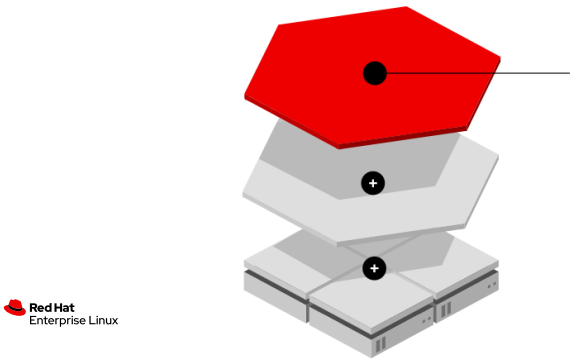

Red Hat
Ansible Automation
Platform

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.

RHCE8辅导8-15

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

node1	172.25.250.9	ansible managed node
node2	172.25.250.10	ansible managed node
node3	172.25.250.11	ansible managed node
node4	172.25.250.12	ansible managed node
node5	172.25.250.13	ansible managed node

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
set sts=2
set et
set sw=2
```

八、创建并使用磁盘分区和逻辑卷

8.1 创建并使用磁盘分区（老题目）

- 创建名为 partition.yml 的 playbook，对所有的节点进行操作：
  - 在 vdb 上创建一个主分区， 1500MiB
  - 使用 ext4 文件系统进行格式化
  - 将文件系统挂载到 /newpart
  - 如果分区大小不满足，产生报错信息： could not create partation of that size 则创建分区大小变成 800MiB
  - 如果磁盘不存在，产生报错信息： disk does not exist
- 考试的时候，只有创建逻辑卷的题目。模拟环境下需要结合创建分区和卷组模拟考试环境。

做题流程

实验环境，bastion 没有 vdb，servera、serverb、serverc、serverd 有 vdb，大小5G

```
[student@workstation ansible]$ vim partition.yml
---
- name: create a partition
  hosts: all
  tasks:
    - name: check vdb info
      shell: "ls -l /dev/vdb"
      register: disk_info
      ignore_errors: yes
    - name: show debug message
      debug:
        msg: "disk does not exist"
      when: disk_info.rc != 0
    - name: use block create partition
      block:
        - name: create 1500MB partition
          parted:
            device: /dev/vdb
            number: 1
            state: present
            part_end: 1500MiB
      rescue:
        - name: show message
          debug:
            msg: "could not cerate partition of that size"
        - name: create 800M partition
          parted:
            device: /dev/vdb
            number: 1
            state: present
            part_end: 800MiB
      always:
        - name: mkfs filesystem
          filesystem:
            fstype: ext4
            dev: /dev/vdb1

