻松的搞定午夜上线啦。

1.2 需求

最小需求如下：

- OS: Redhat Enterprise Linux 6 or CentOS 6
- 2 Core
- 2GB RAM
- 20GB Hard Disk

1.3 已知问题

- 在单核 CPU 服务器上任务无法运行(在调用 pexpect 时会报错“ValueError:

I/O operation on closed file”),如果要在单核 CPU 服务器上运行，请将 pexpect 降为 2.4 版本

1.4 版本历史

版本	日期	描述（修改原因）	作者（修改者）
V1.0	2014-08-11	正式版 1.0	Alex li

2. 开始

2.1 安装

(1) Requirements

- Pip
- Virtualenv
- Mysql-server, mysql-devel
- Openldap-devel

(2) Install

- 添加系统用户

```
useradd ansible
su - ansible
```

- 配置 virtualenv 环境

```
virtualenv envansible
source envansible/bin/active
```

- 下载源码

```
git clone https://github.com/alaxli/ansible_ui.git
```

- 安装依赖库

```
cd ansible-ui
pip install -r requirements.txt
pip install PIL --allow-external PIL --allow-unverified PIL
```

- 配置 ldap、数据库和邮件服务器信息

```
cd desktop/core/internal
vim settings_local.py
# 修改 LDAP Database Mail 和 ansible_playbook 命令位置(which ansible_playbook)配置
如果需要使用 ldap, 还需要修改 settings.py, 去掉下面行的注释
# 'desktop.core.auth.backend.LdapBackend',
```

- 配置数据库

```
create database ansible CHARACTER SET utf8;
grant all on ansible.* to ansibleuser@'localhost' identified by '*****';
```

- 初始化数据库

```
python manage.py schemamigration desktop.apps.account --init
python manage.py schemamigration desktop.apps.ansible --init
python manage.py syncdb
```

```
python manage.py migrate ansible
python manage.py migrate account
python manage.py migrate kombu.transport.django
python manage.py migrate djcelery
python manage.py migrate guardian
```

- 配置 celery

```
修改 celery-conf/supervisord.conf
[inet_http_server] #配置 web 管理 supervisor
[program:ansible_celeryd] #修改 command 中 virtualenv 和 ansible_ui home
```

- 启动 celery

```
supervisord -c celery-conf/supervisord.conf
```

- 配置 ansible

```
cp ansible-conf/ansible.cfg ~/.ansible.cfg
```

(3) Run

- 直接运行

```
python manage.py runserver ip:8000
```

- Apache+wsgi

```
修改 apache-conf/ansible.cfg : ansible_ui_dir, 指向实际目录
修改 django.wsgi : yourvirtualenv 指向实际目录
拷贝 apache-conf/ansible.cfg 到 apache 配置目录下
重启 httpd
```

2.2 快速开始

这里我们将快速的实现一个任务，该任务很简单，执行一个 ping 操作，并返回结果。
通过这个任务，可以快速感受一下 Ansible UI 平台的简单易用，并验证平台的正确。

任务摘要如下：

- 登录
- 生成密钥
- 配置用户密钥
- 创建项目
- 配置主机
- 部署密钥
- 配置任务

- 执行任务

准备好了么，让我们开始！

(1) 登录

访问 http://your_ansible_ui_url ,输入用户名密码（执行 `python manage.py syncdb` 时创建）

Demo 地址: <http://115.28.87.99:8888> ， 用户名: admin, 密码: ansible



(2) 生成 SSH 密钥

```
$ssh-keygen -t dsa -b 1024
Generating public/private dsa key pair.
Enter file in which to save the key (/root/.ssh/id_dsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_dsa.
Your public key has been saved in /root/.ssh/id_dsa.pub.
The key fingerprint is:
e5:f5:b6:5b:63:fc:2d:12:7e:99:16:d8:75:86:d0:0f root@vm-10-154-252-57
```

命令生成一套密钥文件 `id_dsa`（私钥）和 `id_dsa.pub`（公钥）。

(3) 配置用户密钥

点击 Ansible UI 平台右上角“我的信息”，点击“证书”，将（2）中生成的私钥文件内容粘贴在“SSH Private Keys(DSA)”处，生成密钥时如果设置了解析密码，则输入 SSH Password，没有则留空，点击“save”按钮保存。

admin的证书

信息

证书

SSH Password

if modify ssh key, re-input password

SSH Private Keys (DSA)

```
-----BEGIN DSA PRIVATE KEY-----
MIIBvAIBAABgQDwdtTHvXM0fSTnRCSzS44CKwFrOdXZLSwt
5CvPMdkiyhn3NIIKoPMgKrrqkUWQmDbjRCkyOj98xgVHWDyJt
4tJnch/ofFXzZr0Dmujrp1nw6JonLAvOotAtasMs5Hkl+CvDbxsqI
qX+c10JeUdbA4mWPtaf6nznPNAoGAJa88OEebPYYOzsURpzs4
m6KnQyzFd54GW7EQRy7u+u3Blq3ozYw1Oe2EC5uFDIGrYs
/kPxJvV7AC2Zxu3UTL+G6+vtyF8H6eiWVE29K0sNfv4B8Rzp5f
Ca1PFCECgYEA32d0UhdMzC7Qcmys9iBafPyICDf76kxrCsRF
ojkT8mqroc8wB6yUvZX4APZ4ZmVphNYCvMwQ9kBNosIjm2Hc
W6/o4TJ95RyjckBfa8HBh9rtV3pCSJ5iotPYOkSpHAJXMGXWfe
```

Save

通常情况下，用户证书初次配置即可，以后无需配置。只有当登录服务器的公钥文件发生变更时，才需要重新配置用户密钥。

(4) 创建项目

点击“项目”菜单右上角的“新的项目”，我们建立一个测试项目 **test**。

新的项目

项目名称

test

描述

test

SCM类型

NONE

SCM URL

SCM URL ...

