Red Hat Services Management and Automation

> - Service / RHCE7
>   - man
>   - --help
> - Ansible / RHCE8
>   - https://docs.ansible.com/ansible/2.8/index.html#
>   - keywords
>   - name
>   - hosts

        - tasks
        - vars
        - loop(for)
    - Module Index
  - ansible-doc

```
$ ansible-doc -l | grep KEYWORD
$ ansible-doc MODULE-NAME
```

本课程基于

- 红帽 Ansible 引擎 2.9
- 红帽企业 Linux 8.1

[kiosk@foundation]

所有的lab脚本存储位置

```
$ ls /content/courses/rh358/rhel8.1/grading-scripts/
```

1 管理网络服务

network+NetworkManager <=7
NetworkManager(**nmcli**) >=8

- Systemd回顾
- NetworkManager回顾
- 自动化配置服务和网络接口

```
-RHEL>=7 /pid=1=systemd
# systemctl list-unit-files | grep KEYWORD
# systemctl status KEYWORD

* # systemctl enable --now DAEMON

# systemctl enable DAEMON
# systemctl start DAEMON
* # systemctl restart DAEMON  # <=- run + vim/conf
# systemctl status DAEMON

# systemctl get-default
# systemctl -t target
-CLI  RHEL>=7                          RHEL<6
# systemctl isolate multi-user.target   # init 3
-GUI
# systemctl isolate graphical.target    # init 5
```

```
# systemctl set-default multi-user.target

# systemctl list-dependencies graphical.target  | grep target

# systemctl status crond.service
# systemctl stop crond.service
# systemctl mask crond.service
# systemctl is-active crond.service
# systemctl start crond || echo 无法启动
# systemctl unmask crond.service
# systemctl start crond && echo 可以启动
# systemctl is-active crond.service
```

Controlling Network Services

```
USERCTL=yes|no
```

> - **doc**
>
>   # grep -r BOOPRO /usr/share
>   *# vim /usr/share/doc/initscripts/sysconfig.txt
> - **man**
>   # man -k ifcfg
>   *# man nm-settings-ifcfg-rh

[student@workstation ~]

```
$ lab servicemgmt-netservice start
```

[root@workstation]

```
# systemctl status chronyd
# systemctl restart chronyd
# systemctl status chronyd
```

[root@servera]

```
# systemctl status chronyd
# systemctl start chronyd
# systemctl status chronyd

# systemctl is-enabled chronyd
# reboot

# systemctl is-active chronyd
```

[student@workstation ~]

```
$ lab servicemgmt-netservice finish
```

Configuring Network Interfaces

> ```
> $ MANWIDTH=120 man nmcli | grep nmcli.*add
> ```

[student@workstation ~]

```
$ lab servicemgmt-netreview start
```

[root@servera]

```
# ip link

# nmcli con show

# nmcli con add con-name eth1 \
type ethernet \
ifname eth1

# nmcli con show

# nmcli con mod eth1 ipv4.addresses 192.168.0.1/24 \
ipv4.method manual
# cat /etc/sysconfig/network-scripts/ifcfg-eth1

# nmcli con up eth1
# ip addr show dev eth1

# ping -c 2 192.168.0.1
# echo $?

# ip route

# cat /etc/resolv.conf
```

[student@workstation ~]

```
$ lab servicemgmt-netreview finish
```

指导练习: 自动化配置服务和网络接口

> - roles:
>   - rhel-system-roles.`network` # 建议
> - -m `shell` -a 'nmcli ...'
> - -m `nmcli` -a '...' # 不建议，需要额外安装相应依赖包pkg

[student@workstation]

```
$ lab servicemgmt-automation start
```

```
$ ansible-playbook configNet.yml
```

