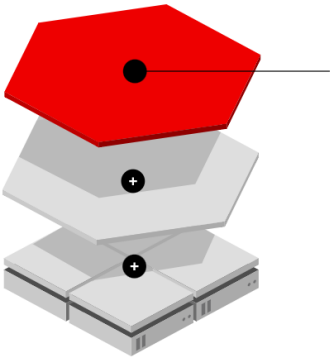

Red Hat
Ansible Automation
Platform

Ansible Automation Platform builds on the foundation provided by Red Hat Enterprise Linux and Red Hat OpenShift and integrates with the rest of our portfolio so you can build and operate automation at scale.

Red Hat
Enterprise Linux

RHCE8辅导1-7

环境描述

考试使用一台管理节点以及 5 台受管主机，考试要求所有操作在 catherine 用户下完成，SSH Key 已经默认配置。

在考试期间，除了你就做位置的台式机之外，在台式机系统中还有多个虚拟机系统。你不具有台式机系统的 root 访问权限，但具有对虚拟机系统完整 root 访问权。

- 练习环境使用 **student** 或者 devops 用户替代 catherine（grep） 用户
  - 模拟环境**设置受管理主机 student 用户 sudo 免密码**
  - devops 需要在 bastion 主机设置 sudo 权限
- 考试环境虚拟机描述：
  - workstation.lab.example.com ansible 管理服务器
  - node1.lab.example.com 节点 1
  - node2.lab.example.com 节点 2
  - node3.lab.example.com 节点 3
  - node4.lab.example.com 节点 4
  - node5.lab.example.com 节点 5

系统	IP地址	Ansible 角色
control	172.25.250.254	ansible control node 管理节点
node1	172.25.250.9	ansible managed node 受控节点
node2	172.25.250.10	ansible managed node 受控节点
node3	172.25.250.11	ansible managed node 受控节点
node4	172.25.250.12	ansible managed node 受控节点
node5	172.25.250.13	ansible managed node 受控节点

这些系统的 IP 地址采用静态设置。请勿更改这些设置。

主机名称解析已配置为解析上方列出的完全限定主机名，同是也解析主机端名称。

账户信息

所有系统的 root 密码是 redhat

请勿更改 root 密码。除非另有指定，否则这将是用于访问其他系统和服务的密码。此外，除非另有指定，否则此密码也应用于你创建的所有用户，或者任何需要设置密码的服务。

为方便起见，所有系统上已预装了 SSH 密钥，允许在不输入密码的前提下通过 SSH 进行 root 访问。请勿对系统上的 root SSH 配置文件进行任何更改。

Ansible 控制节点上已创建了账户 greg。此账户预装了 SSH 密钥，允许在 Ansible 控制节点和各个 Ansible 受管节点之间进行 SSH 登录。请勿对系统上的 greg SSH 配置文件进行任何修改。你可以从 root 账户使用 su 访问此用户账户。

```
重要信息
除非另有指定，否则你所有工作（包括 Ansible playbook、配置文件和主机清单等）应当保存在控制节点上的目录 /home/greg/ansible 中，并且应当归 greg 用户所有。所有 Ansible 相关的命令应当由 greg 用户从 Ansible 控制节点上的这个目录运行。
```

其他信息

一些考试项目可能需要修改 Ansible 主机清单。你要负责确保所有以前的清单组和项目保留下来，与任何其他更改共存。你还要有确保清单中所有默认的组和主机保留你进行的任

何更改。

考试系统上的**防火墙默认为不启用，SELinux 则处于强制模式**。

- 提示：实验环境只有 workstation、bastion、servera、serverb、serverc、serverd，可以使用环境中的 **bastion主机来模拟 node5(servere)** 主机。
- Ansible 控制节点上已创建了用户帐户 student。此帐户预装了 SSH 密钥，允许在 Ansible 控制节点和各个 Ansible 受管节点之间进行 SSH 登录。请勿对系统上的 student SSH 配置文件进行任何修改。您可以从 root 帐户使用 su 访问此用户帐户。
- 在开始实验前，恢复所有虚拟机快照：
  [root@foundation0 ~]# rht-vmctl fullreset all