组

group name ...

提交

(5) 配置主机

在项目列表中找到 **test** 项目，点击项目名称进入。

点击“通用管理”面板中的“主机管理”，我们来加入一台主机。



(6) 部署密钥

添加完了主机，我们需要在主机上配置公钥（步骤 2 中生成的.pub 文件），这样就能通过平台 ssh 登录到目标主机。

点击“通用管理”面板中的“部署密钥”，按照下图进行配置。

test / 部署密钥

选择资产: hosts

主机: test

用户: root

密码:

SSH Public Key: ssh-dss
AAAAB3NzaC1kc3MAAACBAN5bskW9zMDQ7AaCdEWNuo15YuBaWVbCKolb78egHfkgSuKdpVTfrfP7dXzv+5ww+0XwST6SdJDpJ1osepd+mSI9gQ7On1dq6QVDTMQ0IOHBJzRr+5NVUS5qW2BNaSZ/FSLiNhGMqh6EIUjtuNMbEerXtgQAAAIbqbOt7dCKmHaYdDYqItgMvUS2uLTTp3eX2sGZGWD7HUnl5tLnHzi/nxnCQfciS/SJZDQ9minUXDih6eURUHl8PUS8csaXdmrQoHC0qa6H6mg5rZQS6iSlgMH3G9FWLa

提交

执行结果: copying key to root@ :~/.ssh/authorized_keys... SUCCESS!

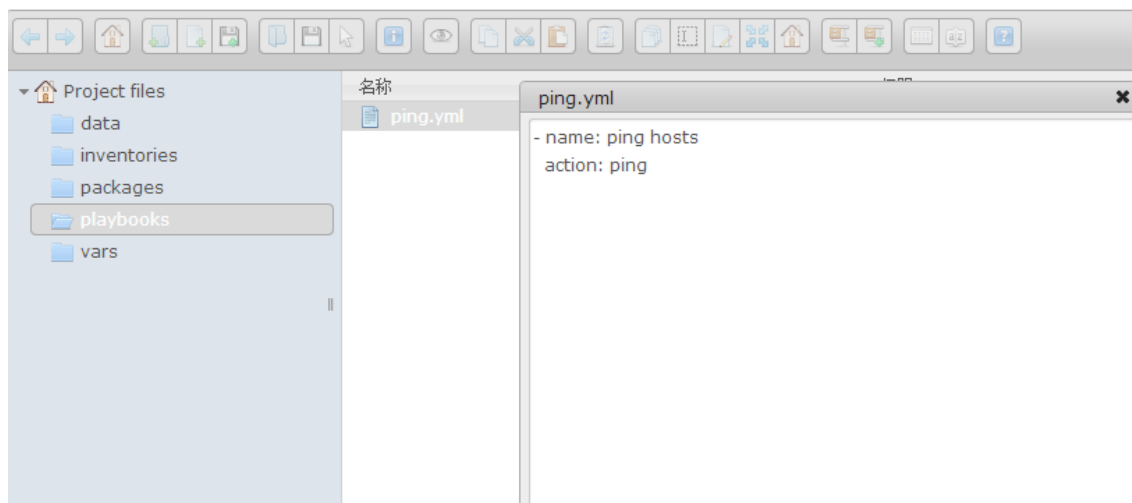
其中，主机部分即为步骤 5 中配置的组名或者直接输入 IP 也可以。用户和密码为主机上系统账号和密码。点击“提交”，在执行结果中进行查看。

