RHCSA

考前须知

- 1、同学把备注改一下，例如：北京-Linux-张三
- 2、线下班例如修改成：北京-1901Linux-张三
- 3、RHCE RHCSA练习环境在群公告里面，自行下载，学习资料在群文件和课程学习资料里面
- 4、技术问题找强哥老师解答，预约考试找解老师
- 5、预约考试，提前半个月告知解老师，把考试城市、个人信息、红帽邮箱私聊给解老师
- 6、需带身份证进入考场
- 7、记号自己注册的红帽邮箱，考前考官会让你确认姓名和邮箱是否有误

考试做题准备

- 1、点开左上角红帽logo打开终端
- 2、做题前先查看yum-config-manager是否能用，没有需手动配置yum源(/etc/yum.repo.d/)

环境准备

RHCSA做题的虚机node1会提供root密码，node2需要自己破解root密码

注意VIM可能要配置完yum源才能安装使用

在练习期间，您将操作下列虚拟系统：

System	IP Address	Role
node1.domain250.example.com	172.25.250.100	需要配置的虚拟机
node2.domain250.example.com	172.25.250.200	需要配置的虚拟机
classroom(yum源仓库,NTP服务器),utility(podman容器镜像仓库)		保持开机状态

帐户信息： f0主机的root密码为Asimov， f0上其他用户的密码均为redhat。f0里面所有虚拟系统的 root 密码是 redhat，请勿更改 root 密码。 serverb的root密码未知，需要进行密码重 置，要求请查看后面的题目。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 配置文件进行任何修改

开启虚拟机：

```
[kiosk@foundation0 ~]$ rht-vmctl reset all 或者 rht-vmctl start all
#下面四行忽略
[root@foundation0 ~]# virsh start bastion
[root@foundation0 ~]# virsh start node1
[root@foundation0 ~]# virsh start node2
[root@foundation0 ~]# virsh list --all
```

说明：考试需要通过图形界面对虚拟机进行开机(start)，关机(poweroff)，重启(reboot) 和重置(rebuilt)操作，重置虚拟机后，虚拟机所有的配置将会清空

---

node1

一、配置网络环境

主机名： node1.domain250.example.com IP 地址: 172.25.250.100/24 网关： 172.25.250.254 DNS： 172.25.250.254

设置主机名
```
# hostnamectl set-hostname node1.domain250.example.com
```
查看所有网络接口信息
```
# nmcli connection show
```
设置静态IP、配置开机自动激活
```
# nmcli connection modify "Wired connection 1" ipv4.addresses 172.25.250.100/24 ipv4.gateway 172.25.250.254 ipv4.dns 172.25.250.254
ipv4.method manual connection.autoconnect yes
```
激活配置
```
# nmcli connection up "Wired connection 1"
```
检查网卡是否配置成功
```
# ifconfig
```
检查主机名是否配置成功
```
# hostname
```

注意事项

1、设置网卡时需注意网卡名是否带有特殊符号，包括含有空格，都需要用引号把名字括起来

2、设置网卡时需注意*method*前面是要有*ipv4*.的，发现部分同学会漏了这个

二、配置yum源

http://foundation0.ilt.example.com/dvd/BaseOS

http://foundation0.ilt.example.com/dvd/AppStream

yum-config-manager可用的情况下
```
# yum-config-manager --add-repo "http://foundation0.ilt.example.com/dvd/BaseOS"
# yum-config-manager --add-repo "http://foundation0.ilt.example.com/dvd/AppStream"
```
手动配置yum源
```
# vi /etc/yum.repo.d/foundation0.ilt.example.com_dvd_BaseOS.repo
#yum源库的名字，方便在执行yum源时显示库是否配置成功
[foundation0.ilt.example.com_dvd_BaseOS]
name=http://foundation0.ilt.example.com_dvd_BaseOS
#yum源库存在的路径，告诉yum源从哪里能找到软件
baseurl=http://foundation0.ilt.example.com/dvd/BaseOS
#enabled为1是用来提醒yum源这个库是使用的
enabled=1
#gpgcheck为0是用来提醒yum源这个库不需要校验
gpgcheck=0
# vi /etc/yum.repo.d/foundation0.ilt.example.com_dvd_AppStream.repo
[foundation0.ilt.example.com_dvd_AppStream]
name=foundation0.ilt.example.com_dvd_AppStream
baseurl=http://foundation0.ilt.example.com/dvd/AppStream
enabled=1
gpgcheck=0
```
清空yum源
```
# yum clean all
```
初始化并列出所有可用yum源
```
# yum repolist
```
(考试机可能没有vim)安装vim测试yum源是否能正常使用
```
# yum install vim -y
```

注意事项

1、配置*yum*源时需检查是否能用$yum-config-manager$

2、*yum*源库的名字可以自定义，除非题目有要求

3、填写*yum*源的*baseurl*路径时能复制尽量复制别手敲

三、配置Selinux

在非标准端口82上运行WEB服务遇到问题，请解决： web 服务器能够提供/var/www/html 中所有现在有的 html 文件 Web 服务器通过端口82访问 Web 服务器在系统启动时能自动启动

检查Selinux是否开启
```
# egrep -i selinux /etc/sysconfig/selinux |egrep -v "#"
```

```
selinux未开启就需要进配置文件把selinux设置为enforcing
# vim /etc/sysconfig/selinux
SELINUX=enforcing
重启使其生效
# reboot
检查semanage是否有安装
# semanage -h
如果没有需安装semanage
# yum install policycoreutils-python -y
先进/etc/httpd/conf/httpd里看看端口号是多少
# egrep -i listen /etc/httpd/conf/httpd.conf |egrep -v '#'
重启apache
# systemctl restart httpd
启动apche报错查看错误日志
# journalctl -xn
日志里会出现一行"# semanage port -a -t PORT_TYPE -p tcp 82"
使用semanage放通Selinux安全策略端口
# semanage port -a -t http_port_t -p tcp 82
检查端口是否放通成功
# semanage port -l | grep 'http'
重启apache并查看是否启动成功
# systemctl restart httpd
# netstat -lnpt|grep httpd
检查页面是否正常
# systemctl restart httpd
# netstat -lnpt|grep httpd
无法访问
使用restorecon来恢复SELinux文件属性即恢复文件的安全上下文
# restorecon -Rv /var/www/html/
再次检查
# curl http://servera.lab.example.com:82/a.htm
```