重要信息

除非另有指定，否则您的所有工作（包括 **Ansible playbook、配置文件和主机清单**等）应

当保存在控制节点上的目录 **/home/student/ansible** 中，并且应当归 student 用户所有。

所有 Ansible 相关的命令应当由 student 用户从 Ansible 控制节点上的这个目录运行。**其他**

信息

一些考试项目可能需要修改 Ansible 主机清单。您要负责确保所有以前的清单组和项目保留下来，与任何其他更改共存。您还要有确保清单中所有默认的组和主机保留您进行的任何更改。

考试系统上的防火墙默认为不启用，SELinux 则处于强制模式。根据题目要求确认是否需

要打开配置防火墙规则。

如果需要安装其他软件，您的系统和 Ansible 控制节点可能已设置为指向 content 上的下述存储库：

http://content/rhel8.0/x86_64/dvd/BaseOS http://content/rhel8.0/x86_64/dvd/AppStream

一些项目需要额外的文件，这些文件已在以下位置提供：

http://materials

产品文档可从以下位置找到：

http://materials/docs/ansible/html

其他资源也进行了配置，供您在考试期间使用。关于这些资源的具体信息将在需要这些资源的项目中提供。

重要信息

请注意，在评分之前，您的 Ansible 受管节点系统将重置为考试开始时的初始状态，您编写的Ansible playbook 将通过以 student 用户身份从控制节点上的目录 /home/student/ansible 目录运行来应用。在playbook 运行后，系统会对您的受管节点进行评估，以判断它们是否按照规定进行了配置。

考试题目

在您的系统上执行以下所有步骤。

安装和配置 Ansible

创建和运行 Ansible 临时命令安装软件包

使用 RHEL 系统角色

使用 Ansible Galaxy 安装角色创建和使用角色

从 Ansible Galaxy 使用角色创建和使用逻辑卷

生成主机文件

修改文件内容

创建 Web 内容目录生成硬件报告

创建密码库 创建用户帐户

更新 Ansible 库的密钥

创建cron定时计划任务

题目描述

vim 设置

```
[student@workstation ~]$ cat ~/.vimrc
set ai
set ts=2
```

一、安装并配置Ansible

```
题目描述：
按照下方所述，在控制节点 control.example.com 上安装和配置 Ansible：
安装所需的软件包
创建名为 /home/greg/ansible/inventory 的静态清单文件，以满足以下要求：
node1 是 dev 主机组的成员
node2 是 test 主机组的成员
node3 和 node4 是 prod 主机组的成员
node5 是 balancers 主机组的成员
prod 组是 webservers 主机组的成员
创建名为 /home/greg/ansible/ansible.cfg 的配置文件，以满足以下要求：
主机清单文件为 /home/greg/ansible/inventory
playbook 中使用的角色的位置包括 /home/greg/ansible/roles
```

- 在 workstation 上安装并配置 ansible, 要求入下：
- 安装相应的软件包
    - 创建一个静态 inventory 到 /home/student/ansible/inventory, 清单包含：
        - servera 属于 dev 组
        - serverb 属于 test 组
        - serverc 和 serverd 属于 prod 组

    - servere 属于 balancers 主机组
    - prod 组属于 webservers 主机组
- 创建名为 /home/student/ansible/ansible.cfg 的配置文件
- 主机清单文件为 /home/student/ansible/inventory
- playbook 中使用的角色目录为 /home/student/ansible/roles

做题流程

在 workstation 主机执行初始化脚本

```
[student@workstation ~]$ lab intro-install start
```

安装 ansible 模块，创建脚本目录

```
[student@workstation ~]$ sudo yum install ansible -y

[student@workstation ~]$ mkdir -p ansible/roles
```

- 编写清单文件

```
[student@workstation ~]$ cd ansible

[student@workstation ansible]$ vim inventory
[dev]
servera

[test]
serverb

[prod]
serverc
serverd

[balancers]
servere  # 模拟环境用 bastion 代替

[webservers:children]
prod
```