(7) 配置任务

点击“通用管理”面板中的数据管理，进入目录浏览。目录浏览包含了所有的任务配置（包括主机配置），可以像操作系统目录一样进行操作。

打开 playbooks 目录，新建一个文件 ping.yml，输入以下内容：

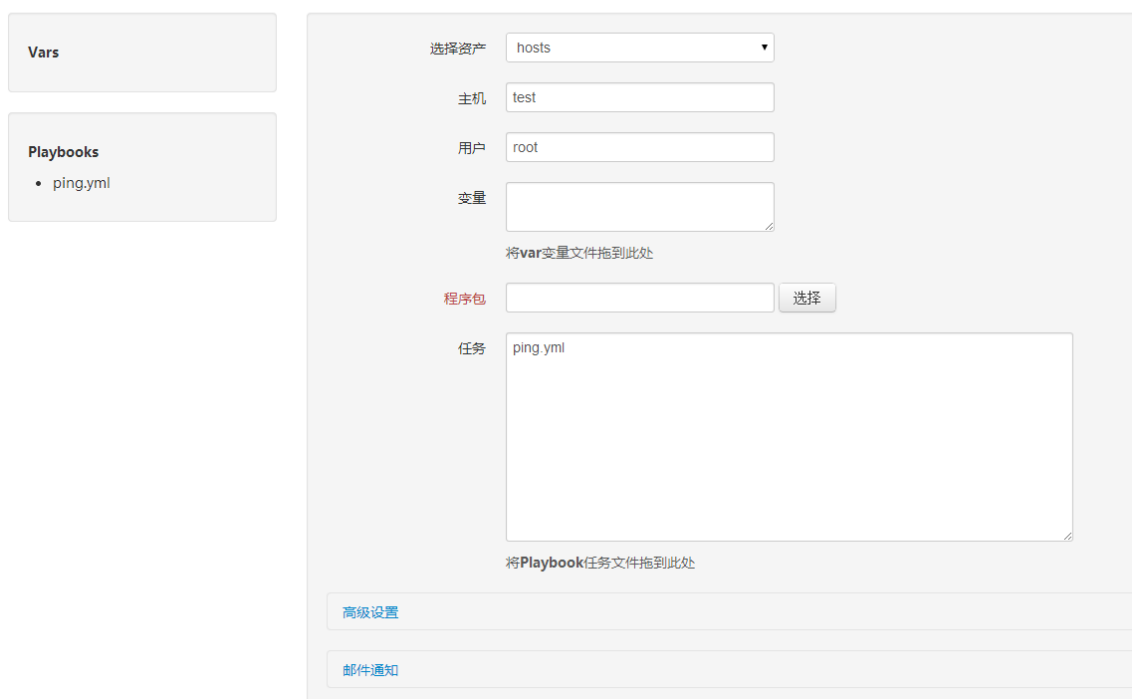
```
- name: ping hosts
  action: ping
```



保存，我们就有了第一个任务配置。

(8) 执行任务

点击“执行”面板中的“部署任务”按钮，进行任务部署。



可以通过“拖拽”操作，将 `playbooks` 拉入任务列表中，点击“执行”按钮，跳转结果页面，状态：`successful`。

**test / 执行结果**状态: **successful****stdout**

```
PLAY [test] *****

GATHERING FACTS *****
ok: [localhost]
ok: [10.163.177.163]

TASK: [ping hosts] *****
ok: [10.163.177.163]
ok: [localhost]

PLAY RECAP *****
10.163.177.163      : ok=2    changed=0    unreachable=0    failed=0
localhost          : ok=2    changed=0    unreachable=0    failed=0
```

至此，我们已经完成了一个简单的任务，并验证了平台的正确运行。

3. 用户手册

3.1 登录

访问 http://your_ansible_ui_url ,输入用户名密码（执行 `python manage.py syncdb` 时创建）

Demo 地址: <http://115.28.87.99:8888> , 用户名: admin, 密码: ansible



目前，登录方式提供 LDAP 方式和数据库方式。

3.2 用户

用户登录后，在执行任务前，需要首先配置自己的密钥。

点击 Ansible UI 平台右上角“我的信息”，点击“证书”，将密钥中的私钥文件内容粘贴在“SSH Private Keys (DSA)”处，生成密钥时如果设置了解析密码，则输入 SSH Password，没有则留空，点击“save”按钮保存。生成密钥的过程请参考 2.2 节的第（2）条。

admin的证书

信息

证书

SSH Password

if modify ssh key, re-input password

SSH Private Keys (DSA)

-----BEGIN DSA PRIVATE KEY-----
MIIBvAIBAABgQDwdtTHvXM0fSTnRCSzS44CKwFrOdXZLSwt
5CvPMdkiyhn3NIIKoPMgKrrqkUWQmDbjRCkyOj98xgVHWDyJt
4tJnch/ofFXzZr0Dmujrp1nw6JonLAvOotAtasMs5Hkl+CvDbxsqI
qX+c10JeUdbA4mWPtaf6nznPNAoGAJa88OEebPYYOzsURpzs4
m6KnQyzFd54GW7EQRy7u+u3Blq3ozYw1Oe2EC5uFDIGrYs
/kPxJvV7AC2Zxu3UTL+G6+vtyF8H6eiWVE29K0sNfv4B8Rzp5f
Ca1PFCECgYEA32d0UhdMZC7Qcmys9iBafPylCDf76kxrCsRF
ojkT8mqroc8wB6yUvZX4APZ4ZmVphNYCvMwQ9kBNosIjm2Hc
W6/o4TJ95RyjckBfa8HBh9rtV3pCSJ5iotPYOkSpHAJXMGXWfe

Save

通常情况下，用户密钥初次配置即可，以后无需配置。只有当登录服务器的公钥文件发生变更时，才需要重新配置用户密钥。

3.3 项目

点击 Ansible UI 平台“项目”菜单，点击页面右上角的“新的项目”，进入创建项目页面。

项目名称

test

描述

test

SCM类型

NONE

SCM URL

SCM URL ...

