**今日所需资料：podman与**[**rhel-8.2-x86\_64-dvd.iso**](javascript:void(0);)

* **设置虚拟机的主机名**

]# **hostnamectl set-hostname rhel8.tedu.cn**

开启一个新的终端验证

* **关闭所有虚拟机的SELinux**

[root@rhel8 ~]# **vim /etc/selinux/config**

SELINUX=**disabled**

* **设置所有虚拟机防火墙**

[root@rhel8 ~]# **systemctl stop firewalld**

[root@rhel8 ~]#  **systemctl disable firewalld**

* **构建Yum仓库**

[root@rhel8 ~]# **mkdir /dvd**

[root@rhel8 ~]# **mount /dev/cdrom /dvd**

mount: /dvd: WARNING: device write-protected, mounted read-only.

[root@rhel8 ~]# **ls /dvd**

[root@rhel8 ~]# **vim /etc/yum.repos.d/dvd.repo**

**[haha]**

**name=haha**

**baseurl=file:///dvd/AppStream/**

**enabled=1**

**gpgcheck=0**

**[xixi]**

**name=haha**

**baseurl=file:///dvd/BaseOS/**

**enabled=1**

**gpgcheck=0**

[root@rhel8 ~]# **yum repolist -v**

[root@rhel8 ~]# **yum -y install vsftpd**

[root@rhel8 ~]# **vim /etc/fstab**

**此处省略一万字……**

**/dev/cdrom /dvd iso9660 defaults 0 0**

[root@rhel8 ~]# **umount /dvd**

[root@rhel8 ~]# **ls /dvd**

[root@rhel8 ~]# **mount -a**

mount: /dvd: WARNING: device write-protected, mounted read-only.

[root@rhel8 ~]# **ls /dvd**

* **修改网卡命名规则**

]# **ifconfig | head -2**

]# **vim /etc/default/grub**

…….

GRUB\_CMDLINE\_LINUX="……. quiet **net.ifnames=0 biosdevname=0**"

…….

]# **grub2-mkconfig -o /boot/grub2/grub.cfg**

Generating grub configuration file ...