[student@workstation]

```
$ lab servicemgmt-automation finish
```

2 配置网络聚合 配置网络Team

> - 同一个服务，多个网卡，多个IP
> - 多个网卡，对应同一个IP
>   - 带宽增加
>   - 冗余

> - 管理网络Team
> - 自动化网络Team

> 带外管理，vnc

> VIP - [eth0 + eth1]
>
> RHEL<=6 `bond`, RHEL>=7 `team`，RHEL9 `bond`

```
# man -k nmcli
# man nmcli-examples | grep -A 2 nmcli.*team
  $ nmcli con add type team con-name `Team1` ifname `Team1` config `team1-master-
json.conf`
  $ nmcli con add type ethernet con-name Team1-slave1 ifname `em1` master `Team1`
  $ nmcli con add type ethernet con-name Team1-slave2 ifname `em2` master `Team1`

# man -k team
# man teamd.conf | grep backup
  "runner": {"name": "activebackup"},
```

```
-RHEL=7
# nmcli con add \
type team \
con-name Team1 \
ifname Team1 \
config '"runner": {"name": "activebackup"}'

-RHEL=8
[root@serverb ~] privbr2/eth1+privbr2/eth2
# nmcli con add type team \
    con-name Team1 \
    ifname Team1 \
    team.runner activebackup

# nmcli con add type ethernet \
    con-name Team1-slave1 ifname eth1 master Team1
# nmcli con add type ethernet \
    con-name Team1-slave2 ifname eth2 master Team1
```

```
# nmcli dev status
# ls /etc/sysconfig/network-scripts/ifcfg-*

# nmcli connection up Team1-slave1
# nmcli connection up Team1-slave2

# nmcli dev status

# cat -n /etc/sysconfig/network-scripts/ifcfg-Team1

# man nmcli | grep nmcli.*mod
# nmcli connection \
    mod Team1 ipv4.method manual ipv4.addresses 10.1.1.2/8
# nmcli connection up Team1
# ip a s | grep -w inet
# ping -c 4 10.1.1.1

# teamdctl Team1 state
# nmcli connection down Team1-slave1
# teamdctl Team1 state
```

```
# ping -c 4 10.1.1.1
```

[root@servera ~] eth2

```
# nmcli connection delete Wired\ connection\ 2

# nmcli con add type ethernet ifname eth2 connection.id con-eth2 connection.autoconnect
true ipv4.method manual ipv4.addresses 10.1.1.1/8
Connection 'con-eth2' (292e1612-7d87-4677-bc43-57bd0be09d41) successfully added.

# nmcli con up con-eth2
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/33)
```

3 管理DNS和DNS服务器

> - /etc/nsswitch.conf，定义解析顺序（hosts: files/hosts dns/resolv.conf）
>   - /etc/hosts，本地解析
>   - /etc/resolv.conf，通过DNS服务解析

> - 描述DNS服务
> - 使用Unbound配置缓存名称服务器
> - 排故DNS问题
>   - **host** www.mi.com,
>   host 172.25.254.250 172.25.254.254
>   - nslookup www.mi.com,
>   nslookup 172.25.254.250 172.25.254.254
>   - dig www.mi.com ,
>   dig -x 172.25.254.250 @172.25.254.254
> - 🚩 使用BIND 9配置权威名称服务器
> - 自动化配置DNS

ID	TYPE	主配置文件*2	区域文件（正向|反向）*2	operation	PKG
1	**Master**	Y	Y	edit	bind
2	Slave	Y	N	copy	bind
3	Cache	Y	N	-	unbound
4	Forward	Y	N	-	bind

> ip, 8.8.8.8, 114.114.114.114

FQDN　www.mi.com.

$ hostname -s `Enter`　　www

`.` 根域

`com.` 类别

`mi.com.`　　一级域名

`mail.mi.com.`　二级域名

```
$ grep ^hosts /etc/nsswitch.conf
hosts:      files dns myhostname

- files = hosts
- dns = resolve.conf
```

- hosts
  - Linux, MacOS - /etc/hosts
  - Windows - C:\windows\system32\drivers\etc\hosts
- dns
  - permanent
    - $ nmcli con mod CN ipv4.dns 8.8.8.8
  - active
    - $ cat /etc/resolve.conf