- 编写 ansible.cfg 权限配置文件参数

```
[student@workstation ansible]$ pwd
/home/student/ansible

[student@workstation ansible]$ vim ansible.cfg
[defaults]
inventory = /home/student/ansible/inventory
ask_pass = false
host_key_checking = false
roles_path = /home/student/ansible/roles
#roles_path=/home/student/ansible/roles:/etc/ansible/roles:/usr/share/ansible/roles
remote_user = student

[privilege_escalation]
become=True
become_method=sudo
become_user=root
become_ask_pass=False
```

- 验证清单主机

```
[student@workstation ansible]$ ansible --version
[student@workstation ansible]$ ansible dev --list-hosts
[student@workstation ansible]$ ansible test --list-hosts
[student@workstation ansible]$ ansible prod --list-hosts
[student@workstation ansible]$ ansible balancers --list-hosts
[student@workstation ansible]$ ansible prod --list-hosts
[student@workstation ansible]$ ansible all --list-hosts
[student@workstation ansible]$ ls
ansible.cfg  inventory  roles
```

二、创建并运行 ansible 临时命令

作为系统管理员，你需要在受管节点上安装软件。

● 创建名为 **/home/student/ansible/adhoc.sh** 的 shell **脚本**，该脚本将使用 Ansible 临时命令在各个受管节点上安装yum 存储库：

仓库 1：

- 仓库名： EX294_BASE
- 描述为： EX294 base software
- Base URL： http://content.example.com/rhel8.0/x86_64/dvd/BaseOS

- 需要开启 GPG 签名验证
- GPG 密钥的地址是： http://content.example.com/rhel8.0/x86_64/dvd/RPM-GPGKEY-redhat-release
- 仓库的状态是启用状态

仓库 2：

- 仓库名：EX294_STREAM
- 描述为：EX294 stream software
- Base URL： http://content.example.com/rhel8.0/x86_64/dvd/AppStream
- 需要开启 GPG 签名验证
- GPG 密钥的地址是： http://content.example.com/rhel8.0/x86_64/dvd/RPM-GPGKEY-redhat-release
- 仓库的状态是启用状态

做题流程

- 编写 adhoc.sh 脚本文件

```
[student@workstation ~]$ vim /home/student/ansible/adhoc.sh
#!/bin/bash
ansible all -m yum_repository -a 'name="EX294_BASE" description="EX294 base software"  bas
eurl="http://content.example.com/rhel8.0/x86_64/dvd/BaseOS" gpgcheck=yes  gpgkey="http://c
ontent.example.com/rhel8.0/x86_64/dvd/RPM-GPG-KEY-redhat-release"  file=rh294_base'

ansible all -m yum_repository -a 'name="EX294_STREAM" description="EX294 stream software"
 baseurl="http://content.example.com/rhel8.0/x86_64/dvd/AppStream" gpgcheck=yes gpgkey="ht
tp://content.example.com/rhel8.0/x86_64/dvd/RPM-GPG-KEY-redhat-release" file=rh294_stream'

[student@workstation ansible]$ chmod a+x adhoc.sh
[student@workstation ansible]$ ./adhoc.sh
```

- 验证

```
[student@workstation ansible]$ ansible servera -m shell -a 'yum repolist'
```

三、安装软件包

```
题目描述：
创建一个名为 /home/greg/ansible/packages.yml 的 playbook ：
将 php 和 mariadb 软件包安装到 dev、test 和 prod 主机组中的主机上
将 RPM Development Tools 软件包组安装到 dev 主机组中的主机上
将 dev 主机组中主机上的所有软件包更新为最新版本

创建一个名为/home/student/ansible/cron.yml 的 playbook:
- 将在 dev 主机上运行一个周期性计划任务
- 此任务将在每天下午的 2 点执行: /etc/cron.daily/logrotate

按照下方所述，创建一个定时计划任务：
创建一个playbook文件名为create_crontab.yml.要求prod组内成员，执行调度周期性Cron作业，要求每天的2点、
5点执行以下命令 "ls –alh > /dev/null"
```