done

]# **reboot**  #重启系统

]# **getenforce** #查看SELinux状态

]# **ifconfig | head -2** #查看网卡名称

* **配置IP地址**

[root@rhel8 ~]# **nmcli connection add type ethernet ifname eth0 con-name eth0**

[root@rhel8 ~]# **nmcli connection modify eth0**

**ipv4.method manual ipv4.addresses 192.168.4.100/24 connection.autoconnect yes**

[root@rhel8 ~]# **nmcli connection up eth0**

[root@rhel8 ~]# **ifconfig | head -2**

**模块化安装：类似于安装一组软件包**

[root@rhel8 ~]# **yum module list**

[root@rhel8 ~]# **yum -y module install container-tools**

**今日所需资料：podman**

* **将真机的podman目录传递到虚拟机rhel8中**

[root@localhost ~]# **ls /linux-soft/1**

cobbler.zip podman tools.tar.gz

[root@localhost ~]# **ls /linux-soft/1/podman/**

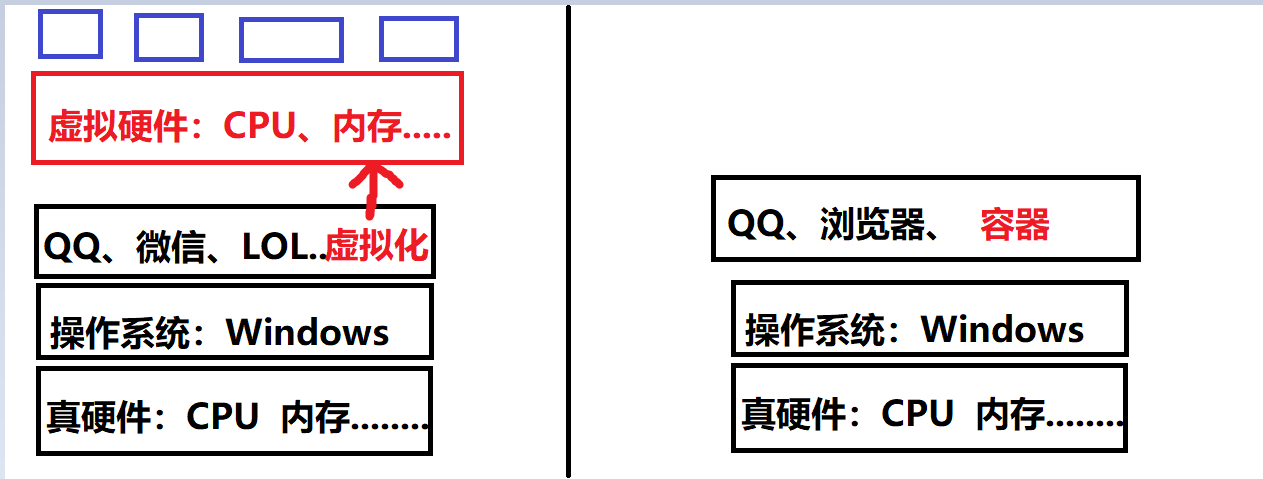
httpd.tar.gz myos.tar.gz nginx.tar.gz

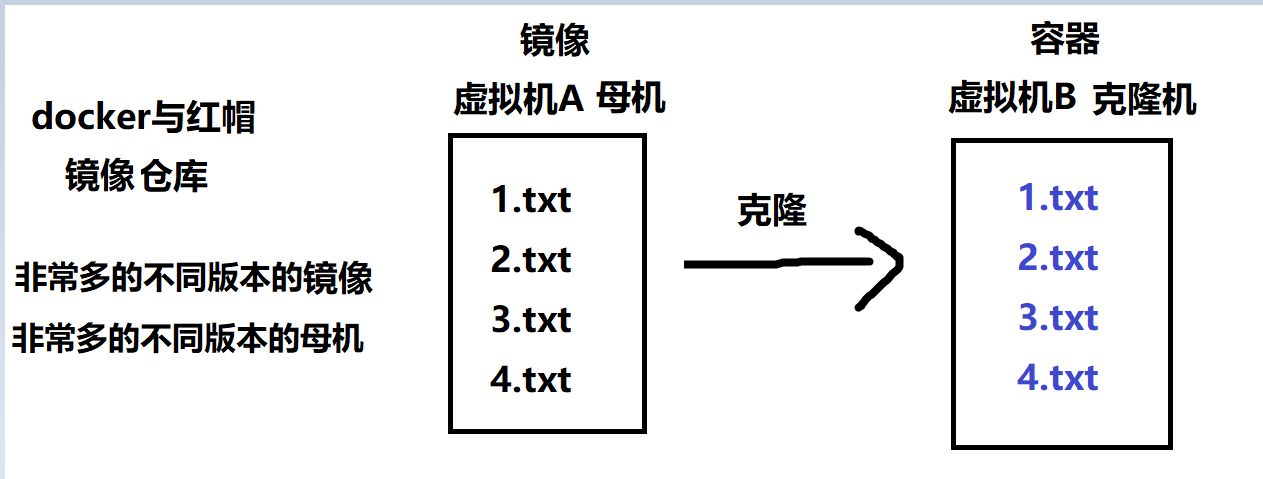
[root@localhost ~]# **scp -r**  **/linux-soft/1/podman/ root@192.168.4.100:/root**

#####################################

1. **容器基础概述**
   1. Linux中的容器是装应用的
   2. 容器就是将软件打包成标准化单元，用于开发、交付和部署
   3. 容器技术已经成为应用程序封装和交付的核心技术