注意事项

- *Selinux在做题到交卷必须一直开着，否则RHCSA全部0分*

四、用户及组管理

组名为 sysmgrs natasha 用户的附属组是 sysmgrs harry 用户的附属组是 sysmgrs sarah 用户的shell为非交互式shell，但不是 sarah 组的成员 natasha、harry、sarah 的密码是 flectrag

```
创建用户组
# groupadd sysmgrs
创建用户并给用户附加一个组
# useradd natasha -G sysmgrs
# useradd harry -G sysmgrs
创建用户并指定用户不能登录到终端
# useradd sarah -s /sbin/nologin
检查组是否添加成功
# grep sysmgrs /etc/group
检查用户是否添加成功
# tail -3 /etc/passwd
或
# grep /bin/bash /etc/passwd
# grep sarah /etc/passwd
明文设置用户密码
# echo "flectrag" | passwd --stdin natasha
# echo "flectrag" | passwd --stdin harry
# echo "flectrag" | passwd --stdin sarah
检查密码是否设置成功
# su - natasha
$ su - harry
$ su - natasha
$ su - sarah
$ exit
```

- **注意事项**

- 1、*root使用su切换时是不需要输入密码，所以要切换到别的用户再切回上个用户测试密码*
- 2、因为*zhuzhuxia*这个用户设置了无法登陆终端的权限，所以*su - zhuzhuxia*会提示该用户不可用

五、计划任务cron

配置 cron 作业，每隔 25 分钟运行以下命令： logger "EX200 in progress"，以用户 natasha 身份运行

配置 cron 作业，每天14点23分运行以下命令： logger "EX200 in progress"，以用户 natasha 身份运行

```
查看服务是否为开机自启状态
# systemctl status crond
创建natasha的计划任务
# crontab -e -u natasha
*/2 * * * * logger "EX200 in progress"
23 14 * * * logger "EX200 in progress"
检查
# crontab -u natasha -l
```

命令说明

Minute	Hours	Day-of-Month	Month	Day-of-Week	Command
分钟	小时	日	月	周（星期）	命令
0-59	0-23	1-31	1-12	0-6	job

时间： *:每 */5:每隔分钟 ,:不同的时间段

-:表示范围

星期0为星期日

六、目录权限设置

创建目录，并按以下要求设置： mkdir /home/managers **目录属于 sysmgrs 组 目录可以被 sysmgrs 的组成员读取、写入和访问，其他任何用户不具 备这些权限（root 用户除外） 在/home/managers 目录中创建的文件的所属组自动变成 sysmgrs 组**

```
创建managers文件夹
# mkdir /home/managers
改变文件夹的属组
# chown :sysmgrs /home/managers
或
# chgrp sysmgrs /home/managers
修改用户对文件夹使用权限，2为赋予sgid隐藏权限
chmod赋予的隐藏权限分为:"4"是SUID普通用户提权，普通用户对该文件的所有操作相当于是root权限(只针对二进制、可执行的文件);"2"是SGID获得该程序所属用户组的权限，这个用户组的所有用户在该文件夹下创建的任何一个文件夹或文件它们的群组都会与此目录的群组相同;"1"是SBIT限制用户对文件执行的权限，当用户在该目录下建立文件或目录时，仅有自己与 root才有权力删除。
# chmod 2770 /home/managers
检查文件夹是否成功
# ll -d /home/managers
或
# ls -ld /home/managers
```

命令说明

chown 可以用来更改文件的所有者和所有组，如chown harry:harry /home/ ，意为把home的所有者和所有组改成harry

chgrp 允许普通用户改变文件所属的组，只要该用户是该组的一员，但不能改变所有者

chmod 控制用户对文件的权限的命令;chmod赋予的高级权限分为:"4"是SUID普通用户提权，普通用户对该文件的所有操作相当于是root权限(只针对二进制、可执行的文件);"2"是SGID获得该程序所属用户组的权限，这个用户组的所有用户在该文件夹下创建的任何一个文件夹或文件它们的群组都会与此目录的群组相同;"1"是SBIT限制用户对文件执行的权限，当用户在该目录下建立文件或目录时，仅有自己与 root才有权力删除。

2770 "2"为sgid,"7"为wrx(w读 + r 写 + x 执行)，"0"为什么权限都没有

注意事项

- 注意题目要求要不要赋予高级权限

七、autofs自动挂载

配置 autofs，自动挂载远程用户的家目录： NFS 服务器 172.25.250.254(dns别名： materials.example.com) 共享/rhome，包含用户remoteuser1家目录 remoteuser1 的家目录是 172.25.254.254:/rhome/remoteuser1 remoteuser1 的家目录应该自动挂载到/rhome/remoteusr1 家目录允许remoteuser1用户写入 remoteuser1 从共享目录里导出文件

```
安装autofs挂载文件系统服务
# yum -y install autofs
autofs主配置文件
# vim /etc/auto.master
/rhome /etc/auto.rhome
配置子配置文件按照“挂载目录 挂载文件类型及权限 :设备名称”的格式进行填写，可以参考"/etc/auto.misc"的格式
# vim /etc/auto.rhome
remoteuser1 -rw,sync 172.25.254.254:/rhome/remoteuser1
另一种写法:
remoteuser1 -rw,sync materials.example.com:/rhome/remoteuser1
重启服务并设置开机自启
# systemctl restart autofs
# systemctl enable autofs
检查是否部署完成
# su - remoteuser1
$ pwd
$ df -Th
```

命令说明

/etc/exports文件:是NFS的一个配置文件，NFS在运行时会去读这个文件

/rhome 172.25.254.254/24(rw,sync,no_root_squash):"/rhome":nfs的共享路径,这个路径必须真实存在;"172.25.254.254/24":表示在这个网段内的所有主机都可以挂载/rhome,单写一个ip也可以;"(rw,sync,no_root_squash)":rw表示以读写方式进行挂载;sysnc表示数据同步写入内存和磁盘;no_root_squash表示客户端使用root身份登录，获得所有权限

- 如果切换到用户左边显示为bash说明autofs子配置文件写错或是需要更改挂载文件夹的权限

八、配置文件权限

将文件/etc/fstab 复制到/var/tmp/fstab，并配置权限： /var/tmp/fstab 属于 root 用户和 root 组 /var/tmp/fstab 不能被任何人执行 用户 natasha 有读写权限 用户 harry 没有读写权限 所有其他用户包括以后的用户只能够读取

```
拷贝fstab文件
# cp /etc/fstab /var/tmp
查看文件是否拷贝成功
# ls -lh /var/tmp/fstab
查看acl基本权限
# getfacl /var/tmp/fstab6/21
设置acl基本权限
# setfacl -m u:natasha:rw /var/tmp/fstab
# setfacl -m u:harry:- /var/tmp/fstab
查看acl基本权限
# getfacl /var/tmp/fstab
```

命令说明

acl访问控制权限

setfacl 给文件添加访问控制权限,"setfacl -m u:natasha:rw /var/tmp/fstab"给文件fstab添加用户natasha并赋予这个用户读写权限;"setfacl -m u:harry:-/var/tmp/fstab"给文件fstab添加用户harry并设置这个用户没有权限操作文件