        - name: mount filesystem to mount point
          mount:
            path: /newpart
            src: /dev/vdb1
            fstype: ext4
            state: mounted
```

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

PLAY [create a partition] ******************************************************************
```

```
*****************************************************************

TASK [Gathering Facts] ********************************************************************
*****************************************************************
ok: [serverd]
ok: [serverb]
ok: [serverc]
ok: [bastion]
ok: [servera]

TASK [check vdb info] *********************************************************************
*****************************************************************
fatal: [bastion]: FAILED! => {"changed": true, "cmd": "ls -l /dev/vdb", "delta": "0:00:00.
006278", "end": "2022-05-08 00:46:46.023841", "msg": "non-zero return code", "rc": 2, "sta
rt": "2022-05-08 00:46:46.017563", "stderr": "ls: cannot access '/dev/vdb': No such file o
r directory", "stderr_lines": ["ls: cannot access '/dev/vdb': No such file or directory"],
"stdout": "", "stdout_lines": []}
...ignoring
changed: [servera]
changed: [serverb]
changed: [serverd]
changed: [serverc]

TASK [show debug message] *****************************************************************
*****************************************************************
skipping: [servera]
skipping: [serverb]
ok: [bastion] => {
    "msg": "disk does not exist"
}
skipping: [serverc]
skipping: [serverd]

TASK [create 1500MB partition] ************************************************************
*****************************************************************
fatal: [bastion]: FAILED! => {"changed": false, "err": "Error: Could not stat device /dev/
vdb - No such file or directory.\n", "msg": "Error while getting device information with p
arted script: '/sbin/parted -s -m /dev/vdb -- unit 'KiB' print'", "out": "", "rc": 1}
changed: [serverb]
changed: [serverc]
changed: [serverd]
changed: [servera]

TASK [show message] ***********************************************************************
*****************************************************************
ok: [bastion] => {
    "msg": "could not cerate partition of that size"
}

TASK [create 800M partition] **************************************************************
*****************************************************************
fatal: [bastion]: FAILED! => {"changed": false, "err": "Error: Could not stat device /dev/
vdb - No such file or directory.\n", "msg": "Error while getting device information with p
arted script: '/sbin/parted -s -m /dev/vdb -- unit 'KiB' print'", "out": "", "rc": 1}
```

```
TASK [mkfs filesystem] ********************************************************************
*****************************************************************
fatal: [bastion]: FAILED! => {"changed": false, "msg": "Device /dev/vdb1 not found."}
changed: [serverb]
changed: [servera]
changed: [serverc]
changed: [serverd]

TASK [mount filesystem to mount point] ****************************************************
*****************************************************************
changed: [servera]
changed: [serverb]
changed: [serverd]
changed: [serverc]

PLAY RECAP ********************************************************************************
*****************************************************************
bastion                    : ok=4    changed=1    unreachable=0    failed=2    skipped=0
rescued=1    ignored=1
servera                    : ok=5    changed=4    unreachable=0    failed=0    skipped=1
rescued=0    ignored=0
serverb                    : ok=5    changed=4    unreachable=0    failed=0    skipped=1
rescued=0    ignored=0
serverc                    : ok=5    changed=4    unreachable=0    failed=0    skipped=1
rescued=0    ignored=0
serverd                    : ok=5    changed=4    unreachable=0    failed=0    skipped=1
rescued=0    ignored=0
```

仅参考用

```
[student@workstation ansible]$ cat lvnew2021.7.5.yml
---
- name: create lv
  hosts: test
  tasks:
    - name: check vg is exit
      shell: 'vgdisplay vg0'
      register: result
      ignore_errors: yes
    - name: show debug message
      debug:
        msg: "Volume group does not exit"
      when: result.rc != 0
      failed_when: result.rc != 0

    - name: create new partition
      parted:
        device: /dev/vdb
```

```
        number: 1
        state: present
        part_start: 1MiB
        part_end: 801MiB
        flags: [ lvm ]
    - name: create vg
      lvg:
        vg: vg0
        pvs: /dev/vdb2
    - name: Use block create lvm
      block:
        - name: create 1500MB partitions
          lvol:
            vg: vg0
            lv: lv0
            size: 1500
      rescue:
        - name: report size not enough
          debug:
            msg: Could not create logical volume of that size
        - name: create lv use research vg
          lvol:
            vg: vg0
            lv: lv0
            size: 100%VG
        - name: Create a logical volume
          filesystem:
            dev: /dev/vg0/lv0
            fstype: ext4
```

8.2 创建并使用逻辑卷（考试）

```
题目描述：
创建一个名为 /home/greg/ansible/lv.yml 的 playbook ，它将在所有受管节点上运行以执行下列任务：
创建符合以下要求的逻辑卷：
逻辑卷创建在 research 卷组中
逻辑卷名称为 data
逻辑卷大小为 1500 MiB
使用 ext4 文件系统格式化逻辑卷

如果无法创建请求的逻辑卷大小，应显示错误信息：Could not create logical volume of that size
并且应改为使用大小 800 MiB。

如果卷组 research 不存在，应显示错误信息：Volume group done not exist
不要以任何方式挂载逻辑卷
```