组

测试

提交

其中，如果 SCM 类型不为 NONE，SCM URL 的内容需要唯一，SCM URL 可以不是实际的 scm 地址，它的作用是与 jenkins 平台通信时识别项目的标识，所以必须是唯一的。

如果配置了组，则在项目列表页中会有组标签，点击进入后，可查看同组的所有项目。

项目

<div>所有 测试</div>				
项目名称	描述	SCM类型	创建者	操作
test	test	NONE	admin	编辑 删除 管理

点击“编辑”按钮，可以编辑项目信息。

点击“删除”按钮，将删除项目。删除前会弹出对话框确认。

点击“管理”按钮，则可以对用户赋予访问项目等相关权限（见 3.4 节）。

3.4 权限

Ansible UI 平台提供了 4+1 的权限管理方式，即 4 个项目级权限和 1 个 SuperAdmin 权限。

- 访问：能够查看项目的基本信息，包括主机、任务 yml 文件、任务模板和 log。
- 配置：能够配置主机、任务文件、数据文件，保存任务模板配置、删除模板。
- 执行：能够部署任务、执行脚本、传输文件，部署密钥，管理预约任务，重启任务。
- 管理：能够修改、删除项目，控制项目的用户权限。
- SuperAdmin：超级管理员，能够访问管理后台（见 5.2），管理所有项目、用户和任务信息，通常仅平台管理员拥有该权限。

用户创建项目的同时，即获得了该项目的管理权限，可以增加、删除用户，并对用户权限进行管理，因而可以轻松实现项目的“自治”管理。

项目权限配置页面如下：

test / 管理项目

Security

如果项目可以开放公共访问权限，可以添加AnonymousUser用户

AnonymousUser用户可以拥有访问和配置权限，不能拥有执行和管理权限

用户	访问	配置	执行	管理	所有
AnonymousUser	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

需要注意一点：权限之间不是继承关系，是相互独立的，比如如果只有管理权限，而没有访问权限，是无法访问项目的。

用户需要首先在平台存在(ldap 账号需登录一次)，才能被识别，进行权限添加等操作。

另外，系统提供了一个公共账号 AnonymousUser，如果想开放公共访问权限，可以添加 AnonymousUser 用户，该用户可以拥有访问和配置权限，不能拥有执行和管理权限，即使勾选也不会生效。

3.5 主机管理

主机管理是配置需要管理的客户机，管理有两个入口，如下图，分别为主机管理和浏览。



“主机管理”按钮提供直接的 hosts 文件管理，且只管理 hosts 这一个文件（实际上主机文件可以多个），可以满足基本的使用。

“浏览”按钮提供了文件夹式的管理，inventories 目录即为存放主机文件的地方，可以编辑包括 hosts 资产文件在内的文件。

主机管理文件使用 ini 格式。基本的用法为 hosts 和 groups，例如：

```
[test]
10.204.8.228

[product]
10.200.10.10

[all:children]
test
product
```

另外，主机资产管理还提供了 host 变量、group 变量以及动态主机文件等高级功能，可以访问 http://docs.ansible.com/intro_inventory.html 进行查看。

3.6 部署密钥

Ansible UI 平台通过 SSH 密钥的方式来访问客户机。因此，在执行任务前，需要先向客户机部署公钥，以达成访问条件。

通过“部署密钥”功能模块来进行。



在部署密钥操作执行前，需要首先生成一套免检权密钥，过程参考 2.2 节的第 (2) 条，这里假定已经拥有一套密钥。

配置选项中，资产和主机分别为 3.5 节中配置的主机文件和组，主机也可以直接添加主机名或者 IP，多个主机之间用“；”隔开。用户和密码分别为客户机的系统账号和密码，SSH Public Key 为密钥对中的公钥内容。点击“提交”后，执行结果在当前页显示。

test / 部署密钥

选择资产 hosts

主机 test

用户 root

密码

SSH Public Key

```
ssh-dss
AAAAB3NzaC1kc3MAAACBAN5bskW9zMDQ7AaCdEWNuo15YuBaWwVbCKolb78egHfkgSuKdpVTrf
P7dXzv+5ww+0XwST6SdJDpJ1osepd+mSI9gQ7On1dq6QVDTMQ0IOHBjZrR+5NVUS5qW2BNaSZ/F
SLiNhGMqh6EIUjtuNMbEerXtgQAAAIbqbOt7dCKmHaYdYqItgMvUS2uLTP3eX2sGZGWD7HUnl5tL
nlHZi/nxnCQfciS/SJZDQ9minUXDIh6eURUHI8PUS8csaXdmrQoHC0qa6H6mg5rZQS6iSlgMH3G9fWL
```