九、时间客户端NTP

配置 node1 为 NTP 的客户端，与时间服务器172.25.254.254(materials.example.com) 同步时间

```
安装NTP客户端chrony
# yum -y install chrony
设置NTP同步地址
# vim /etc/chrony.conf
(可参考文件里面的写法，把中间的地址换成需要同步的地址)
server 172.25.254.254 iburst
或
server materials.example.com iburst
重启服务并设置开机自启
# systemctl restart chronyd
# systemctl enable chronyd
检查时间是否同步成功
# timedatectl
```

十、创建用户

创建用户manalo，用户的 ID 为 3533，用户密码为flectrag

```
创建manalo用户并设置uid为3533
# useradd manalo -u 3533
# echo "flectrag" | passwd --stdin manalo
检查方法可参考第四题
```

十一、查找文件

查找属于jacques 用户所属的文件，并拷贝到/root/findfiles目录

```
想脚本有分数就执行下面这句，考试不需要
# touch /tmp/{gamelan,jacques,libWedgeit.so.1.2.3}
# chown jacques:jacques /tmp/{gamelan,jacques,libWedgeit.so.1.2.3}
先创建一个存放jacques 用户所有文件的文件夹
# mkdir /root/findfiles
使用find文件搜索命令查找用户文件并拷贝到刚创建的文件夹中
# find / -user jacques -exec cp -a {} /root/findfiles \;
```

命令说明

find / -user jacques -exec cp -a {} /root/findfiles \;

"find":文件搜索命令;"/":表示从根目录开始查找;"-user":表示查找文件时只搜索和这个用户相关的文件;"-exec":表示调用并执行一条shell命令，"{}"为前面查找到的内容,"\;"为"exec"的命令结束符;"cp":拷贝命令，"-a":在这里表示复制所有查找到的文件。

注意事项

- 使用exec命令，该命令的格式必须要有"{}"和"\;"

十二、查找文件中的内容

查找文件/usr/share/xml/iso-codes/iso_639_3.xml 中包含字符串 ng 的所有行

将找到的内容按原有顺序放到文件/root/list 中，/root/list不能包含空行

```
使用grep命令限制需要查找的内容，如果内容没有特殊符号可以不用加引号
# grep "ng" /usr/share/xml/iso-codes/iso_639_3.xml > /root/list
```

十三、文件归档

创建为/root/backup.tar.gz或/root/backup.tar.bz2 的归档包，用来压缩/etc目录

```
tar打包并压缩文件(考试时注意看题目使用参数)
# tar -czvf /root/backup.tar.gz /etc
# tar -cjvf /root/backup.tar.bz2 /etc
检查是否打包压缩成功
# ll /root/backup.tar.gz
# file /root/backup.tar.gz
```

命令说明

tar -czvf /root/backup.tar.gz /etc

tar打包命令中:

- "c":为指定归档包的创建路径;
- "a":为自定义归档包类型，可以是gz，bz2;
- "z":为把指定的文件压缩成gzip(gz)属性;
- "j":为把指定的文件压缩成bz2属性;
- "v":显示打包的详细过程，这个参数"v"可以不加;
- "f":指定需要被打包的文件

file辨识文件类型命令,用来告诉你这个文件是什么类型的

注意事项

注意看题目是要求打包成什么类型的归档包

十四、设置sudo免密

允许sysmgrs组所有成员使用sudo时不需要密码

```
编辑/etc/sudoers配置文件，进入第110行
# vi /etc/sudoers +110
##vi命令行模式yy把现在选中的这条复制下来，p粘贴
# %wheel        ALL=(ALL)       NOPASSWD: ALL
##组名改成sysmgrs
%sysmgrs        ALL=(ALL)       NOPASSWD: ALL
```

十五、配置创建新用户的密码策略

创建新用户时，默认密码策略为20天后，密码会过期

```
编辑/etc/login.defs新建用户策略文件，进入第25行
# vi /etc/login.defs +25
##修改当行的数字为20
PASS_MAX_DAYS   20
```

十六、配置容器以使其自动启动

容器题外话

利用注册表服务器上的 rsyslog 镜像，创建一个名为 logserver 的容器 将其配置为以 systemd 服务的形式运行，且仅面向现有用户 wallah 该服务应命名为 container-logserver，并应在系统重新引导后自动启动 而无需任何手动干预

```
考试前先检查node1有没有临时日志信息
$ ssh root@node1
# ll -d /run/log/journal
drwxr-sr-x. 3 root systemd-journal 60 Mar 24 17:56 /run/log/journal
没有就创建
# mkdir /var/log/journal
# chown root:systemd-journal /var/log/journal
# chmod 2755 /var/log/journal
# systemctl restart systemd-journald
# cp /var/log/journal/*/system.journal /home/walhalla/container_logfile/
# chown -R wallah ~wallah
# ll -Z /home/wallah/container_logfile/
设置用户密码，知道密码可以不用
# echo redhat|passwd --stdin wallah
设置容器永久化
# vim /etc/systemd/journald.conf
#Storage=auto
改为
Storage=persistent
# exit
# ssh wallah@node1
登录到仓库
$ podman login registry.lab.example.com
admin
redhat321
搜索镜像
$ podman search registry.domain250.example.com/
INDEX NAME DESCRIPTION STARS
OFFICIAL AUTOMATED
example.com registry.domain250.example.com/rhel8/rsyslog 0
...
$ podman run -d --name logserver -v /home/wallah/container_logfile:/var/log/journal:Z registry.domain250.example.com/rhel8/rsyslog
停掉容器
$ podman stop logserver
/.config/systemd/user
检查systemd的状态
$ loginctl enable-linger
$ loginctl show-user wallah
创建存放进程目录
$ mkdir -p ~/.config/systemd/user/
$ cd ~/.config/systemd/user/
生成一个进程
$ podman generate systemd -n logserver -f
$ systemctl --user enable --now container-logserver
$ systemctl --user status container-logserver
执行容器语句
$ podman exec logserver ls /var/log/journal
system.journal
user-1004.journal
$ podman exec logserver logger -p authpriv.info suibian
```

附加题

以下题型考试随机出一题

一、创建shell脚本

由于考完试的同学说得有点乱，现在还不能确定题目是否为查找小于10M并且有组id的文件，所以写了一个大概，准确性还是要最近考试的小白鼠验证(同学)

在/usr/bin目录下创建一个repwis脚本，查找/usr目录大小范围在10-50K之间并且组id不为root的文件且拥有sgid的文件，把查到的文件结果拷贝到/root/myfiles文件夹内

```
创建脚本文件,文件类型可以不用指定
#vi /usr/bin/repwis

#!/bin/bash
```

```
##使用awk过滤出GID(组ID)
for i in `awk -F':' '{print $3}' /etc/group`
do
##提醒脚本跑到哪个GID
    echo GID is $i
##由于我自己的环境root组文件过多，为了检验脚本，就把root用户过滤掉
    if [ 0 != $i ];then
##使用find查找，没有特殊要就就只要find这行
        find /usr -size +10k -size -50k -gid $i -perm -4000 -exec cp -a {} /root/myfiles \;
    fi
done
检查拷贝的文件是否符合
#ll -h /root/myfiles/
或
#ls -lh /root/myfiles/
```