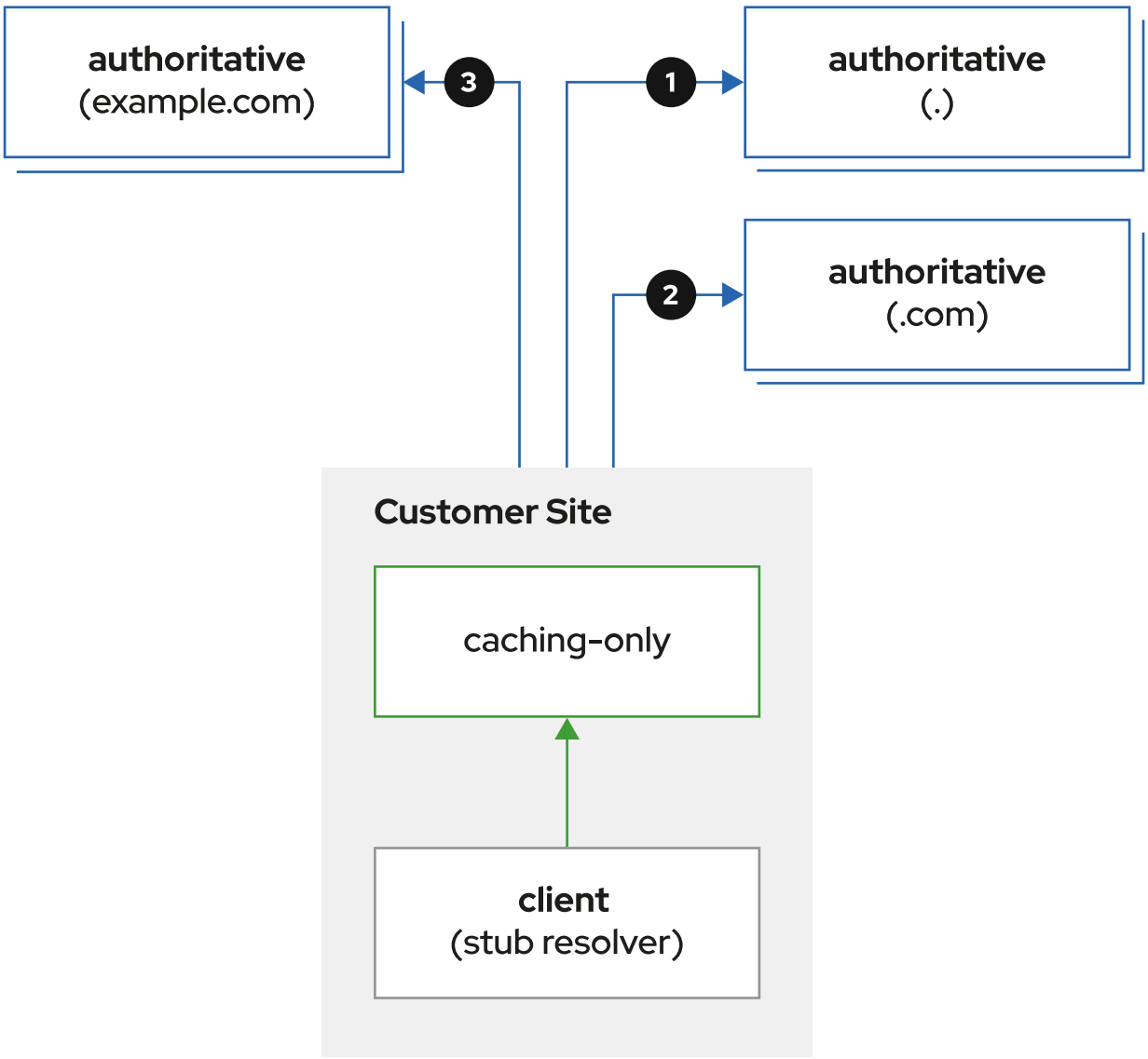
1️⃣ domain/conf_file***1*** *permission*2, file "example.com.localhost"; ,file "25.172.loopback";

2️⃣ zone/zone_file***2***　正向`lab.example.com.`；　反向`25.172.in-addr.arpa.`

3️⃣ recorder

hostname　`A` ip　　Address，主机名解析成IP地址；
反转HID `PTR` hostname　把向指针，IP 地址解析成主机名

`SOA` = `NS` = servera.lab.example.com==.==



Client looking up www.example.com, no records cached.