此题需要部分节点中存在 research 卷组，但官方的练习环境中未配置，需要在做题之需要创建一个 add.yml 的脚本，内容如下，添加卷组：
（如下脚本只需要在官方练习环境下添加，为完成逻辑卷题目，初始化练习环境）。

```
[student@workstation ansible]$ cat pre_lvm.yml
---
- name: create 2000M partition
  hosts: servera,serverb
  tasks:
    - name: create 2000M partion
      parted:
        device: /dev/vdb
        number: 2
        state: present
        part_start: 1501MiB
        part_end: 3500MiB
    - name: create 2000M vg
      lvg:
        vg: research
        pvs: /dev/vdb2
- name: create 1000M partition
  hosts: serverc,serverd
  tasks:
    - name: create 1000M partion
      parted:
        device: /dev/vdb
        number: 2
        state: present
        part_start: 1501MiB
        part_end: 2500MiB
    - name: create 1000M vg
      lvg:
        vg: research
        pvs: /dev/vdb2
```

- 创建一个 playbook 名字叫 /home/student/ansible/lv.yml，对所有的节点操作：
- 创建一个逻辑卷要求如下：
  - LV 创建在 vg0 这个卷组里
  - LV 的名字叫 lv0
  - LV 的大小是 15000MiB
  - 格式化成 ext4 文件系统

- 如果 LV 的大小不满足，产生报错信息： Could not create lv of that size

  并且将创建分区大小变成 800MiB
- 如果卷组 vg0 不存在，显示错误信息: Volume group does not exist
- 不要使用任何方式挂载逻辑卷

做题流程

- 考试的时候，五台主机中，有3台已经创建好了 vg0 卷组，可以直接创建逻辑卷，有2台没有创建卷组。
- 官方模拟环境下，可以自定义脚本，先创建分区和卷组来模拟。

查询磁盘设备事实变量

```
ansible dev -m setup -a 'filter=*lvm*'
ansible dev -m setup -a 'filter=*device*'
```

模拟环境参考

```
[student@workstation ansible]$ cat lv.yml
---
- name: create and use lvm
  hosts: all
  tasks:
    - name: verify vg research exist
      shell: "vgs | grep research"
      ignore_errors: yes
      register: vg_info
    - name: print vg message
      debug:
        msg: 'research vg does not exist'
      when: vg_info.rc !=0
    - name: verify vg research exist
      debug:
        msg: "vg group does not exist"
      when: '"research" not in ansible_lvm.vgs'
    - name: create data
      block:
        - name: create 1500MB lvm
          lvol:
```

```
            lv: data
            vg: research
            state: present
            size: 1500
          when: '"research" in ansible_lvm.vgs'
      rescue:
        - name: show error message
          debug:
            msg: "could not create lv of that size"
        - name: create 800M lvm
          lvol:
            lv: data
            vg: research
            size: 800
            state: present
          when: '"research" in ansible_lvm.vgs'
      always:
        - name: create filesystem
          filesystem:
            fstype: ext4
            dev: /dev/research/data
            force: yes
          when: '"research" in ansible_lvm.vgs'
```

方法一：nanjing

```
[student@workstation ansible]$ vim lv.yml

---
- name: create and use lvm
  hosts: all
  tasks:
    - name: check vg research exist
      debug:
        msg: "vg group does not exist"
      when: '" research" not in ansible_lvm.vgs'
     #when: '" ansible_facts.ansible_lvm.vgs.research is undefined'
    - name: create lvm
      block:
        - name: create 1500MB
          lvol:
            lv: data
            vg: research
            state: present
            size: 1500m
          when: '"research" in ansible_lvm.vgs'
      rescue:
        - name: show error
          debug:
```

```
            msg: "could not create lv of that size"
        - name: create 800M
          lvol:
            lv: data
            vg: research
            size: 800M
            state: present
          when: '"research" in ansible_lvm.vgs'
      always:
        - name: mkfs ext4
          filesystem:
            fstype: ext4
            dev: /dev/research/data
            force: yes
          when: '"research" in ansible_lvm.vgs'
```

方法二：hangzhou

```
[student@workstation ansible]$ vim lv.yml
---
- name: create and use lvm
  hosts: all
  tasks:
    - name: verify vg research exist
      shell: "vgs | grep research"
      register: vg_info
    - name: print vg message
      debug:
        msg: 'research vg does not exist'
      when: vg_info.rc !=0

    - name: create data
      block:
        - name: create 1500MB lvm
          lvol:
            lv: data
            vg: research
            state: present
            size: 1500
          when: '"research" in ansible_lvm.vgs'
      rescue:
        - name: show error message
          debug:
            msg: "could not create lv of that size"
        - name: create 800M lvm
          lvol:
            lv: data
            vg: research
            size: 800
```

```
        state: present
      when: '"research" in ansible_lvm.vgs'
  always:
    - name: create filesystem
      filesystem:
        fstype: ext4
        dev: /dev/research/data
        force: yes
      when: '"research" in ansible_lvm.vgs'
```