脚本说明

(#!/bin/bash):"#!"是一个约定的标记，用来告诉系统该用什么编译器执行这个脚本,/bin/bash:也就是说使用默认Shell 。

在这里需要用到for循环对查找到的GID(组id)进行遍历处理。

(awk -F':' '{print $3}' /etc/group):"awk"用来对文本进行分割处理;"-F"指定一个分隔符,":"使用:(引号)来分隔文本内容;"{print $3}":执行打印语句,打印出第三列的内容。

```
例:
取第一行的内容
#awk 'NR==1 {print $1}' /etc/group
root:x:0:
以:分割，打印出分割后的第一列、第二列、第三列内容
#awk -F':' 'NR==1 {print $1,$2,$3}' /etc/group
root x 0
以:分割，打印出分割后的第一列内容
#awk -F':' 'NR==1 {print $1}' /etc/group
root
```

find -perm用于指定权限，suid 4000 sgid 2000 两者都有为6000 sbit 1000

二、欢迎语

附加题，考试随机出一题

登录到普通用户后提示一句欢迎语'hello word !'或者用户在登陆前显示欢迎语提示

```
配置登录后的欢迎语
重定向欢迎语写入到/etc/motd
# echo 'hello word !' >> /etc/motd
配置登录前的欢迎语
重定向欢迎语写入到/etc/issue
# echo 'hello word !' >> /etc/issue
从底层登录到系统检查
```

三、设置用户默认创建文件UMASK

附加题，考试随机出一题

用户natasha默认创建文件权限为r--r--r--默认创建的目录为r-xr-xr-x

```
# su - natasha
把umask写进用户家目录下的bashrc
echo 'umask 222'>>.bashrc
```

命令说明

umask用于控制用户的默认创建权限

umask 第一位是特殊权限默认为0加不加都行，没需求只用输三位，第二位属主权限，第三位组权限，第四位其他人权限

UMASK值	文件	目录

UMASK值	文件	目录
0	6	7
1	6	6
2	4	5
3	4	4
4	2	3
5	2	2
6	0	1
7	0	0

三、生成环境变量

生成一个环境变量rhcsa，用户在使用这个变量时输出一句话"This is RHCSA!"

自定义一个命令rhce，用户在使用这个变量时输出一句话"This is RHCE!"

```
使用export生成环境变量，并写进用户家目录下的bashrc
# echo 'export rhcsa="echo This is RHCSA!"' ~/.bashrc
自定义命令
su - user
$ echo 'alias rhce="echo This is RHCE!"'>>~/.bash
或
#cat >>/usr/local/bin/rhce<<eof
> #!/bin/bash
> echo This is RHCE!
> eof
# chmod +x /usr/local/bin/rhce
```

命令说明

~/用户家目下

export:设置或显示环境变量（只在当前shell生效）

cat>>file<<eof重定向用于创建文件或在文件内追加内容，主要用于分行写入文件，结束时需要写个eof

---

node2

一、*重置root密码*

将 node2 主机的root密码设置成flectrag

```
重启
内核选择界面按e，将光标移到Linux 那一行末尾添加如下内容：rd.break console=tty0
按下：Ctrl + X进入grup救援模式
重新挂载/目录
# mount -o rw,remount /sysroot
进入/目录
# chroot /sysroot
设置root密码
# echo "flectrag"|passwd --stdin root
或
下面设置密码需要敲两次密码
# passwd
flectrag
flectrag
在根目录下创建重新刷新SELinux 安全上下文标记的文件
```

```
# touch /.autorelabel
退出重启系统
# exit
# reboot
```

注意事项

1、rw和remount之间的符号是 ”,” (逗号)

2、touch创建的是一个在/目录下的隐藏文件所有是.autorelabel，别弄错成在当前目录下创建autorelabel

---

二、配置yum源

foundation0.ilt.example.com

foundation0.ilt.example.com

```
yum-config-manager可用的情况下
# yum-config-manager --add-repo "http://foundation0.ilt.example.com/dvd/BaseOS"
# yum-config-manager --add-repo "http://foundation0.ilt.example.com/dvd/AppStream"
手动配置yum源
# vi /etc/yum.repo.d/http://foundation0.ilt.example.com/dvd/BaseOS.repo
#yum源库的名字，方便在执行yum源时显示库是否配置成功
[http://foundation0.ilt.example.com/dvd/BaseOS]
name=http://foundation0.ilt.example.com/dvd/BaseOS
#yum源库存在的路径，告诉yum源从哪里能找到软件
baseurl=http://foundation0.ilt.example.com/dvd/BaseOS
#enabled为1是用来提醒yum源这个库是使用的
enabled=1
#gpgcheck为0是用来提醒yum源这个库不需要校验
gpgcheck=0
# vi /etc/yum.repo.d/http://foundation0.ilt.example.com/dvd/AppStream.repo
[http://foundation0.ilt.example.com/dvd/AppStream]
name=http://foundation0.ilt.example.com/dvd/AppStream
baseurl=http://foundation0.ilt.example.com/dvd/AppStream
enabled=1
gpgcheck=0
清空yum源
# yum clean all
初始化并列出所有可用yum源
# yum repolist
(考试机可能没有vim)安装vim测试yum源是否能正常使用
# yum install vim -y
```

注意事项

1、配置yum源时需检查是否能用$yum-config-manager$

2、yum源库的名字可以自定义，除非题目有要求

3、填写yum源的baseurl路径时能复制尽量复制别手敲

三、调整逻辑卷

将逻辑卷lv01的大小调整到230M，确保文件系统的内容保持不变 调整后的逻辑卷的大小范围在217到243M的范围内都是可以接受的

```
查看需要调整的逻辑卷所挂载的路径和现有的大小、名字
# lvscan
查看卷组的容量有多大
# vgs
使用lvm逻辑卷扩容
# lvextend -L 230M /dev/myvol/vo
检查lvm大小是否符合
# lvscan
重新扩容过的文件系统
# resize2fs /dev/myvol/vo
或
# xfs_growfs /dev/myvol/vo
查看是否扩容成功
# df -Th
```

命令说明

lvextend -L 230M /dev/myvol/vo:"-L"指定逻辑卷的大小;"300M"指定要扩展的大小可以为KB,MB,GB;" /dev/myvol/vo":指定要扩容的逻辑卷路径

resize2fs /dev/myvol/vo:"resize2fs"为ext文件系统的刷新命令

xfs_growfs /dev/myvol/vo:"xfs_growfs"为xfs文件系统的刷新命令

注意事项

看清题目要求是扩容ext类型的文件系统还是xfs类型的文件系统，ext类型就用$resize2fs$刷新，xfs类型就用xfs_growfs

---

四、添加交换分区(SWAP虚拟内存)