→ iterative lookups (assuming not cached) → recursive lookups

```
windows dns本地存在缓存，可以全用下面的命令清除
X:\> ipconfig /flushdns
```

CLASS		NID	HID	zone	PTR
A	**10**.1.2.3/8	10	1.2.3	10.in-addr.arpa.	3.2.1
B	**172.25**.254.9/16	172.25	254.9	25.172.in-addr.arpa.	9.254
C	**192.168.9**.10/24	192.168.9	10	9.168.192.in-addr.arpa.	10

> DNS解析
>
> - 递归查询，一级一级查找

- 迭代查询，并行

4 管理DHCP和IP地址分配

- 🚩 使用DHCP配置IPv4地址分配
- 配置IPv6地址分配
- 自动化配置DHCP

场景：无线路由、PXE、网络Ghost

作用：分配网络参数，除了mac（花钱IEEE）

工作原理：客户端广播，从先应答的服务器获得IP

协议：67/UDP, 68/UDP `/etc/services`

5 管理打印机和打印文件

- 配置和管理打印机
- 🚩 自动化配置打印机

client = windows, mocos

ID		
1	**printer + nic**	network printer server
2	os/router + printer + share	printer server
2	os/windows + printer + share	printer server
2	os/mac + printer + share	printer server
2	os/linux + printer + share	printer server

```
-631
# grep Listen /etc/cups/cupsd.conf
Listen localhost:631
Listen /var/run/cups/cups.sock

-ipp
# grep -w 631 /etc/services
ipp             631/tcp                         # Internet Printing Protocol
ipp             631/udp                         # Internet Printing Protocol

-lp...
# man -k cup
```

6 配置邮件传输

- 配置一个仅发送邮件服务器
- 🚩 自动配置Postfix

	Windows	Linux	
MTA Mail Transport Agent	Microssoft/Exchange, IBM/Domino	RHEL8 **postfix** RHEL<=7 sendmail qmail IBM/Domino	邮局
MDA mail delivery agents			邮递员
MUA mail user agent	outlook, foxmail	GUI: evolution, thunderbird CLI: **mail**, **mutt**	客户

	PROTOCOL	PORT	SSL	Package	
发送	smtp	25	urd 465	postfix	
接收	imap	143	imaps 993	dovecot	同步sync
接收	pop3	110	pop3 995	dovecot	拷贝copy

7 配置MariaDB SQL数据库

- 🚩 安装MariaDB数据库
- 🚩 MariaDB中SQL管理
- 🚩 MariaDB用户和访问权限
- 备份和🚩 恢复MariaDB
- 自动化部署MariaDB

mysql -=> mariadb

- 关系型数据库(有关系的表格)　　oracle, oracle-mysql, mariadb, db2, sql-server
- 非关系型数据库(Key=value)　　redis, memcache

SQL	COMMENT	CMD（help CMD;）
DDL	数据**定义**语言（结构）	**create**, alter, drop, **show**, use, DESCRIBE
DML	数据**操纵**语言（内容）	**select**, insert, update, delete
DCL	数据**控制**语言（权限）	**grant**, revoke

8 配置Web服务器

> - 使用Apache HTTPD配置一个基本Web服务器
> - 使用Apache HTTPD配置和排故虚拟主机
> - 配置Apache HTTPD HTTPS
> - 使用Nginx配置一个Web服务器
> - 自动化配置Web服务器

> ⚠️ 当存在虚拟主机时，第一个虚拟主机会覆盖默认的web站点

TYPE	PKG	URL	name | ip | port	
http	httpd	http://www0.lab.example.com	基于 名称 的 虚拟主机	80
vhost	httpd	http://webapp0.lab.example.com	基于 名称 的 虚拟主机	80
https	mod_ssl	https://www0.lab.example.com	基于 端口 的 虚拟主机	443
permission	http_manual	http://www0.lab.example.com/manual	权限	

密文	明文	公钥 == Locker	私钥 == Key
https	http + ssl	/PATH/*.crt	/PATH/*.key
ssh	telnet	~/.ssh/id_rsa.pub	~/.ssh/id_rsa

	优点	缺点	
apache	稳定，组件多	重量	LAMP
nginx	反向代理，轻量		LNMP

> Client Server
>
> https (cert+key)

> (cert + plantext)encrypt + keys == decrept -=> plantext

ID	自签名证书	公共证书
1	通过命令（openssl）直接生成	申请
2	免费（默认1年，可改）	收费，免费（3月）
3	测试	生产
4	无网络	需要 internet

9 调整Web服务器流量

> - 使用Varnish缓存静态内容
> - 使用HAProxy终止HTTPS流量和配置负载均衡
> - 自动化调整Web服务

10 提前基于文件的网络存储

> - 导出NFS文件系统 - Like Linux
> - 提供SMB文件共享 - 跨平台 Windows
> - 自动化提供文件存储

NAME			Windows	Linux	TYPE	
NAS	Network Attached Storage	网络附加存储	samba(SMB)	nfs	dir	远程
SAN	Storage Area Network	存储区域网络	target / iSCSI	同前	block	远程
DAS	Dirctor Attached Storage	直连附加存储				本地