九、生成一个 hosts 文件

```
题目描述：
将一个初始模板文件从 http://rhgls.realm8.example.com/materials/hosts.j2 下载到 /home/greg/ansible

完成该模板，以便用它生成以下文件：针对每个清单主机包含一行内容，其格式与 /etc/hosts 相同

创建名为 /home/greg/ansible/hosts.yml 的 playbook ，它将使用此模板在 dev 主机组中的主机上生成文件 /etc/myhosts 。

该 playbook 运行后， dev 主机组中主机上的文件 /etc/myhosts 应针对每个受管主机包含一行内容：
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
172.24.2.6 node1.realm8.example.com node1
172.24.2.7 node2.realm8.example.com node2
172.24.2.8 node3.realm8.example.com node3
172.24.2.9 node4.realm8.example.com node4
172.24.2.10 node5.realm8.example.com node5
注：清单主机名称的显示顺序不重要。
```

- 下载 http://materials.example.com/hosts.j2 的初始化模板文件到 /home/student/ansible 目录
- 完成该模板，以便可以使用它以与 /etc/hosts 相同的格式生成每个库存主机的文件
- 创建一个 playbook 名为 /home/devops/ansible/hosts.yml, 来使用模板为 **dev 组**中的主机生成 /etc/myhosts 文件
- 完成时， myhosts 文件内容为：
  127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4

::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
172.25.250.10 servera.lab.example.com servera
172.25.250.11 serverb.lab.example.com serverb
172.25.250.12 serverc.lab.example.com serverc
172.25.250.13 serverd.lab.example.com serverd
172.25.250.14 servere.lab.example.com servere
注意：生成的文件顺序不对没有关系。

做题流程

- 模拟环境为：http://content.example.com/download/hosts.j2

```
[student@workstation ansible]$ wget http://content.example.com/download/hosts.j2

[student@workstation ansible]$ cat hosts.j2
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

[student@workstation ansible]$ vim hosts.j2
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
{% for host in groups['all'] %}
{{hostvars[host]['ansible_facts']['default_ipv4']['address']}} {{ hostvars[host]['ansible_
facts']['fqdn']}} {{hostvars[host]['ansible_facts']['hostname'] }}
{% endfor %}
```

实验环境编写 jinja2 模板文件，设置 fact 变量

```
[student@workstation ansible]$ cat hosts.j2
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
{% for host in groups['all'] %}
{{ ansible_facts['default_ipv4']['address'] }} {{ ansible_facts['fqdn'] }} {{ ansible_fact
s['hostname'] }}
{% endfor %}
```

编写 hosts.yml 脚本

```
[student@workstation ansible]$ vim hosts.yml
---
- name: create /etc/myhosts for dev
  hosts: all   # 这里设置 all，不然搜集不到其他主机的fact信息
  tasks:
    - name: copy j2
      template:
        src: hosts.j2
        dest: /etc/myhosts
      when: ansible_hostname in groups["dev"]
      #when: '"dev" in group_names'
```

验证

```
[student@workstation ansible]$ ansible dev -m shell -a 'cat /etc/myhosts'
servera | CHANGED | rc=0 >>
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
172.25.250.10 servera.lab.example.com servera
172.25.250.11 serverb.lab.example.com serverb
172.25.252.14 bastion.lab.example.com bastion
172.25.250.12 serverc.lab.example.com serverc
172.25.250.13 serverd.lab.example.com serverd
```

十、修改文件内容

```
按照下方所述，创建一个名为 /home/student/ansible/issue.yml 的 playbook ： 该 playbook 将在所有清单主机上运行

该 playbook 会将 /etc/issue 的内容替换为下方所示的一行文本： 在 dev 主机组中的主机上，这行文本显示为：Development

在 test 主机组中的主机上，这行文本显示 为：Test

在 prod 主机组中的主机上，这行文本显示 为：Production
```