向 serverb 添加一个额外的交换分区756M。交换分区应在系统启 动时自动挂载。不要删除或改动系统上的任何现有交换分区

```
使用dd if of创建虚拟内存(个人感觉用这个做比较容易且方便)
# dd if=/dev/zero of=/root/swap.jpg bs=756 count=1M
制作swap
# mkswap /root/swap.jpg
写入挂载
# vi /etc/fstab
/root/swap.jpg   swap    swap    defaults        0 0
挂载swap
# swapon -a
# swapon -s
查看是否创建成功
# free -h
或者使用磁盘分区分出一个swap
# fdisk /dev/vdb
# mkswap /dev/vdb2
记住分出来的UUID，记不住可以用blkid查看
# blkid
UUID=*
写入挂载
# vim /etc/fstab
UUID=* swap swap defaults 0 0
挂载
# swapon -a
# swapon -s
检查
# free -h
```

命令说明

dd:用指定大小的块拷贝一个文件，并在拷贝的同时进行指定的转换。

if=文件名：输入文件名，缺省为标准输入。即指定源文件。

of=文件名：输出文件名，缺省为标准输出。即指定目的文件。bs=bytes：同时设置读入/输出的块大小为bytes个字节。

count=blocks：仅拷贝blocks个块，块大小等于ibs指定的字节数。

dd if=/dev/zero of=/root/swap.jpg bs=756 count=1M表示从一个空设备"/dev/zero"中取出一个756块大小为1M的数据块给到一个指定的文件"/root/swap.jpg"，文件类型名字随便取

swapon -a:激活swap分区(读取/etc/fstab)

swapon -s: 显示简短的装置讯息

/etc/fstab里面可写UUID也可以写要挂载的文件系统路径，考试不分那种写法，但是用磁盘分区做的文件系统从安全角度来说还是UUID比较好

五、创建VDO卷

建议使用VDO创建，这个比较好理解

根据以下要求，创建新的VDO卷: 使用未分区的磁盘 该卷的名称为 vdough 该卷的逻辑大小是 50G 该卷使用 xfs 文件系统格式化 该卷在系统启动时挂载到/vbread

```
安装VDO工具
# yum -y install vdo kmod-kvdo
# systemctl enable vdo
# systemctl restart vdo
创建VDO虚拟磁盘
# vdo create --name=vdough --device=/dev/vdc --vdoLogicalSize=50G
# mkfs.xfs /dev/mapper/vdough
查看udev事件队列
# udevadm settle
# blkid /dev/mapper/vdough
# vim /etc/fstab
/dev/mapper/vdough /vbread xfs _netdev 0 0
创建挂载的路径
# mkdir /vbread
# mount -a
# df -Th
# reboot
或使用stratis卷管理文件系统
# yum -y install stratisd stratis-cli
# systemctl restart stratisd
# systemctl enable stratisd
使用stratis创建池
# stratis pool create vdough /dev/vdc
查看创建的池是否成功
# stratis blockdev list vdough
前面是池的名字，后面是卷的名字
# stratis filesystem create vdough vdough
查看创建的卷是否成功
# stratis filesystem list
查看UUID
# lsblk --output=uuid /stratis/vdough/vdough
UUID*
# mkdir /vbread
# vim /etc/fstab
UUID=* /vbread xfs defaults,x-systemd.requires=stratisd.service 0 0
# mount -a
# df -Th
# reboot
```

命令说明

vdo create --name=vdough --device=/dev/vdc --vdoLogicalSize=50G

VDO:创建虚拟磁盘

- --name:后面跟vdo卷的名称，随便写
- --device:后面跟真实的物理磁盘
- --vdoLogicalSize:后面跟vdo卷的容量，这里按真实物理空间的1.5倍

udevadm settle:可以用来监视和控制udev运行时的行为，请求内核事件，管理事件队列，以及提供简单的调试机制;"settle" 查看udev事件队列，如果所有的events已处理则退出

stratis卷管理文件系统

- stratis pool create vdough /dev/vdc:从设备"/dev/vdc"中创建一个名为vdough的"池"
- stratis blockdev list vdough:查看创建的池使用的物理设备是否符合
- stratis filesystem create vdough vdough:创建卷
- stratis filesystem list:查看创建的卷是否符合

注意事项

虚拟磁盘写入挂载文件的方式和普通磁盘的有点不同，需添加一条 ”$defaults,x-systemd.requires=$“等号后面跟的是挂载使用的服务,如 ”$x-systemd.requires=vdo.service$“、 ”$default$

六、创建逻辑卷

根据如下要求，创建新的逻辑卷： 逻辑卷的名字为qa, 卷组为 qagroup, 大小是60个PE qagroup 的 PE 大小是 16M 格式化成 ext3 文件系统，系统启动时自动挂载到/ mnt/qa

```
第一步创建一个基本分区
# fdisk -l /dev/vdb
# fdisk /dev/vdb
n
```

```
p
回车
回车
回车
w
第二步创建pv物理卷
# pvcreate /dev/vdb3
第三部创建vg逻辑卷组
# vgcreate qagroup /dev/vdb3 -s 16M
第四部创建lv逻辑卷
# lvcreate -l 60 -n qa qagroup
或
# lvcreate -L 240M -n qa qagroup
制作文件系统
# mkfs.ext3 /dev/qagroup/qa
挂载路径
# mkdir /mnt/qa
# vim /etc/fstab
/dev/qagroup/qa /mnt/qa ext3 defaults 0 0
# mount -a
```

命令说明

pvcreate:创建物理卷,处于LVM最底层，可以是物理硬盘或者分区。

vgcreate qagroup/dev/vdb3 -s 16M:创建逻辑卷组,"qagroup":卷组名，根据题目要求改;"/dev/vdb3":之前划分好的物理卷;"-s 16M":指的是在分区的时候指定vg的大小。

lvcreate -l 60 -n qa qagroup:创建逻辑卷,"-l 10":指定PE;默认情况一块PE为4M,目前题中指定为16M;"-n qa":指定逻辑卷的名字,一般"-n"后面跟逻辑卷名，名按题目要求改动;"qagroup":之前创建出来的逻辑卷组。

七、系统优化设置

为系统选择建议的 tuned 配置集，并将它设为默认设

```
安装工具
# yum -y install tuned
重启并设置开机自启
# systemctl restart tuned --now
查看可使用的配置文件
# tuned-adm list
设置profile为推荐值
# tuned-adm profile virtual-guest
设置确认
# tuned-adm active
```

RHCE

从练习二开始看

RHCE ansible+playboot剧本基本上没什么可以注意的，唯一注意的点就只有playbook剧本里的语法、一些变量和模块拼写的错误，只要用心去写这些都是可以避免的。