* 优点
  + 相比于传统的虚拟化技术，容器更加简洁**高效**
  + 传统虚拟机需要给每个VM安装操作系统
  + 容器使用的共享公共库和程序





* 镜像是启动容器的核心，镜像由镜像仓库提供
* 在podman中容器是基于镜像启动的
* podman和容器的关系
  + podman是完整的一套容器管理系统
  + podman提供了一组命令，让用户更加方便直接地使用容器技术，而不需要过多关心底层内核技术
* podman所需软件
  + 系统软件，位于 rhel-8.2-x86\_64-dvd 源中
  + 利用系统光盘，构建Yum仓库

[root@localhost ~]# yum -y module install container-tools

* **获取镜像**
* 镜像的名称标识
  + 每一个镜像都对应唯一的镜像 id
  + 镜像名称（姓氏） + 标签（名字） = 唯一
  + 每一个镜像都有标签，如果没写就是默认标签 latest
  + 我们在调用镜像的时候，如果没有指定标签也是 latest
* 查找镜像（需要能访问互联网）
  + podman search 关键字

podman search httpd

* 下载镜像（需要能访问互联网）
  + podman pull 镜像名称:标签

podman pull localhost/myos:latest

* **导入镜像**
  + **podman load -i 备份文件.tar.gz**

**podman load -i /root/httpd.tar.gz**

* **镜像管理命令**
* 查看镜像
  + podman images
* 删除镜像
  + podman rmi 镜像名称:镜像标签
* **镜像管理练习**

]# **podman images** #查看当前有哪些镜像

]# **podman load -i /root/httpd.tar.gz**  #导入镜像

]# **podman images**  #查看当前有哪些镜像

]# **podman load -i /root/nginx.tar.gz**  #导入镜像

]# **podman load -i /root/myos.tar.gz**  #导入镜像

]# **podman images**  #查看当前有哪些镜像

* **镜像删除练习**

]# **podman images** #查看当前有哪些镜像

]# **podman rmi 2f5** #按照镜像的ID值，删除镜像

]# **podman images** #查看当前有哪些镜像

]# **podman load -i /root/myos.tar.gz** #导入镜像

]# **podman images**  #查看当前有哪些镜像

]# **podman images**

]# **podman rmi localhost/myos:nginx**  #删除镜像

]# **podman images**

]# **podman rmi localhost/myos:latest**  #删除镜像

]# **podman images**

1. **使用容器**

* podman run 命令
  + podman run -选项 镜像名称:镜像标签 启动命令
* 查看 run 的选项
  + podman help run
  + man podman-run
* run = 创建 + 启动 + 进入
* podman run 命令的选项
  + 选项 -i，交互式
  + 选项 -t，终端
  + 选项 -d，后台运行
  + 选项 --name 容器名字
* 启动容器，并进入容器

podman run -it myos:latest /bin/bash

* 可以通过命令行提示符，判定自己是否进入容器了
* **容器管理命令**
* 启动容器
  + podman run –选项 镜像名称:镜像标签 启动命令
* 查看容器
  + podman ps [ -a 所有容器id ] [ -q 只显示容器 id ]
* 删除容器
  + podman rm 容器id
* 容器管理命令启动、停止、重启
  + podman start|stop|restart 容器id
* 进入容器
  + podman exec -it 容器id 启动命令
* **容器初步练习**

]# **touch /etc/resolv.conf**

]# **podman run -it localhost/myos:httpd /bin/bash**

[root@2b0b7c62ab42 /]# **cat /etc/redhat-release**

[root@2b0b7c62ab42 /]# **useradd dcc**

[root@2b0b7c62ab42 /]# **id dcc**

[root@2b0b7c62ab42 /]# **exit**

[root@rhel8 ~]# **podman ps -a** #查看当前系统容器

[root@rhel8 ~]# **podman start 2b**  #利用容器id开启容器

[root@rhel8 ~]# **podman exec -it 2b /bin/bash** #进入容器

[root@2b0b7c62ab42 /]# **id dcc**

[root@2b0b7c62ab42 /]# **exit**

[root@rhel8 ~]# **podman ps -a**

[root@rhel8 ~]# **podman stop 2b** #利用容器id停止容器

[root@rhel8 ~]# **podman ps -a**

[root@rhel8 ~]# **podman rm 2b** #利用容器id删除容器

[root@rhel8 ~]# **podman ps -a**

* **容器放入后台练习**

]# **podman run --name abc01 -d localhost/myos:httpd**

]# **podman ps -a**

]# **podman exec -it abc01 /bin/bash** #进入abc01容器

[root@962aa837e17b html]# **cd /**

[root@962aa837e17b /]# **useradd tc**

[root@962aa837e17b /]# **id tc**

uid=1000(tc) gid=1000(tc) groups=1000(tc)