提交

执行结果

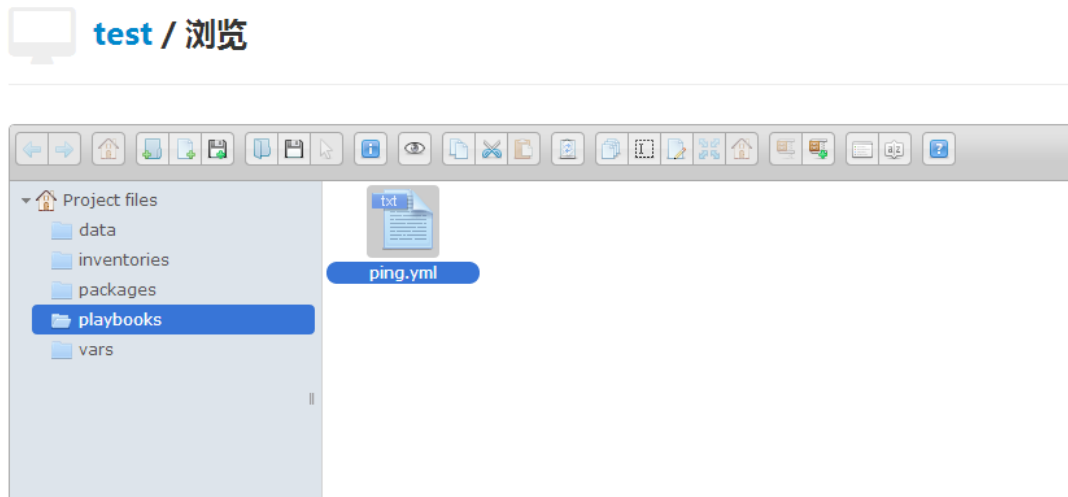
```
copying key to root@ : ~/.ssh/authorized_keys... SUCCESS!
```

3.7 数据

数据的管理包括任务 yaml 文件和用户自定义数据，管理入口包括“数据管理”按钮和“浏览”。



点击后，进入目录管理。数据目录包括 playbooks、data 和 packages，里面的文件可以进行常见的富文本操作，包括查看、编辑、保存、删除、复制等等，可以新建目录和文件，可以上传文件，还可以选择自己喜欢的查看方式等等。总之，它就像你的电脑文件夹一样好用。



在介绍数据目录之前，我们先来简单认识一下 ansible 的 playbooks。Playbooks 可以看做是 ansible 的核心内容，一个 playbooks 文件（yaml 文件）构成了一项任务。一个简单的 playbooks 文件如下：

```
---
- hosts: webservers
  vars:
    http_port: 80
    max_clients: 200
    remote_user: root
  tasks:
    - name: ensure apache is at the latest version
      yum: pkg=httpd state=latest
    - name: write the apache config file
      template: src=/srv/httpd.j2 dest=/etc/httpd.conf
      notify:
        - restart apache
    - name: ensure apache is running
      service: name=httpd state=started
  handlers:
    - name: restart apache
      service: name=httpd state=restarted
```

这个文件实现了配置 apache 并重启的功能。它包括主机、变量、用户和任务。其中，hosts 是由 inventory 中的主机文件来定义（3.5 节），vars 由后面将介绍到的变量文件来定义（3.8 节），而 tasks 任务模块则是 playbooks 的核心部分。

数据管理即围绕 tasks 模块进行展开。

- playbooks 目录：存放 yaml 任务配置文件。任务通过对 ansible 的 modules 进行组合

配置，来实现自己的目的。一个简单的配置如下：

```
- name: ensure apache is at the latest version
  yum: pkg=httpd state=latest
```

通常有 `name` 行和 `module` 行，`name` 行功能类似于注释，并在任务执行时提示进行到了哪一步，`module` 行是真正执行任务的地方。

Module 是 ansible 实现的，也可以自己编写扩展。目前，官方模块有近 200 个，能够满足基本的配置需求，而 github 上第三方的模块也有很多。要想编写出更为完美的任务配置，需要深入了解 ansible 的 playbooks 和 modules。

参考链接：playbooks: <http://docs.ansible.com/playbooks.html>

modules: http://docs.ansible.com/modules_by_category.html

- `data` 目录：存放用户自定义的数据和文件。除了任务文件，用户还有一些配置、数据文件，可以保存在这个目录下。
- `packages` 目录：存放从 jenkins 平台传递过来的程序包。通过持续集成插件（3.15 节），可以实现从 jenkins 平台将程序包自动存储到 Ansible UI 平台。在执行任务时，通过选择程序包，进行程序的部署（3.15 节）。

3.8 变量

定义全局变量，通过 `{{ varname }}` 的方式，在 playbooks 中进行引用。变量文件同样在“浏览”中以目录方式进行管理。变量文件示例如下：

```
# user config
user : ansible
group : ansible
userhome : /works/ansible
workspace : "{{userhome}}/servers/"

# password
ansiblepassword: $1$SomeSalt$Yt7ba.Teae6Jp0Hmzb//
rootpassword: $1$SomeSalt$7J6NXgOrUNCwKzBJ7sv/
```