- 创建 **/home/student/ansible/install_packages.yml** 的 playbook：
  - 将 php 和 mariadb 软件包安装到 dev、test 和 prod 主机组中的主机上
  - 将 RPM Development Tools 软件包组安装到 dev 主机组中的主机上
  - 将 dev 主机组中主机上的所有软件包更新为最新版本
- 创建一个名为/home/student/ansible/cron.yml 的 playbook:
  - 将在 dev 主机上运行一个周期性计划任务
  - 此任务将在每天下午的 2 点执行: /etc/cron.daily/logrotate

做题流程

inventory_hostname 和 ansible_hostname 都是 ansible的特殊变量

https://docs.ansible.com/ansible/latest/reference_appendices/special_variables.html

```
# 带 block
[student@workstation ansible]$ vim install_packages.yml
---
- name: install software
  hosts: dev,test,prod
  tasks:
    - name: install the latest version
```

```
      yum:
        name: "{{ item }}"
        state: latest
      loop:
        - php
        - mariadb
    - block:
      - name: install the '@RPM Development Tools' package group
        yum:
          name: "@RPM Development Tools"
          state: present
        when: "'dev' in group_names"
      - name: upgrade all packages
        yum:
          name: '*'
          state: latest
        when: "'dev' in group_names"
```

```
不带 block
[student@workstation ansible]$ vim install_packages.yml
---
- name: install software
  hosts: dev,test,prod
  tasks:
    - name: install the latest version
      yum:
        name: "{{ item }}"
        state: latest
      loop:
        - php
        - mariadb
    - name: install the '@RPM Development Tools' package group
      yum:
        name: "@RPM Development Tools"
        state: present
      when: "'dev' in group_names"
    - name: upgrade all packages
      yum:
        name: '*'
        state: latest
      when: "'dev' in group_names"
```

方法一：

```
[student@workstation ansible]$ vim install_packages.yml
---
- name: install php and mariadb package
  hosts:
    - dev
    - test
    - prod
  tasks:
    - name: install packages
      yum:
        name:
          - php
          - mariadb
        state: present
    - name: install development tools group
      yum:
        name: "@Development tools"
        state: present
      when: ansible_hostname in groups['dev']
    - name: update all packages
      yum:
        name: '*'
        state: latest
      when: ansible_hostname in groups['dev']

[student@workstation ansible]$ ansible-playbook install_packages.yml
```

方法二：

```
- name: install package
  hosts: dev,test,prod
  tasks:
    - name: install the latest version
      yum:
        name: "{{ item }}"
        state: present
      loop:
        - php
        - mariadb

- name: install group package and update
  hosts: dev
  tasks:
    - name: install RPM
      yum:
        name: "@RPM Development Tools"
        state: present
```

```
    - name: update packages
      yum:
       name: '*'
       state: latest
```

编写计划任务

```
[student@workstation ansible]$ vim cron.yml
---
- name: cron-test
  hosts: prod
  tasks:
    - name: cron
      cron:
        name: add cron
        minute: 0
        hour: 2,5
        job: ls –alh >/dev/null
        state: present
```

验证

```
[greg@control ansible]$ ansible prod -m shell -a 'crontab -l -u root'
node4 | CHANGED | rc=0 >>
#Ansible: add cron
0 2,5 * * * ls –alh >/dev/null

node3 | CHANGED | rc=0 >>
#Ansible: add cron
0 2,5 * * * ls –alh >/dev/null
```

四、使用 RHEL system role

```
题目描述：
安装 RHEL 系统角色软件包，并创建符合以下条件的 playbook /home/greg/ansible/timesync.yml ：
在所有受管节点上运行
```

```
使用 timesync 角色
配置该角色，以使用当前有效的 NTP 提供商
配置该角色，以使用时间服务器 172.20.20.254
配置该角色，以启用 iburst 参数
```