SERVICE	SAMBA	NFS
NAS	directory	directory
OS	windows	Like Unix
Permission	user	ip, network, hostname, domain
conf	/etc/samba/smb.cfg	/etc/exports
sharename	[custom]	/path
Pkg	samba / DAEMON, samba-common / conf, samba-client / smbclient	nfs-utils
fstab/filesystem	cifs (cifs-utils)	nfs
DAEMON	smb/SERVICE, nmb/NAME	**nfs-server, nfs-secure-server** / RHEL>=7, nfs, nfs-secure / RHEL<7
firewall	samba	nfs -=> # mount, [port-mapper/**rpc-bind**, mountd] -=> $ showmount
Client-cmd	smbclient - like ftp interactive	-
autofs	Yes	Yes

	Windows	Linux	QUOTA	PERM
Local	ntfs	ext4, xfs	Y	Y
Remote	cifs	nfs	N	Y

- samba

ID	共享				smbclient
1	共享级	share	win98		
2	**用户级**	user	winxp	Everyone | tom, jerry	-L -N | -U tom%password
3	server级共享	server	ldap		
3	域级共享	domain	AD		

```
Windows 切换身份
m1 X:\> net use * /del
m2 注销
m3 重启
```

> fuse-sshfs

OS	LOCAL: user	LAN: samba user
Windows	tom%ttt	tom%ttt
Linux	tom%-	tommy%tmm

11 访问基于块的网络存储

> - 提供iSCSI存储
> - 访问iSCSI存储
> - 自动化配置iSCSI Initiator

	Client	Server
SAN / block	iscsi	target
CMD	iscsiadm	targetcli / ls
	iscsiadm	block
	lsblk	iscsi - block
	fstab / `_netdev`	iscsi - port
acl	/etc/iscsi/init*	iscsi - acl
True	iscsid	target
False	iscsi	targetd

```
# iscsiadm --mode session -P 3
```

附录

A0. 4步技巧

ID		
1	word	释意
2	Tab	前2-3个字母
3	man	--help
4	echo $?	== 0

A1. 红帽

ID	URL	说明
RH358	红帽服务管理与自动化	课程代码
EX358	红帽认证服务管理和自动化专家考试	考试代码

A2. 软件

1	VMware	虚拟机
2	Typora	Markdown
3	Xmind	思维导图
4	Snipaste	截图

A3. 培训环境

```
$ cat /etc/rht
RHT_COURSE=rh358
RHT_TITLE="Management and Automation of Linux Network Services (RH358)"
RHT_VMS="bastion workstation servera serverb serverc serverd "
RHT_VM0="classroom "
```

虚拟机	主机名	功能	必须	root	User
VMware	foundation	平台	1	Asimov	kiosk%redhat
KVM	classroom	功能服务器	1	Asimov	instructor%Asimov
KVM	bastion	router	1	redhat	student%student
KVM	workstation	GUI	0	redhat	student%student
KVM	server{a..d}	CLI	0	redhat	student%student

[kiosk@foundation] 注意启动顺序

```
$ rht-vmctl start classroom
$ rht-vmctl start bastion
$ rht-vmctl start workstation
$ rht-vmctl start servera
```

```
$ ping -c 4 workstation
$ ssh root@workstation
```

```
$ ls /content/slides/
```

A4. yaml

```
$ ansible-doc -l | grep keyword
$ ansible-doc module-name
/EX

$ ansible-playbook x.yml
```

> - `---`第一行，可省略
> - 使用`缩进`表示层级关系，`:`上一级以冒号结尾
> - 默认只允许`空格`，缩进不允许使用tab（可以编辑vimrc后支持）
> - 缩进的空格数不重要，只要相同层级的元素左对齐即可（默认两个空格）
> - `#`表示注释
> - `key`:空格`value`

```
# tail -n 1 /etc/bashrc
# vim
:set all
:help tabstop

# ls /etc/vimrc ~/.vimrc

$ cat > ~/.vimrc <<EOF
set number ts=2 et cuc sw=2
EOF
```