有一点需要注意，变量 `{{ package }}` 仅作为程序包变量（见 3.9 节“基本配置”项目）使用，自定义数据请勿使用。

3.9 任务

准备好了主机文件、任务配置文件（yml）和变量文件，就可以开始进行任务啦！



点击“部署任务”，进入任务部署页面。部署配置包含 4 个部分：基本配置，高级设置，邮件通知和预约。

- 基本配置

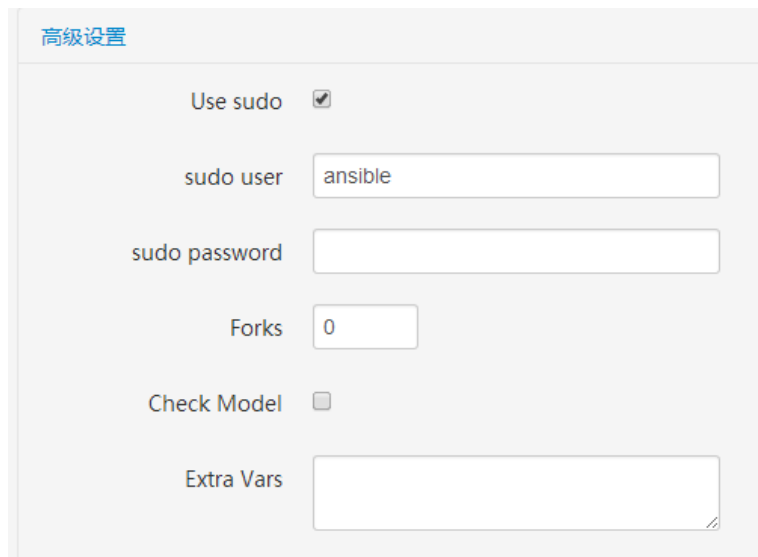
资产和主机分别为 3.5 节中配置的主机文件和组，主机也可以直接添加主机名或者 IP，多个主机之间用“；”分隔；

用户为客户机系统账号；

变量文件通过拖拽的方式拉入；

程序包是通过 jenkins 平台生成的，对应变量{{package}}，通过选择框进行选择；
任务文件通过拖拽的方式拉入和组合。

- 高级设置



高级设置

Use sudo ☒

sudo user

sudo password

Forks

Check Model ☐

Extra Vars

勾选“use sudo”，可以执行 sudo 操作。由 root sudo 到普通账户，无需填写 password；

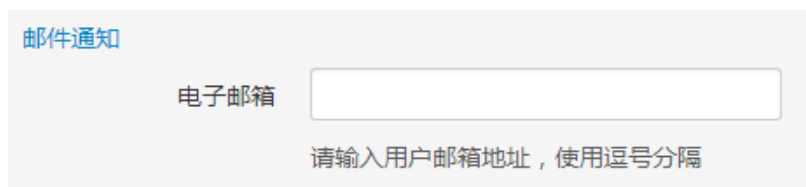
由普通用户 sudo 到 root，需要填写 password；

Forks 配置任务执行时的并发数，通常一个 host 起一个进程，当主机数过多时，
为避免平台资源使用紧张，建议配置该项。默认配置为 100；

Check Model 勾选时，不会实际执行任务，可以看到主机登录是否正常；

Extra Vars 暂未使用。

- 邮件通知



邮件通知

电子邮箱

请输入用户邮箱地址，使用逗号分隔

任务执行结果可以发送邮件，通知用户，多个用户使用逗号分隔邮箱地址。

- 预约



预约

时钟

设置预约执行任务的时钟

Calendar view showing August 2014 with the 6th highlighted.

任务可以预约执行，设定执行的日期和时间，提前安排部署事宜，结合邮件通知，

可以轻松实现工作生活两不误。

点击“执行”，进入任务执行页面。



重启

状态显示了当前任务的状态，包括 new、pending、running、successful、failed、error、canceled 共 7 种状态。

- New: 任务已经建立，但是还未开始
- Pending: 任务已经进入队列，但还未开始执行
- Running: 任务正在执行中
- Successful: 任务执行成功
- Failed: 任务执行失败
- Error: 任务无法执行，通常指平台本身出现异常
- Canceled: 任务被取消

点击右上角的“重启”按钮，可以直接重启当前任务。

stdout

```
PLAY [test] *****

GATHERING FACTS *****
ok: [localhost]
ok: [10.163.177.163]

TASK: [ping hosts] *****
ok: [10.163.177.163]
ok: [localhost]

PLAY RECAP *****
10.163.177.163      : ok=2    changed=0    unreachable=0    failed=0
localhost          : ok=2    changed=0    unreachable=0    failed=0
```

Stdout 显示了任务的标准输出，可以看到任务执行的每一步及结果，可以看到每个主机执行任务的成功、失败、变更和无法登录的数量统计，在 debug 任务时极为有用。

当任务执行时发生异常，则会触发 traceback，便于查找原因。

traceback

```

Traceback (most recent call last):
  File ".../apps/ansible/tasks.py", line 252, in run
    status, stdout, stderr = self.run_pexpect(job_pk, args, cwd, env, passwords)
  File ".../s/ansible/tasks.py", line 227, in run_pexpect
    child.sendline(AESdecrypt("passwd56",profile.ssh_password))
  File ".../pps/account/crypt.py", line 41, in AESdecrypt
    decodedCiphertext = ciphertext.decode("hex")
  File ".../python2.6/encodings/hex_codec.py", line 42, in hex_decode
    output = binascii.a2b_hex(input)
TypeError: Odd-length string

```

在页面的左下角，还能看到当前任务所配置的 inventory 和 playbook，点击查看。

Ansible Resources[Inventory Playbook](#)