- 安装 RHEL system roles 软件包，并创建符合以下条件的 playbook /home/student/ansible/configure_timesync.yml：
  - 在所有受管节点上运行
  - 使用 timesync 角色
  - 配置该角色，以使用当前有效的 NTP 提供商
  - 配置该角色，以使用时间服务器 172.20.20.254（classroom.example.com）
  - 配置该角色，以启用 iburst 参数

做题流程

安装 rhel-system-roles

```
[student@workstation ansible]$ sudo yum install -y rhel-system-roles

[student@workstation ~]$ pwd
/home/student

[student@workstation ~]$ cd ~/ansible/roles/
[student@workstation roles]$ pwd
/home/student/ansible/roles

[student@workstation roles]$ rpm -ql rhel-system-roles | grep sync
/usr/share/ansible/roles/linux-system-roles.timesync
/usr/share/ansible/roles/rhel-system-roles.timesync

[student@workstation ansible]$ cp -av /usr/share/ansible/roles/rhel-system-roles.timesync
 /home/student/ansible/roles


[student@workstation roles]$ cd
[student@workstation ~]$ ls
ansible  deploy-adhoc  playbook-review  role-system
[student@workstation ~]$ cd ansible/
[student@workstation ansible]$ ls
adhoc.sh  ansible.cfg  install_package.yml  inventory  roles
```

查询角色信息

```
[student@workstation ansible]$ ansible-galaxy list
# /home/student/ansible/roles
- rhel-system-roles.timesync, (unknown version)
```

编写时间同步脚本

```
# 参考样例文件
[student@workstation ansible]$ cat roles/rhel-system-roles.timesync/README.md

[student@workstation ansible]$ cat configure_timesync.yml
---
- name: time sync
  hosts: all
  tasks:
  vars:
    timesync_ntp_servers:
      - hostname: classroom.example.com
        iburst: yes
    timesync_ntp_provider: chrony
  roles:
    - rhel-system-roles.timesync
```

验证

```
[student@servera ~]$ timedatectl
               Local time: Sat 2022-05-07 18:14:58 CST
           Universal time: Sat 2022-05-07 10:14:58 UTC
                 RTC time: Sat 2022-05-07 10:14:57
                Time zone: Asia/Shanghai (CST, +0800)
System clock synchronized: yes
              NTP service: active
          RTC in local TZ: no
[student@servera ~]$ cat /etc/chrony.conf
# Ansible managed

server classroom.example.com iburst

# Allow the system clock to be stepped in the first three updates.
```

```
makestep 1.0 3

# Enable kernel synchronization of the real-time clock (RTC).
rtcsync

# Record the rate at which the system clock gains/losses time.
driftfile /var/lib/chrony/drift
```

五、使用 Ansible Galaxy 安装 roles

- 使用 Ansible Galaxy 和要求文件 /home/student/ansible/roles/requirements.yml 。下载并安装 Ansible Galaxy 角色到 /home/devops/ansible/roles 目录中，可以从下列 URL 下载：
  - 下载 http://materials.example.com/haproxy.tar 这个角色的名字叫 balancer
  - 下载 http://materials.example.com/phpinfo.tar 这个角色的名字叫 phpinfo

  - 模拟环境下载地址 http://172.25.254.102/download/haproxy.tar.gz
  - 模拟环境下载地址 http://172.25.254.102/download/phpinfo.tar.gz

做题流程

```
[student@workstation ansible]$ vim roles/requirements.yml
---
  - src: http://materials.example.com/phpinfo.tar
    name: phpinfo
  - src: http://materials.example.com/haproxy.tar
    name: balancer
```

- 模拟环境 requirement.yml （根据考试要求）

```
# requirements.yml是在/home/student/ansible/roles目录还是/home/student/ansible/目录，根据题目要求创建
```

```
# 模拟环境下载地址
[student@workstation ansible]$ vim roles/requirements.yml
---
  - src: http://content.example.com/download/haproxy.tar.gz
    name: balancer
  - src: http://content.example.com/download/phpinfo.tar.gz
    name: phpinfo
```