> `V` 当前行
>
> `G` Go跳到最后一行
>
> `>` 右缩进(sw=2)

A5. service

STEP	CMD	COMMENT	DNS	DHCP
1	nmcli | nmtui	网络		
2	hostnamectl	主机名		
3	yum search KEYWORD	查安装包名	dns	dhcp
samba, nfs	chmod, chown, setfacl	权限-文件系统(1/4)		
4	yum -y install PKG	安装软件	bind	dhcp-server
5	rpm -qc PKG | man -k nfs	查配置文件	bind	dhcp-server
6	vim /etc/..cfg(sec_service)	编辑(安全1/4)	/etc/named.conf	/etc/dhcp/dhcpd.conf
7	rpm -ql PKG | grep service systemctl list-unit-files | grep KEYWORD	查守护进程	bind	dhcp-server
8	systemctl enable --now DAEMON systemctl restart DAEMON	开机自启，立即启动 配置文件修改后，服务重启生效	named	dhcpd
9	firewall-cmd --permanent --add-service|--add-port ..., firewall-cmd --reload(sec_port)	防火墙(安全1/4)		
10	selinux(1/4)	文件系统-上下文关系 服务安全-布尔值 端口安全-端口标签		

A6. OBJECTIVE: SCORE

```
Manage Network Services: 87%
Manage Firewall Services: 100%
Manage SELinux: 100%
Manage DNS: 0%
Manage DHCP: 100%
Manage printers: 33%
Manage Email services: 100%
Manage a MariaDB database server: 100%
Manage HTTPD web access: 100%
Manage iSCSI: 50%
Manage NFS: 100%
Manage SMB: 75%
Use Ansible to Configure Standard Services: 80%
```

A7. 学习技巧

- word
- `Tab` 补全,`Tab` `Tab` 列出

```
# man command
```

```
# echo $?
```

A8. VMware+software

```
# yum -y install \
open-vm-tools-desktop.x86_64 \
xorg-x11-drv-vmware.x86_64
```

A9. ansible

> - configure
>   - inventory
> - playbook
>   - module

[student@workstation]

```
查询安装包名称，确认是否安装
$ yum search ansible
$ yum list ansible

query config 查找默认的配置文件和主机清单
$ rpm -qc ansible
/etc/ansible/ansible.cfg
/etc/ansible/hosts

确认四种生效的方式，以及优先级。考试时：当前目录的优先级
$ head /etc/ansible/ansible.cfg
1 export ANSIBLE_CONFIG=...
*2 ./ansible.cfg
3 ~/.ansible.cfg
4 /etc/ansible/ansible.cfg

$ mkdir playbook
$ cd playbook/
$ cp /etc/ansible/ansible.cfg .

确认生效的配置文件
$ ansible --version

$ vim /home/student/playbook/ansible.cfg
...
inventory      = /home/student/playbook/inventory
$ cp /etc/ansible/hosts inventory

$ vim inventory
...输出省略...
serverc
serverd

确认生效的主机清单
$ ansible-inventory --graph
@all:
```

```
  |--@ungrouped:
  |   |--`serverc`
  |   |--`serverd`
```

> **module**
>
> - command(id, hostname)
> - shell(*,|)
> - setup(facts)
> - debug(echo)
> - stat(when)

权限

```
$ ansible-doc -h

$ ansible-doc -t connection -l
$ ansible-doc -t connection ssh

$ ansible-doc -t become -l
$ ansible-doc -t become sudo
```

- Root
  - ansible.cfg/remote_user
  - Inventory/ansible_password

```
$ sshpass -p redhat ssh root@localhost id
```

```
$ cat ansible.cfg
inventory      = hosts
host_key_checking = False
remote_user = root
#become=True

$ cat hosts
workstation ansible_password=redhat

$ ansible workstation -a id
workstation | CHANGED | rc=0 >>
uid=0(root) gid=0(root) groups=0(root) ...
```

- User -=> root%redhat
  - ansible.cfg/#remote_user == $USER
  - inventory/ansible_password
  - ansible.cfg/become=True

- inventory/ansible_become_password

[student@bastion ~]

```
$ sshpass -p student ssh $USER@workstation "echo student | sudo -S id"
[sudo] password for student: uid=0(root) gid=0(root) groups=0(root)
context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