Playbook 即是一个完整的 ansible 的 playbook 任务，后台就是执行的这个任务文件。

e58265d8-1ed2-11e4-aa54-00163e003d98.yml

```

- hosts : test
  user  : ansible
  tasks :
    - include : /works/ansible_ui/projects/test/playbooks/ping.yml

```

3.10 任务模板

常用的任务可以保存为任务模板，以方便使用。

在配置任务的页面，点击保存，弹出“模板名称”控件。

The screenshot shows a task configuration interface. At the top, there is a '预约' (Reservation) button. Below it, there are two buttons: '执行' (Execute) and '保存' (Save). The '保存' button is highlighted with a red rectangular box. Below the buttons, there is a label '模板名称' (Template Name) followed by an input field containing the placeholder text 'Template name'.

填写模板名称时，请以项目名称开头，示例：project-product/test-jobname

所有的任务模板会在项目主页面显示。

<div>任务模板</div> <div>任务日志</div>			
模板名称	创建者	创建时间	操作
test-test-ping	admin	八月 11, 2014, 1:43 p.m.	删除

点击任务模板，可以方便的再次执行该任务。

3.11 预约管理

如果设置了任务预约执行，则可以对预约的任务进行查看和删除操作。操作入口为“预约管理”。



点击进入后，可以看到所有预约的任务，并可以查看任务和执行删除操作，目前不能进行修改操作。

test / 预约				
任务名称	创建时间	预约时间	创建者	操作
job-20140724185731-00003	七月 24, 2014, 6:57 p.m.	十月 1, 2014, 6:57 p.m.	admin	删除

3.12 日志

所有执行的任务，都有完善的日志信息记录，可以查看谁在何时执行了哪些任务。入口为项目主页面的“任务日志”区域。

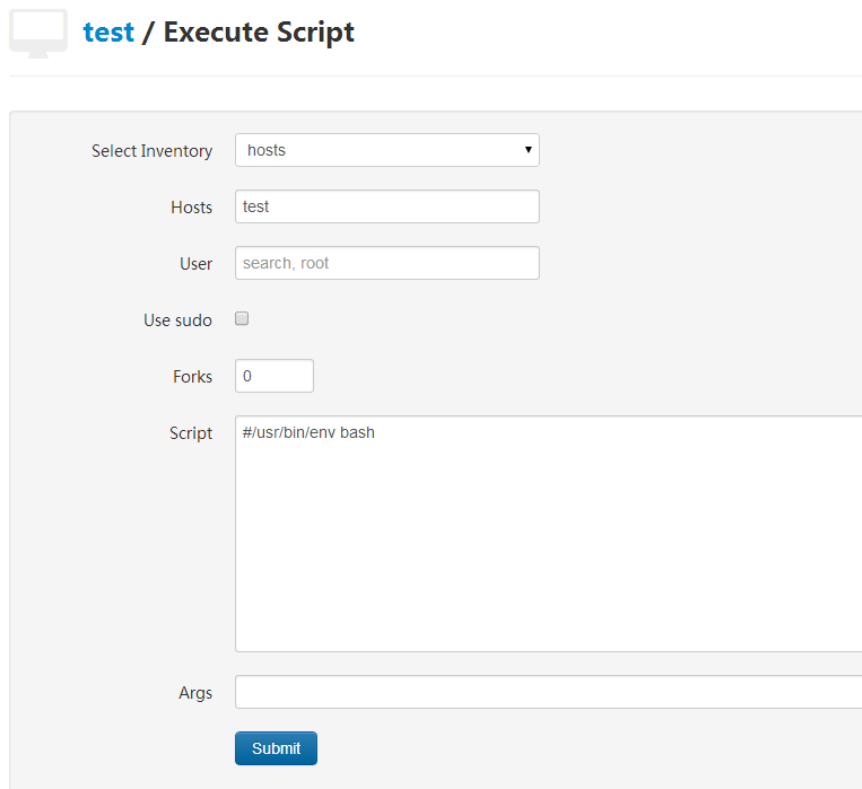
<div>任务模板</div> <div>任务日志</div>				
任务ID	任务状态	创建者	创建时间	操作
job-20140808160627-00004	successful	admin	八月 8, 2014, 4:06 p.m.	查看
job-20140808113953-00003	successful	admin	八月 8, 2014, 11:39 a.m.	查看
job-20140808113850-00002	successful	admin	八月 8, 2014, 11:38 a.m.	查看