刚开始学，容易出现一些普遍的错误，尽量自己学着处理这些报错，实在不行就上群问问。

建议练习再写playbook剧本的时候一步一步来，写完一个任务或一个模块执行一次，这样报错了还比较容易去排查。考试的时候觉得不稳也可以这样子。

文档中的剧本别随便复制，可能格式会有误。(太多了，懒得整理格式)

一、安装及配置 ansible

在 control 上安装并配置 ansible, 要求如下： 安装相应的软件包 创建一个静态 inventory 到/home/greg/ansible/inventory, 清单包含： node1 属于 dev 组 node2 属于 test 组 node3 和 node4 属于 prod 组 node5 属于 balancers 主机组 prod 组属于 webservers 主机组 创建一个/home/greg/ansible/ansible.cfg 使用/home/greg/ansible/inventory 角色目录为/home/greg/ansible/roles

```
# sudo yum -y install ansible
# mkdir -p ansible/roles
配置ansible需要管理的主机
# vim /home/greg/ansible/inventory
[dev]
node1
[test]
node2
[prod]
node3
node4
[balancers]
node5
[webservers:children]#组继承
prod
# cp /etc/ansible/ansible.cfg .
修改当前用户的配置文件
# vim ansible/ansible.cfg
inventory = /home/greg/ansible/inventory
roles_path = /home/greg/ansible/roles
取消ssh验证，也就是第一次ssh机器时不需要按yes更新密钥
host_key_checking = False
remote_user = greg
[privilege_escalation]
become=True
become_method=sudo
become_user=root
become_ask_pass=False
或者通过root用户远程管理
# vim ansible/ansible.cfg
inventory = /home/greg/ansible/inventory
roles_path = /home/greg/ansible/roles
host_key_checking = False
remote_user = root
# vim /home/greg/ansible/inventory
...........
最下面写上一个ansible调用ssh密码变量
[all:vars]
ansible_password=flectrag
# ansible dev --list-hosts
# ansible all --list-hosts
```

二、创建并运行 ansible ad-hoc 命令

创建 ad-hoc 脚本 /home/student/ansible/adhoc.sh，为所有节点配置 yum 仓库： 仓库 1： 仓库名： repo_base 描述为： repo_base Base URL：http://foundation0.ilt.example.com/dvd/BaseOS 开启 GPG 签名验证，密钥为：http://foundation0.ilt.example.com/dvd/RPM-GPGKEY-redhat-release 仓库的状态是 enabled 仓库 2： 仓库名： repo_stream 描述为： repo_stream Base URL：http://foundation0.ilt.example.com/dvd/AppStream 开启 GPG 签名验证，密钥为：http://foundation0.ilt.example.com/dvd/RPM-GPGKEY-redhat-release 仓库的状态是 enabled

```
# ansible all -m ping -o
# ansible-doc yum_repository
# Shift + G
# vim adhoc.sh
#!/bin/bash
ansible all -m yum_repository -a "file=repo_base name='repo_base' description=repo_base
baseurl='http://foundation0.ilt.example.com/dvd/BaseOS' gpgcheck=yes gpgkey='http://foundation0.ilt.example.com/dvd/RPM-GPGKEY-
redhat-release' enabled=yes"
ansible all -m yum_repository -a "file=repo_stream name='repo_stream' description=repo_stream
baseurl='http://foundation0.ilt.example.com/dvd/AppStream'
gpgcheck=yes gpgkey='http://foundation0.ilt.example.com/dvd/RPM-GPGKEY-redhat-release' enabled=yes"
# chmod +x adhoc.sh
# ./adhoc.sh
# ansible all -m shell -a "yum repolist"
```

三、安装软件包

创建/home/student/ansible/install_packages.yml 的 playbook： 在 dev、test、prod 组中安装 php 和 mariadb 软件包 在 dev 组中安装 Development Tools 包组 在 dev 组中的主机升级所有的软件包

```
# vim install_packages.yml
---
- name: install php and mariadb
 hosts:
 - dev
 - test
 - prod
 tasks:
 - name: install packages
 yum:
 name:
 - php
 - mariadb
 state: present
- name: install development tools group
 hosts: dev
 tasks:
 - name: install dev tools
 yum:
 name: "@Development Tools"
 state: present
 - name: update all
 yum:
 name: '*'
 state: latest
# rpm -qa php mariadb
# yum grouplist
```

四、使用 RHEL system role

安装 RHEL system roles 软件包，并创建一个 playbook 名字为 timesync.yml 在所有受管主机中运行 使用 timesync 角色 配置角色使用的时间服务器是 172.25.254.254 配置时间同步的方式是 iburst

```
# ansible all -m ping -o
# sudo yum -y install rhel-system-roles
# vim ansible.cfg
roles_path = /home/greg/ansible/roles:/usr/share/ansible/roles
# ansible-galaxy list
# cp -r /usr/share/doc/rhel-system-roles/timesync/example-timesync-playbook.yml /home/greg/ansible/timesync.yml
修改主机
#vim /home/greg/ansible/timesync.yml
---
- hosts: all
  vars:
    timesync_ntp_servers:
      - hostname: 172.25.254.254
        iburst: yes
  relos:
    - rhel-system-roles.timesync

修改服务器主机
# ansible-playbook timesync.ym
```

五、使用 Ansible Galaxy 安装 roles

使用名为 requirements.yml 的文件下载并安装 Ansible Galaxy 角色到 /home/greg/ansible/roles 目录中，可以从下列 URL 下载：http://materials.example.com/haproxy.tar 这个角色的名字叫 balance http://materials.example.com/phpinfo.tar 这个角色的名字叫 phpinfo

```
# vim roles/requirements.yml
---
- src: http://materials.example.com/haproxy.tar
 name: balance
- src: http://materials.example.com/phpinfo.tar
 name: phpinfo
# ansible-galaxy install -r /home/greg/ansible/roles/requirements.yml -p /home/greg/ansible/roles
# ansible-galaxy list
```

六、创建并使用 role

在/home/greg/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，并自动运行 启用防火墙，开放 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

```
防火墙也需要自启动
# cd /home/greg/ansible/roles
# ansible-galaxy init apache
# cd ..
# vim roles/apache/tasks/main.yml
---
- name: install apache
 yum:
 name: httpd
 state: latest
- name: start service apache
 service:
 name: httpd
 state: started
 enabled: yes
- name: start service firewalld
 service:
 name: firewalld
 state: started
 enabled: yes
- name: open firewalled port
 firewalld:
 service: http
 permanent: yes
 state: enabled
 immediate: yes
- name: template a file
 template:
 src: index.html.j2
 dest: /var/www/html/index.html
# vim roles/apache/templates/index.html.j2
Welcome to {{ ansible_fqdn }} on {{ ansible_default_ipv4.address }}
```