- 创建一个 playbook 名为 /home/devops/ansible/issue.yml，修改 /etc/issuse 的内容，要求如下:

- 在 dev 主机组上，单行显示的内容是: Development
- 在 test 主机组上，单行显示的内容是: Test
- 在 prod 主机组上，单行显示的内容是： Production

做题流程

编写 issue.yml 脚本

```
[student@workstation ansible]$ vim issue.yml
---
- name: content
  hosts: all
  tasks:
    - name: content1
      copy:
        content: "Development"
        dest: /etc/issue
      when: "inventory_hostname in groups.dev"
    - name: content2
      copy:
        content: "Test"
        dest: /etc/issue
      when: "inventory_hostname in groups.test"
    - name: content3
      copy:
        content: "Production"
        dest: /etc/issue
      when: "inventory_hostname in groups.prod"
```

另一种方法是在 host_vars 中针对不同的主机定义变量

十一、创建一个 WEB 内容目录

```
按照下方所述，创建一个名为 /home/greg/ansible/webcontent.yml 的 playbook ：
该 playbook 在 dev 主机组中的受管节点上运行

创建符合下列要求的目录 /webdev ：

所有者为 webdev 组

具有常规权限：owner=read+write+execute ， group=read+write+execute ，other=read+execute
```

```
具有特殊权限：设置组 ID

用符号链接将 /var/www/html/webdev 链接到 /webdev

创建文件 /webdev/index.html ，其中包含如下所示的单行文件： Development

在 dev 主机组中主机上浏览此目录（例如 http://node1.realm8.example.com/webdev）将生成以下输出：Deve
lopment
```

- 创建一个 playbook 名字叫 /home/devops/ansible/webcontent.yml，要求如下：
- 这个 playbook 运行在 dev 主机组
  - 创建 /webdev 目录，要求如下：
  - 目录属于 webdev 组
  - 权限是 2755
  - 软链接 /var/www/html/webdev 到 /webdev
  - 在 /webdev 中创建 index.html, 内容为: Development
  - 通过 servera.lab.example.com/webdev 能访问到内容

做题流程

编写 webcontent.yml，本题使用 bastion 主机模拟

方法一：

```
- name: 创建 Web 内容目录
  hosts: dev
  tasks:
    - name: add webdev
      group:
name: webdev
state: present
- name: Create a directory if it does not exist
file:
path: /webdev

state: directory
group: webdev mode: '2775'
```

```
setype: httpd_sys_content_t

- name: Create a symbolic link
 file:
src: /webdev

dest: /var/www/html/webdev
state: link
- name: Copy using inline content

copy:

content: 'Development'
dest: /webdev/index.html
setype: httpd_sys_content_t
- name: httpd
   service:
   name: httpd
enabled: yes
      state: restarted
```

方法二：

```
[student@workstation ansible]$ vim webcontent.yml
---
- name: deploy web service
  hosts: dev
  tasks:
    - name: Ensure http install
      yum:
        name: httpd
        state: present

    - name: Ensure firewalld is enabled
      firewalld:
        service: http
        state: enabled
        permanent: yes
        immediate: yes

    - name: Restart service
      service:
        name: "{{ item }}"
        state: restarted
        enabled: yes
      loop:
        - httpd
        - firewalld
```

```
    - name: Ensure group "webdev" exists
      group:
        name: webdev
        state: present

    - name: create directory if not exist
      file:
        path: /webdev
        group: webdev
        mode: '2775'
        state: directory
        setype: httpd_sys_content_t

    - name: Set ink for web directory
      file:
        src: /webdev
        dest: /var/www/html/webdev
        state: link

    - name: touch web index.html
      copy:
        content: "Development"
        dest: /webdev/index.html
        group: webdev
        setype: httpd_sys_content_t
        mode: 644

# 可以增加以下模块
    - name: modify httpd.conf file
      replace:
        path: /etc/httpd/conf/httpd.conf
        regexp: 'Options Index FollowSymLinks'
        replace: 'Options FollowSymLinks'
      notify: restart httpd service
  handlers:
    - name: restart httpd service
      service:
        name: httpd
        state: restarted

[student@workstation ansible]$ curl http://servera/webdev/
Development
```