3.13 执行脚本

考虑到兼容用户已有的 shell 脚本，系统还提供了直接执行脚本的功能。入口为项目主页面的“脚本执行”按钮。



脚本执行配置页面同部署任务配置相似，不同的地方在于需要将脚本内容粘贴到“Script”控件中，而不是“拖拽”任务（yaml）文件。



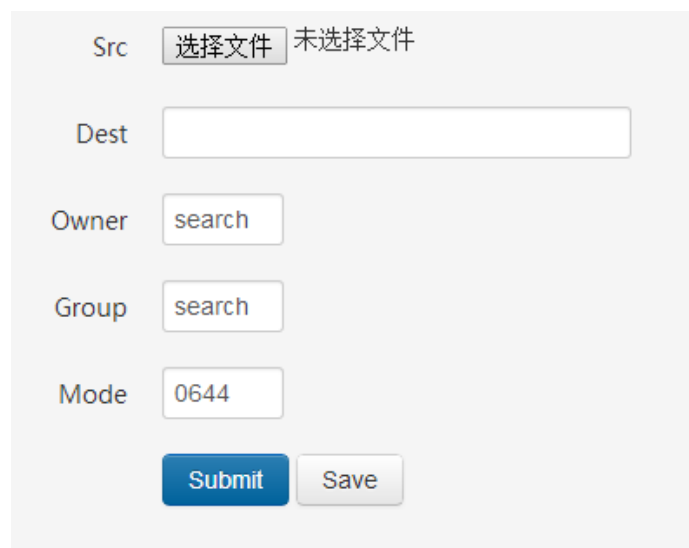
另外，脚本执行功能不支持预约执行和任务模板功能。

3.14 传输文件

系统还提供了将本地文件传输到客户机的功能。入口为项目主页面的“文件传输”按钮。



文件传输配置页面同部署任务配置多有类似，不同之处在于需要配置文件来源和目的目录，同时可以配置文件属主和模式。



需要注意的是，文件来源不是服务器上，而是用户当前操作的电脑。另外，这种方式并不适合于大数据传输。

文件传输功能同样不支持预约执行和任务模板功能。

3.15 持续集成

Ansible UI 平台提供了用于 jenkins 平台的插件。通过该插件，在 jenkins 平台上编译成功的程序包，会自动打包成 tar.gz 压缩包，传递到 Ansible UI 平台。

在任务配置页面，可以通过选择程序包，并在 yml 任务文件中使用 `{{ package }}` 变量引用，实现程序的自动部署。程序历史版本将被保存，可以方便的实现版本回滚操作。



Version	Date
317	2014-01-02-10:36:50
397	2014-01-08-18:10:16
595	2014-01-17-10:43:41
966	2014-02-14-05:56:03
1963	2014-03-12-05:55:54
2236	2014-03-16-05:55:52
4807	2014-04-22-05:55:57
5064	2014-04-24-05:55:57

使用该插件方式如下：

在 jenkins 平台上的项目配置中，增加 Post Build Task (C++) 或 Post Steps (maven)。根据程序语言的不同，增加如下内容。

```
# for c++, using blade
$JENKINS_HOME/scripts/store4blade.py $JENKINS_HOME $JOB_NAME $BUILD_NUMBER
# for java, using maven
python $JENKINS_HOME/scripts/store4maven.py $JENKINS_HOME $JOB_NAME $SVN_URL
$SVN_REVISION
```

在 Ansible UI 平台上项目配置中，配置“SCM URL”条目。

新的项目

项目名称

描述

SCM类型

SCM URL

组

注意，jenkins 平台上项目的 svn 地址需要与 Ansible UI 平台上项目的“SCM URL”一致。持续集成插件通过匹配“scm url”和“svn url”的地址来配对 jenkins 项目和 Ansible UI 项目。这意味着，“scm url”和 jenkins 平台上项目的 svn 地址可以是伪造的，不一定需要真实存在，只是一个识别 ID 而已。在使用中，只需要变化一下 jenkins 平台中项目调用持续集成插件的方式即可。

```
# for c++, using blade
$JENKINS_HOME/scripts/store4blade.py $JENKINS_HOME $JOB_NAME $BUILD_NUMBER
$your_svn_url
# for java, using maven
python $JENKINS_HOME/scripts/store4maven.py $JENKINS_HOME $JOB_NAME
$your_svn_url $SVN_REVISION
```

4. 附录

4.1 使用 LDAP 认证

平台支持 ldap 认证，如启用，需要配置程序源码。

修改 settings.py，在 AUTHENTICATION_BACKENDS 配置中，增加 ldap 认证功能。

```
'desktop.core.auth.backend.LdapBackend',
```






















修改 settings_local.py，在 LDAP settings 配置中，填写自己的 ldap 信息。

```
# LDAP settings
NT4_DOMAIN = ""
LDAP_URL = "ldap://ldapserver:port"
BIND_USER = "CN=adreader,OU=xxx,OU=xxx,DC=xxx,DC=xxxx"
BIND_PASSWORD = "*****"
SEARCH_DN = "ou=xxxx,dc=xxxx,dc=xxxx"
```

4.2 后台管理

Ansible UI 平台使用了 django 框架的 admin 模块，因此拥有强大的后台管理功能。访问 <http://your ansible ui url/admin>，使用管理员账户登录。

站点管理

Account	
Profiles	 添加  修改
Ansible	
Jobs	 添加  修改
Packages	 添加  修改
Projects	 添加  修改
Auth	
用户	 添加  修改
组	 添加  修改
Djcelery	
Crontabs	 添加  修改
Intervals	 添加  修改
Periodic tasks	 添加  修改
Tasks	 修改
Workers	 添加  修改

可以看到，对于用户账户、任务、项目、程序包以及 celery 任务等，都可以进行管理。具体不再详述。

注意，请不要使用后台修改用户密钥，会造成格式错误。