七、使用来自 Ansible Galaxy 的 roles

做此题之前，先停一下 node5 主机上的 httpd 服务，但会导致node1-4 重启时无法启动到登录界 面，不过不 影响。停止方法： sudo systemctl stop httpd 创建一个 playbook 名字叫/home/greg/ansible/roles.yml, 要求如下： 这个 playbook 运行在 balancers 这个主机组上，并使用 balance 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://node5.lab.example.com 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 node2.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：node5.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://node3.lab.example.com/hello.php 看到如下输出： Hello PHP World from node3.lab.example.com 类似的，浏览器访问 URL：http://node4.lab.example.com/hello.php 看到如下输出： Hello PHP World from node4.lab.example.com

```
bastion ( role: balancer ) LB
serverc,serverb ( role: apache,phpinfo )
先使用phpinfo、apache角色，再使用balance角色
# vim roles.yml
---
- name: use apache and php
 hosts: webservers
 roles:
 - apache
 - phpinfo
- name: use role balancer
 hosts: balancers
 roles:
 - balance
# ansible-playbook roles.yml
```

八、创建并使用逻辑卷

创建一个 playbook 名字叫/home/greg/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 lv01 LV 的大小是 1500M 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 logical vol of that size 则创建分区大小变成 800M 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

练习环境里的vg逻辑卷组需要自己创建出来，考试是不用的

```
# vim lv.yml
---
- name: create lv
 hosts: all
 tasks:
 - block:
 - name: create lv01 of 1500M
 lvol:
 vg: vg0
 lv: lv01
 size: 1500
 - name: ext4 filesystem
 filesystem:
 fstype: ext4
 dev: /dev/vg0/lv01
 rescue:
 - debug:
 msg: could not create logical vol of that size
 - name: create lv01 of 800M
 lvol:
 vg: vg0
 lv: lv01
 size: 800
 when: ansible_lvm.vgs.vg0 is defined
 - name: ext4 filesystem
 filesystem:
 fstype: ext4
 dev: /dev/vg0/lv01
 when: ansible_lvm.vgs.vg0 is defined
 ignore_errors: yes
 - debug:
 msg: volume group does not exist
 when: ansible_lvm.vgs.vg0 is undefined
# ansible-playbook lv.yml
```

九、生成一个hosts 文件

在 /home/greg/ansible/ 目录下编辑一个hosts.j2 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/greg/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 node1.lab.example.com servera 172.25.250.11 node2.lab.example.com serverb 172.25.250.12 node3.lab.example.com serverc 172.25.250.13 node4.lab.example.com serverd 172.25.250.14 node5.lab.example.com bastion 注意：生成的文件顺序不对没有关系。

```
# vim hosts.j2
*
*
{% for host in groups['all'] %}
{{ hostvars[host]['ansible_facts']['default_ipv4']['address'] }} {{ hostvars[host]['ansible_facts']['fqdn'] }}
{{ hostvars[host]['ansible_facts']['hostname'] }}
{% endfor %}
# vim hosts.yml
---
- name: create myhosts
 hosts: all
 tasks:
 - name: template a file /etc/myhosts
 template:
 src: /home/greg/ansible/hosts.j2
 dest: /etc/myhosts
 #when: '"dev" in group_names'
 when: 'inventory_hostname in groups.dev'
# ansible-playbook hosts.yml
```

十、修改文件内容

创建一个 playbook 名为/home/greg/ansible/issue.yml，修改/etc/issue 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

```
# vim issue.yml
---
- name: replace issue
 hosts: all
 tasks:
 - name: replace Development
 copy:
 content: "Development\n"
 dest: /etc/issue
 when: inventory_hostname in groups.dev

 - name: replace Test
 copy:
 content: "Test\n"
 dest: /etc/issue
 when: inventory_hostname in groups.test
 - name: replace Production
 copy:
 content: "Production\n"
 dest: /etc/issue
 when: inventory_hostname in groups.prod
# ansible dev -m shell -a "cat /etc/issue"
# ansible test -m shell -a "cat /etc/issue"
# ansible prod -m shell -a "cat /etc/issue"
```

十一、 创建一个 WEB 内容目录

创建一个 playbook 名字叫/home/greg/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 2775 软链接/var/www/html/webdev 到/webdev 在/webdev 中创建 index.html, 内容为: Development 通过 node1.lab.example.com/webdev/ 能访问到内容

```
# vim webcontent.yml
---
- name: create web directory
 hosts: dev
 roles:
 - apache
 tasks:
 - name: create group
 group:
 name: webdev
 state: present
 - name: create directory webdev
 file:
 state: directory
 path: /webdev
 mode: '2775'
 group: webdev
 - name: create a symbolic link
 file:
 state: link
 src: /webdev
 dest: /var/www/html/webdev
 - name: create web content
 copy:
 content: "Development\n"
 dest: /webdev/index.html
 setype: httpd_sys_content_t
# ls -lhd /webdev
# ls -lhZ /webdev/index.html
# ls -lh /var/www/html/webdev
# curl http://node1.lab.example.com/webdev/
```

十二、创建硬件报告

创建一个 playbook 名字叫/home/greg/ansible/hwreport.yml, 在所有的受管主机上 创建/root/hwreport.txt, 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, 并 修 改 成 为 hwreport.txt 如果硬件不存在，则对应的值为：NONE

```
# vim hwreport.yml
---

- name: hw report
```

```
 hosts: all
 vars:
 hw_all:
 - hw_name: HOST
 hw_cont: "{{ inventory_hostname | default('NONE', true) }}"
 - hw_name: MEMORY
 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: download hw_temp
 get_url:
 url: "http://materials.example.com/hwreport.empty"
 dest: /root/hwreport.txt
 - name: generate hw report
 lineinfile:
 path: /root/hwreport.txt
 regexp: "^{{ item.hw_name }}="
 line: "{{ item.hw_name }}={{ item.hw_cont }}"
 loop: "{{ hw_all }}"
# ansible-playbook hwreport.yml
```

十三、创建一个包含密码的vault 文件

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

```
# vim ansible.cfg
vault_password_file = /home/greg/ansible/secret.txt
# vim locker.yml
---
pw_developer: Imadev
pw_manager: lmamgr
# vim secret.txt
redhat
# ansible-vault encrypt /home/greg/ansible/locker.yml
```