```
$ cat ansible.cfg
inventory      = hosts
host_key_checking = False
#remote_user = root
become=True

$ cat hosts
workstation ansible_password=student ansible_become_password=student

$ ansible workstation -a id
workstation | CHANGED | rc=0 >>
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-
s0:c0.c1023
```

A10. vim

4gg :4
Ctrl-v block visual
G Go end
I Insert

```
sw=shift
cursorcolumn = 查看列对齐
cursorline = 查看水平对齐
$ echo set number sw=2 ts=2 et cursorcolumn cursorline > ~/.vimrc
```

4

Shfit - v

G

Shift - >

. 重复上一次操作

ID		
1	`u`	Undo
2	`5` `g` `g`	Go
3	`Ctrl`-`V`	Virtual block
4	`j`* n	⬇️
5	`I`	III12iii3aaa456AAA
6	`Esc`	
7	`o`	Open
8	`x`	

A11. 培训环境 2 练习环境

> **VMware配置建议修改**
>
> - CPU * ==8==，根据物理机CPU，相等即可
> - MEM * ==8G==。根据物理机MEM=8GB，设置6GB，最小4GB
>
> **foundation**确认是否为`20年08月04日`版本
>
> ```
> $ ls /content/manifests/
> RH358-RHEL8.1-1.r2020080409-ILT+RAV-7-en_US.icmf
> ```

STEP	说明	
1	VMware恢复快照`INIT`	Cpu + Mem
2	启动虚拟机	
3	光驱插入`ex358.iso`， 同时复选`连接CD/DVD驱动器`	
4	执行脚本 $ `/run/media/kiosk/ex358/exam-setup.sh`	kiosk@foundaiton0 开始布署，约6分钟
5	$ echo > /home/kiosk/.ssh/known hosts && ssh root@localhost systemctl poweroff	关机
6	做快照	名称`EX358`
7	开机	

```
$ rht-vmctl status classroom
classroom RUNNING

$ rht-vmctl status all
bastion DEFINED
workstation DEFINED
servera DEFINED
serverb DEFINED
serverc DEFINED
serverd DEFINED
*$ rht-vmctl start bastion

-CMD
*$ rht-vmctl start servera
*$ rht-vmctl start serverb

-ANSIBLE
*$ rht-vmctl start workstation
*$ rht-vmctl start serverc
*$ rht-vmctl start serverd
```

A12. PC+VMware

OS		
win7	VMware-workstation-full-15.5.7	1、删除快照 2、改兼容性 3、改CPU+MEM
win>=8	VMware-workstation-full-16.1.2	

> CPU: AMD
>
> foundation 8.0 +RH358
> foundation 8.2

A13. 培训环境 KVM 快照

```
KVM自动关机，然后快照
$ rht-vmctl save all

查看快照
$ rht-vmctl listsaves all

确认恢复开机快照
$ rht-vmctl restore all -y
```

A14. 权限

ID	TYPE	BASE	ENHANCED
B1	filesystem	# ls -ld **/**var/www/html/	# ls -ldZ /var/www/html/ drwxr-xr-x. 2 root root system_u:object_r:==httpd_sys_content_t==:s0 6 Sep 2 2019 /var/www/html/
B2	service	134 <Directory "/var/www/html"> 147 Options Indexes FollowSymLinks 154 AllowOverride None 159 # ==Require all granted == Require host lab.example.com Require not host lab.example.org 160	# getsebool -a | grep http
B3	firewall	# firewall-cmd --permanent --add-service=http # firewall-cmd --reload	# semanage port -l |grep -w 80 http_port_t tcp 80, 81, 443, 488, 8008, 8009, 8443, 9000
E4	selinux	# getenforce Enforcing	

A15. 使用U盘布署培训环境

STEP		
1	下载==U_128G.vmdk==	群公告/培训环境安装U盘
2	VMware虚拟机，关机状态下添加`已有`磁盘，总线类型`SATA`	
3	启动虚拟机	
4	$ ==su -== Asimov	
5	# `lsblk` sda	确认U_128G，在系统中的磁盘名称
6	插入==64GB==优盘	
7	# `lsblk` sdb	确认优秀，在系统中的磁盘名称
8	# `dd if=/dev/sda of=/dev/sdb`	等一会儿

STEP	
1	物理机优盘启动
2	`f0 rh358`
3	指明时区