使用浏览器访问

http://bastion.lab.example.com/webdev/

```
Indexes 的作用就是当该目录下没有 index.html 文件时，就显示目录结构，去掉 Indexes，Apache就不会显示该目录的列表了。
```

在Options IndexesFollowSymLinks在Indexes前面加上 – 符号。
即： Options -Indexes FollowSymLinks
【备注：在Indexes前，加 + 代表允许目录浏览；加 –代表禁止目录浏览。】

```
---
- name: deploy web service
  hosts: dev
  tasks:
    - name: Ensure http install
      yum:
        name: httpd
        state: present
    - name: Restart service
      service:
        name: "{{ item }}"
        state: restarted
        enabled: yes
      loop:
        - httpd
        - firewalld

    - name: Ensure firewalld is enabled
      firewalld:
        service: http
        state: enabled
        permanent: yes
        immediate: yes


    - name: Ensure group "webdev" exists
      group:
        name: webdev
        state: present

    - name: create directory if not exist
      file:
        path: /webdev
        group: webdev
        mode: '2775'
        state: directory
        setype: httpd_sys_content_t

    - name: Set ink for web directory
      file:
        src: /webdev
        dest: /var/www/html/webdev
        state: link
```

```
    - name: touch web index.html
      copy:
        content: "Development"
        dest: /webdev/index.html
        group: webdev
        setype: httpd_sys_content_t
        mode: 644
    - name: restart httpd service
      service:
        name: httpd
        state: restarted
```

十二、创建硬件报告

```
题目描述：
生成硬件报告
创建一个名为 /home/greg/ansible/hwreport.yml 的 playbook ，它将在所有受管节点上生成含有以下信息的输
出文件 /root/hwreport.txt ：

清单主机名称
以 MB 表示的总内存大小
BIOS 版本
磁盘设备 vda 的大小
磁盘设备 vdb 的大小
输出文件中的每一行含有一个 key=value 对。

您的 playbook 应当：
从 http://rhgl.realm8.example.com/materials/hwreport.empty 下载文件，并将它保存为 /root/hwrepo
rt.txt

使用正确的值改为 /root/hwreport.txt
如果硬件项不存在，相关的值应设为 NONE
```

创建硬件报告：

- 创建一个 名为 /home/devops/ansible/hwreport.yml 的 playbook ，在所有的受管主机上创建 /root/hwreport.yml, 包含如下内容：
  - inventory 的主机名
  - Total memory in MB

- BIOS version
- vda size
- vdb size
- 每行均为一个键值对
- playbook 会下载 http://materials.example.com/hwreport.empty 文件，并修改成为 hwreport.txt
- 如果硬件不存在，则对应的值为：NONE

做题流程

方法一：

```
---
- name: report harware message
  hosts: all
  vars:
    hw_all:
      - hw_name: HOST
        hw_cont: "{{ inventory_hostname | default('NONE', true) }}"
      - hw_name: MEMERY
        hw_cont: "{{ ansible_memtotal_mb | default('NONE', true) }}"
      - hw_name: BIOS
        hw_cont: "{{ ansible_bios_version | default('NONE', true) }}"
      - hw_name: DISK_SIZE_VDA
        hw_cont: "{{ ansible_devices.vda.size | default('NONE', true) }}"
      - hw_name: DISK_SIZE_VDB
        hw_cont: "{{ ansible_devices.vdb.size | default('NONE', true) }}"
  tasks:
    - name: copy template file
      get_url:
        # url: http://materials/hwreport.empty
        url: http://content/download/hwreport.empty
        dest: /root/hwreport.txt
    - name: update report
      lineinfile:
        path: /root/hwreport.txt
        regexp: '^{{ item.hw_name }}='
        line: "{{ item.hw_name }}={{ item.hw_cont }}"
      loop: "{{ hw_all }}"
```