十四、创建用户账户

使用 http://materials.example.com/user_list .yml 文件和/home/greg/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: developer 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

```
# wget http://materials.example.com/user_list .yml
developer ----→ groups.dev or groups.test ------ locker.yml(pw_developer) --- devops
sally (job manager) ------ groups.prod ------ locker.yml(pw_manager) ------ opsmgr
sha512
# vim users.yml
---
- name: create users
 hosts: all
 vars_files:
 - /home/greg/ansible/locker.yml
 - /home/greg/ansible/user_list.yml
 tasks:
 - name: create group devops
 group:
 name: devops
 state: present
 when: inventory_hostname in groups.dev or inventory_hostname in groups.test

 - name: create group opsmgr
 group:
 name: opsmgr
 state: present
 when: inventory_hostname in groups.prod

 - name: create user developer
```

```
 user:
 name: "{{ item.name }}"
 groups: devops
 password: "{{ pw_developer | password_hash('sha512', 'mysecretsalt') }}"
 comment: "{{ item.job }}"
 loop: "{{ users }}"
 when: item.job == 'developer' and ( inventory_hostname in groups.dev or inventory_hostname in groups.test )
 - name: create user manager
 user:
 name: "{{ item.name }}"
 groups: opsmgr
 password: "{{ pw_manager | password_hash('sha512', 'mysecretsalt') }}"
 comment: "{{ item.job }}"
 loop: "{{ users }}"
 when: item.job == 'manager' and inventory_hostname in groups.prod
# ansible_playbook users.yml
```

十五、 更新 Ansible 库的密钥

按照下方所述，更新现有 Ansible 库的密钥 从 http://xxx.example.com/materials/salaries.yml 下载 Ansible 库到 /home/greg/ansible 当前库的密码为: mysecret 新的库密码为: yournewsecret 库使用新密码保持加密状态

salaries.yml文件练习里是没有的，需要在群里下载出来，考试时这个文件是有的

突然发现群里的这个salaries.yml文件解密密码是错的，所以只能自己随便建一个出来练习加密解密

```
创建一个有数据的文件
#echo 'the is encrypt' > salaries.txt
创建两个存放密码的密码谱
# echo 'mysecret' > secret_current.txt
# echo 'yournewsecret' > secret_new.txt
编辑ansible.cfg配置文件，把vault_password_file存在的行注释掉
# vim ansible.cfg
使用 secret_new.txt密码谱加密
# ansible-vault encrypt salaries.txt --vault-password-file=/home/greg/ansible/secret_new.txt
解密
# ansible-vault decrypt salaries.txt --vault-password-file=/home/greg/ansible/secret_new.txt
使用secret_current.txt密码谱加密
# ansible-vault encrypt salaries.txt --vault-password-file=/home/student/ansible/secrect_current.txt
# wget http://materials.example.com/salaries.yml
# echo 'mysecret' > secret_current.txt
# echo 'yournewsecret' > secret_new.txt
用当前库密码解密
# ansible-vault decrypt salaries.yml --vault-password-file=/home/student/ansible/secrect_current.txt
用新库密码加密
# ansible-vault encrypt salaries.yml --vault-password-file=/home/student/ansible/secret_new.txt
# vim ansible.cfg
# 取消注释
vault_password_file = /home/greg/ansible/secrect.txt
```

十六、 安装RHEL角色

安装RHEL角色，并使用SeLinux角色，要求在所有节点运行，将SELINUX设置为强制模式

```
安装角色
# yum install rhel-system-roles -y
# cp -rf /usr/share/ansible/roles/linux-system-roles.selinux ./
安装出来的可以改也可以不改，剧本里面只留那么多，别的多余删掉
# vim selinux.yml
---
- hosts: all
  vars:
    selinux_policy: targeted
    selinux_state: enforcing
  roles:
    - role: rhel-system-roles.selinux
 tasks:
    - name: apply SElinux role
      block:
        - include_role:
            name: rhel-system-roles.selinux

      rescue:
```

```
        - name: check
          fail:
          when: not selinux_reboot_required
        - name: reboot
          shell: reboot
        - name: changed
          include_role:
            name: rhel-system-roles.selinux
```

十七、 创建到期用户账户

创建用户账户，账户jack，新增设置密码有效期为30天。账户jony，新增设置相应的ID1111，用户有效期到2022-01-20

```
# date -d 2022-01-20 +%s #获取对应日期的unix时间戳
1642636800
---
- hosts: all
  vars:
    - users:
        - name: jack
        - name: jony
  tasks:
    - name: create user
      user:
        name: "{{ item.name }}"
        password: "{{ 'redhat'| password_hash( 'sha512' ) }}"
      loop: "{{ users }}"
    - name: Set user validity period
      user:
        name: "{{ item.name }}"
        expires: 1642636800
      loop: "{{ users }}"
      when: item.name == 'jack'
#源设置密码到期参数password_expire_max也被移除，所以只能用shell模块
#    - name: Password expired
#      user:
#       password_expire_max: 30
    - name: Password expired
      shell: chage -M 30 "{{ item.name }}"
      loop: "{{ users }}"
      when: item.name == 'jony'
```

十八、 使用crontab模块

用户jack每三个月的每周日晚上22点39分查看一次自身用户登录情况

```
# vi crontab.yml
---
- hosts: all
  tasks:
    - name: Creates a cron file under /etc/cron.d
      cron:
        name: Login time
        minute: "39"
        hour: "22"
#        day: ""
        month: "*/3"
        weekday: "0"
        user: jack
        job: "(last && lastb)|grep jack"
```

十九、 创建新的磁盘分区

在balancers主机上，划分新的partition，/dev/vdd，编号1，大小1500m，格式化成ext4，mount到/newpart1目录，如果空间不够，分800m，如果没有vdd，报错

练习环境可能需要自己添加一个磁盘

```
# vim partition.yml
---
- name: partition
  hosts: balancers
  tasks:
    - name: Create a directory
      file:
        name: /newpart
        state: directory
    - block:
      - name: device 1500M
        parted:
          device: /dev/vdb
          number: 1
          state: present
          part_end: 1500MiB
      - name: ext4 filesystem
        filesystem:
          fstype: ext4
          dev: /dev/vdb1
      - name: mount
        mount:
          path: /newpart
          src: /dev/vdb1
          fstype: ext4
          state: mounted
      rescue:
      - debug:
          msh: Could not create partation of that size
      - name: device 800M
        parted:
          device: /dev/vdb
          number: 1
          state: present
          part_end: 800MiB
        when: ansible_facts.devices.vdb is defined
      - name: ext4 filesystem
        filesystem:
          fstype: ext4
          dev: /dev/vdb1
        when: ansible_facts.devices.vdb is defined
      - name: mount
        mount:
          path: /newpart
          src: /dev/vdb1
          fstype: ext4
          state: mounted
        when: ansible_facts.devices.vdb is defined
#        ignore_errors: yes
      - debug:
          msg: Disk does not exist
        when: ansible_facts.devices.vdb is undefined
```

结　束

最后说几句，练习文档是死的，方法是活的，尽量多看视频练习，写出适合自己做题的方法，别依赖这篇文档。如果文档中有什么不清楚、错误或者遗漏的群里可以找我，随时可以更改。

最后祝福所有学员都能顺顺利利的通过考试拿高薪。