安装角色

```
[student@workstation ansible]$ pwd
/home/student/ansible
[student@workstation ansible]$ ls
/home/student/ansible/requirements.yml

# 指定角色安装路径 /home/student/ansible/roles
[student@workstation ansible]$ ansible-galaxy install -r requirements.yml -p roles/
- downloading role from http://content.example.com/download/haproxy.tar.gz
- extracting balancer to /home/student/ansible/roles/balancer
- balancer was installed successfully
- downloading role from http://content.example.com/download/phpinfo.tar.gz
- extracting phpinfo to /home/student/ansible/roles/phpinfo
- phpinfo was installed successfully
```

验证角色信息

```
[student@workstation ansible]$ ansible-galaxy list
# /home/student/ansible/roles
- rhel-system-roles.timesync, (unknown version)
- balancer, (unknown version)
- phpinfo, (unknown version)
```

六、创建并使用角色（role）

```
根据下列要求，在 /home/greg/ansible/roles 中创建名为 apache 的角色：
httpd 软件包已安装，设置为系统启用并启动
防火墙已启用并正在运行，并使用允许访问 web 服务器的规则
模板文件 index.html.j2 已存在，用于创建具有以下输出的文件 /var/www/html/index.html：
```

```
 Welcome to HOSTNAME on IPADDRESS
其中，HOSTNAME 是受管节点的完全限定域名，IPADDRESS 则是受管节点的 IP 地址。
```

- 在 /home/devops/ansible/roles 目录中创建名为 apache 的角色，要求：
  - 安装 httpd 软件包，开机启动，并运行
  - 启用防火墙，允许 web 服务
  - 模板文件 index.html.j2，复制到 /var/www/html/index.html，内容如下：Welcome to HOSTNAME on IPADDRESS
  - 其中，HOSTNAME 是受管节点的完全限定域名，IPADDRESS 则是受管节点的 IP 地址。
- 创建 playbook /home/student/ansible/newroles.yml，在 webservers 主机组使用 apahce 的角色。
- （创建 playbook /home/student/ansible/apache.yml，在 webservers 主机组使用 apahce 的角色。）

做题流程

```
[student@workstation ansible]$ cd roles/
[student@workstation roles]$ ansible-galaxy init apache
```

修改角色的 main.yml（练习环境，受控节点防火墙没有开启，需要添加启动防火墙任务）

```
---
- name: ensure httpd is install
  yum:
    name: httpd
    state: latest
- name: ensure httpd is stared and enabled
  service:
    name: httpd
    state: started
    enabled: yes
- name: ensure httpd is stared and enabled
  service:
    name: firewalld
```

```
    state: started
    enabled: yes
- name: open firewall port
  firewalld:
    service: http
    permanent: yes
    state: enabled
    immediate: yes
- name: index html file is installed
  template:
    src: index.html.j2
    dest: /var/www/html/index.html
    owner: root
    group: root
    mode: 0644
```

编写 jinja2 模板文件

```
[student@workstation roles]$ vim  apache/templates/index.html.j2
Welcome to {{ ansible_facts['fqdn'] }} on {{ ansible_facts['default_ipv4']['address'] }}
```

编写 playbook 脚本

```
[student@workstation ansible]$ cd ..
[student@workstation ansible]$ vim newroles.yml
---
- name: use apache role playbook
  hosts: webservers
  tasks:
  roles:
    - apache

[student@workstation ansible]$ ansible-playbook newroles.yml
```

验证

```
[student@workstation ansible]$ curl http://serverc.lab.example.com
welcome to serverc.lab.example.com on 172.25.250.12
```

```
[student@workstation ansible]$ curl http://serverd.lab.example.com
welcome to serverd.lab.example.com on 172.25.250.13
```