方法二：

```
---
- name: print hardware message
  hosts: all
  tasks:
    - name: download file hwreport.empty
      get_url:
        url: http://content/download/hwreport.empty
        dest: /root/hwreport.txt
    - name: add inventory hostname info
      lineinfile:
        path: /root/hwreport.txt
        regexp: '^HOST='
        line: HOST={{inventory_hostname}}
    - name: add memory info
      lineinfile:
        path: /root/hwreport.txt
        regexp: '^MEMORY='
        line: MEMORY={{ ansible_memtotal_mb }}

    - name: add bios info
      lineinfile:
        path: /root/hwreport.txt
        regexp: '^BIOS='
        line: BIOS={{ ansible_bios_version }}
    - name: add vda info
      lineinfile:
        path: /root/hwreport.txt
        regexp: '^DISK_SIZE_VDA'
        line: DISK_SIZE_VDA={{ ansible_devices.vda.size }}

    - name: add vdb info
      lineinfile:
        path: /root/hwreport.txt
        regexp: '^DISK_SIZE_VDB'
        line: DISK_SIZE_VDB={{ ansible_devices.vdb.size | default('NONE',true) }}
```

```
regexp:要在文件的每一行中查找的正则表达式。在修改行时，regexp通常应该匹配行的初始状态以及行替换后的状态，以确保等幂。

所有匹配 ^DISK_SIZE_VDB= 的行进行替换
```

验证

```
[student@workstation ansible]$ ansible all -a 'cat /root/hwreport.txt'
# Hardware report
HOST=servera
MEMORY=821
BIOS=1.11.1-3.module+el8+2529+a9686a4d
DISK_SIZE_VDA=10.00 GB
DISK_SIZE_VDB=5.00 GB
```

十三、创建一个包含密码的 vault 文件

```
题目描述：
按照下方所述，创建一个 Ansible 库来存储用户密码：
库名称为 /home/greg/ansible/locker.yml
库中含有两个变量，名称如下：
pw_developer，值为 Imadev
pw_manager，值为 Imamgr
用于加密和解密该库的密码为 "rhcsa8rhce8"
密码存储在文件 /home/greg/ansible/secret.txt 中
```

- vault 文件名：locker.yml，其中包含两个变量：
  - pw_developer: lmadev
  - pw_manager: lmamgr
  - vault 的加解密密钥为： redhat
  - 密钥保存在 /home/studnet/ansible/secret.txt

做题流程：

编写密码库文件

```
[student@workstation ansible]$ vim locker.yml
---
- pw_developer: Imadev
- pw_manager: Imager
```

编写密码文件

```
[student@workstation ansible]$ vim secret.txt
rhcsa8rhce8
```

也可以在 ansible.cfg 中定义 vault_password_file 指定密码文件路径

```
[student@workstation ansible]$ vim ansible.cfg
vault_password_file = /home/greg/ansible/secret.txt
```

使用 ansible-vault 给库文件加密

```
[student@workstation ansible]$ ansible-vault encrypt --vault-id=/home/student/ansible/secr
et.txt locker.yml
Encryption successful

# 使用 --vault-password-file=
[student@workstation ansible]$ ansible-vault encrypt --vault-password-file=/home/ student/
ansible/secret.txt locker.yml
Encryption successful

[student@workstation ansible]$ cat locker.yml
$ANSIBLE_VAULT;1.1;AES256
35666634336434333631306438373632383661653764353932653664316234313439616164643633
34373337663932333731656232626337343635623131646 30a643837313165633736633239303935
```

查看加密后的库文件

```
[student@workstation ansible]$ ansible-vault view locker.yml
Vault password:
---
- pw_developer: Imadev
- pw_manager: Imager
```

十四、创建用户账户

```
从 http://materials.example.com.com/materials/user_list.yml 下载要创建的用户的列表，并将它保存到
/home/greg/ansible

在本次考试中使用在其他位置创建的密码库 /home/greg/ansible/locker.yml 。创建名为 /home/greg/ansibl
e/users.yml 的 playbook ，从而按以下所述创建用户帐户：
职位描述为 developer 的用户应当：
在 dev 和 test 主机组中的受管节点上创建
从 pw_developer 变量分配密码
是补充组 devops 的成员
密码30天后过期

职位描述为 manager 的用户应当：
在 prod 主机组中的受管节点上创建
从 pw_manager 变量分配密码
是补充组 opsmgr 的成员

密码采用 SHA512 哈希格式。
– 密码30天后过期
– 分配相应ID
您的 playbook 应能够在本次考试中使用在其他位置创建的库密码文件 /home/greg/ansible/secret.txt 正常运
行
```