[root@962aa837e17b /]# **exit**

[root@rhel8 ~]# **podman stop abc01**

[root@rhel8 ~]# **podman rm abc01**

[root@rhel8 ~]# **podman ps -a**

1. **容器进阶-对外发布容器服务**

* 容器可以与宿主机的端口进行绑定
* 从而把宿主机变成对应的服务,不用关心容器的IP地址
* 我们使用 -p 参数把容器端口和宿主机端口绑定
* 同一宿主机端口只能绑定一个容器服务
* -p [可选IP]:宿主机端口:容器端口
* 例如:把宿主机变成 apache

podman run -itd -p 80:80 myos:httpd

* 例如:把宿主机变成 nginx

podman run -itd -p 80:80 myos:nginx

* **容器放入后台,端口绑定练习**

[root@rhel8 ~]# **podman run --name nsdweb -p 80:80 -d localhost/myos:httpd**

[root@rhel8 ~]# **podman ps -a**

[root@rhel8 ~]# **podman exec -it nsdweb /bin/bash**

[root@5b69bf6956b0 html]# **echo wo shi nsdweb > /var/www/html/index.html**

[root@5b69bf6956b0 html]# **/usr/sbin/httpd** #手动启动

[root@5b69bf6956b0 html]# **exit**

exit

[root@rhel8 ~]# **curl http://192.168.4.100**

wo shi nsdweb

[root@rhel8 ~]#

1. **容器进阶-容器共享卷**

* podman容器不适合保存任何数据
* podman可以映射宿主机文件或目录到容器中
  + 目标对象不存在就自动创建
  + 目标对象存在就直接覆盖掉
  + 多个容器可以映射同一个目标对象来达到数据共享的目的
* 启动容器时，使用 -v 映射参数

podman run -itd -v 宿主机对象:容器内对象 镜像名称:标签

* **容器终极练习**

]# **podman stop nsdweb** #停止容器

]# **podman rm -f nsdweb**  #强制删除容器

]# **podman run --name rqweb -p 80:80 -v /opt:/var/www/html -d localhost/myos:httpd**

[root@rhel8 ~]# **echo wo shi niuniubenben > /opt/index.html**

[root@rhel8 ~]# **curl 192.168.4.100**

wo shi niuniubenben

[root@rhel8 ~]#

1. **容器进阶-管理系统服务（了解）**

* **systemd**一个更高效的系统&服务管理器
  + 开机服务并行启动，各系统服务间的精确依赖
  + 服务目录：/usr/lib/systemd/system/
  + 主要管理工具：systemctl
* 管理员服务文件默认路径
  + /usr/lib/systemd/system/
* 生成服务启动配置文件

--files：生成文件类型

podman generate systemd --name 容器名 --files

* 重新加载服务启动配置文件

systemctl daemon-reload

* **容器之光练习（必须是相对路径，当前路径必须是/usr/lib/systemd/system）**

]# **cd /usr/lib/systemd/system**

]# **podman ps -a**

]# **podman generate systemd --name rqw eb --files**

]# **vim container-rqweb.service**

]# **systemctl daemon-reload**  #重新加载服务配置文件

]# **systemctl daemon-reload**

]# **podman stop rqweb #手动停止**

]# **systemctl stop container-rqweb**

]# **podman ps -a**

]# **systemctl start container-rqweb**

]# **podman ps -a**

]# **systemctl enable container-rqweb**  #设置开机自启