七、使用来自 Ansible Galaxy 的 roles

```
题目描述：
根据下列要求，创建一个名为 /home/greg/ansible/roles.yml 的 playbook ：
playbook 中包含一个 play， 该 play 在 balancers 主机组中的主机上运行并将使用 balancer 角色

此角色配置一项服务，以在 webservers 主机组中的主机之间平衡 Web 服务器请求的负载。

浏览到 balancers 主机组中的主机（例如 http://node5.example.com/）将生成以下输出：
Welcom to node3.example.com on 172.24.22.8

重新加载浏览器将从另一 Web 服务器生成输出：
Welcom to node4.example.com on 172.24.22.9

playbook 中包含一个 play， 该 play 在 webservers 主机组中的主机上运行并将使用 phpinfo 角色。

请通过 URL /hello.php 浏览到 webservers 主机组中的主机将生成以下输出：
Hello PHP World from FQDN

其中，FQDN 是主机的完全限定名称。

例如，浏览到 http://node3.example.com/hello.php 会生成以下输出：
Hello PHP World from node3.example.com
另外还有 PHP 配置的各种详细信息，如安装的 PHP 版本等。

同样，浏览到 http://node4.example.com/hello.php 会生成以下输出：
Hello PHP World from node4.example.com
另外还有 PHP 配置的各种详细信息，如安装的 PHP 版本等
```

- 创建一个 playbook 名字叫 /home/admin/ansible/roles.yml要求如下：
  - 这个 playbook 运行在 balancers 这个主机组上，并使用 balancer 的角色。
  - 这个角色配置一个用来在 webservers 主机组上实现一个负载均衡服务
  - 使用浏览器访问 balancers 主机组, 如 http://servere.lab.example.com （bastion）可以看到如下输出：
    welcome to serverc.lab.example.com on 172.25.250.12

    - 重新刷新浏览器，可以看到如下输出：
      welcome to serverd.lab.example.com on 172.25.250.13
    - 这个 playbook 运行在 webservers 主机组上使用 phpinfo 的角色。
- 使用浏览器 URL： servere.lab.example.com/hello.php 访问主机组 webservers 时，看到如下输出：Hello PHP World from FQDN
- 注意， FQDN 是主机的完整域名。
- 比如，浏览器访问 URL：http://serverc.lab.example.com/hello.php 看到如下输出：

  Hello PHP World from serverc.lab.example.com
- 类似的，浏览器访问 URL：http://serverd.lab.example.com/hello.php 看到如下输出：

  Hello PHP World from serverd.lab.example.com

- 修改 balancer tasks/main.yml 文件

```
backend habackend
    server serverc.lab.example.com 172.25.250.12:80 check
    server serverd.lab.example.com 172.25.250.13:80 check
```

- 修改 phpinfo template/hello.php.j2

```
<!DOCTYPE html>
<html>
<body>
<h1>
PHP World from {{ ansible_fqdn }}
</h1>
<?php
phpinfo();
?>
</body>
</html>
```

做题流程

```
[student@workstation ansible]$ vim roles.yml
---
- name: use php role
  hosts: webservers
  roles:
    - phpinfo
- name: use haproxy role
  hosts: balancers  # 练习环境，balncers组的bastion用 servera 或者 serverb 替代
  roles:
    - balancer
  tasks:
    - name: start firewalld service
      service:
        name: firewalld
        state: started
        enabled: yes
    - name: open firewalld port
      firewalld:
        service: http
        permanent: yes
        state: enabled
        immediate: yes

[student@workstation ansible]$ ansible-playbook roles.yml

[student@workstation ansible]$ curl http://servere

# 模拟环境用servera 或者 serverb 替代
[student@workstation ansible]$ curl http://servere
Welcome to serverc.lab.example.com to 172.25.250.12
[student@workstation ansible]$ curl http://servere
Welcome to serverd.lab.example.com to 172.25.250.13
[student@workstation ansible]$ curl http://bastion
Welcome to serverc.lab.example.com to 172.25.250.12
[student@workstation ansible]$ curl http://bastion
Welcome to serverd.lab.example.com to 172.25.250.13

[student@workstation ansible]$ curl http://serverc.lab.example.com/hello.php
[student@workstation ansible]$ curl http://serverd.lab.example.com/hello.php
```


serverc.lab.example.com/hello.php
PHP World from serverc.lab.example.com
PHP Version 7.2.11


serverd.lab.example.com/hello.php
PHP World from serverd.lab.example.com
PHP Version 7.2.11