- 使用 http://materials.example.com/user_list.yml 文件和 /home/devops/ansible/locker.yml 文件，创建 playbook 文件 users.yml 以创建用户
  - dev 和 test 主机组中创建带有工作描述为: devopler 的用户，密码使用 pw_developer 变量， 用户的附属组为 devops
  - prod 主机组中创建带有工作描述为: manager 的用户，密码使用 pw_manager 变量， 用户的附属组为 opsmgr
  - 密码均基于 SHA512 hash 格式创建
  - 密码最大有效期为 30 天

做题流程

下载用户列表文件

```
# 模拟环境下载地址 content.example.com/download/user_list.yml
[student@workstation ansible]$ wget content.example.com/download/user_list.yml

[student@workstation ansible]$ cat user_list.yml
---
users:
```

```
 - name: node1
   job: developer
 - name: node2
   job: developer
 - name: node3
   job: manager
```

编写脚本

```
[student@workstation ansible]$ vim users.yml
---
- name: create user on dev and test
  hosts: dev,test
  vars_files:
    - user_list.yml
    - locker.yml
  tasks:
    - name: create group
      group:
        name: devops
        state: present
    - name: create user from var_files
      user:
        name: "{{ item.name }}"
        password: "{{ pw_developer | password_hash('sha512')}}"
        groups: devops
      loop: "{{ users }}"
      when: item.job == 'developer'
    - name: chage user password policy
      shell: "chage -M 30 '{{ item.name }}'"
      loop: "{{ users }}"
      when: item.job == 'developer'
- name: create user on prod
  hosts: prod
  vars_files:
    - user_list.yml
    - locker.yml
  tasks:
    - name: create group
      group:
        name: opsmgr
        state: present
    - name: create user from var_files
      user:
        name: "{{ item.name }}"
        password: "{{ pw_manager | password_hash('sha512')}}"
        groups: opsmgr
```

```
      loop: "{{ users }}"
      when: item.job == 'manager'
    - name: chage user password policy
      shell: "chage -M 30 '{{ item.name }}'"
      loop: "{{ users }}"
      when: item.job == 'manager'


[student@workstation ansible]$ ansible-playbook --vault-password-file=secret.txt users.yml
或者
[student@workstation ansible]$ ansible-playbook --vault-id=./secret.txt users.yml
```

```
password_expire_max
integer 整数
added in 2.11 of ansible.builtin
Maximum number of days between password change.
Supported on Linux only.

password_expire_min
integer 整数
added in 2.11 of ansible.builtin
Minimum number of days between password change.
Supported on Linux only.
```

十五、修改 ansible vault 的密码

```
按照下方所述，更新现有 Ansible 库的密钥：
从 http://materials/salaries.yml 下载 Ansible 库到 /home/student/ansible
当前的库密码为 insecure8sure
新的库密码为 bbs2you9527
库使用新密码保持加密状态
```

- 修改 http://materials.example.com/salaries.yml 文件的密码，并保存到 /home/student/ansible/salaries.yml
  - 原来的密码为 insecure4sure
  - 新密码为 redhat123

做题流程

模拟环境下载地址：content.example.com/download/salaries.yml

```
[student@workstation ansible]$ wget content.example.com/download/salaries.yml

[student@workstation ansible]$ ansible-vault rekey salaries.yml

[student@workstation ansible]$ ansible-vault view salaries.yml
```

十六、创建定时计划任务

- 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 cron.yml.要求所有主机 natasha 成员，执行调度周期性Cron作业，要求每两分钟执行以下命令 logger "EX294 in progress"。
- 这一题可以和其他题目组合，其实就是多写一个任务而已，注意灵活应对

```
---
- name: configure crontab
  hosts: prod
  tasks:
    - name: create user
      user:
        name: natasha
        state: present
    - name: create crontab
      cron:
        minute: '*/2'
        user: natasha
        job: logger "EX294 in progress"
```