[root@openstack ~]# source keystonerc\_admin

[root@openstack ~(keystone\_admin)]# openstack user list

通过命令行管理用户

• 创建user2用户,指定密码为tedu.cn

[root@vh02 ~(keystone\_admin)]# openstack user create --password

tedu.cn user2

• 设置user2的email地址

[root@vh02 ~(keystone\_admin)]# openstack user set --email

user2@tedu.cn user2

• 列出所有用户

[root@vh02 ~(keystone\_admin)]# openstack user list

• 查看user2信息

[root@vh02 ~(keystone\_admin)]# openstack user show user2

通过命令行管理用户(续1)

• 指定user2可以访问myproject,角色为\_member\_

[root@vh02 ~(keystone\_admin)]# openstack role add --user user2 --

project myproject \_member\_

• 查看user2在myproject中的角色

[root@vh02 ~(keystone\_admin)]# openstack role list --project

myproject --user user2

• 禁用用户

[root@vh02 ~(keystone\_admin)]# openstack user set --disable user2

• 激活用户

[root@vh02 ~(keystone\_admin)]# openstack user set --enable user2

通过命令行管理用户(续2)

**• 修改user2的密码为redhat**

[root@vh02 ~(keystone\_admin)]# openstack user set --password

redhat user2

• 将user2从myproject中移除

[root@vh02 ~(keystone\_admin)]# openstack role remove --project

myproject --user user2 \_member\_

• 删除user2用户

[root@vh02 ~(keystone\_admin)]# openstack user delete user2

创建实例失败(重起服务)

[root@openstack ~]# cd /usr/lib/systemd/system

[root@openstack system]# ls openstack-nova-compute.service

openstack-nova-compute.service

[root@openstack system]# systemctl restart openstack-nova-compute.service

||

重启服务：

for xx in libvirtd messagebus openstack-nova-compute; do service $xx restart; done

报错日志：

cat /var/log/nova/nova-compute.log

2018-08-17 11:56:45.217 1679 INFO #这个地方应该都是info

**Cloud第四天**

**概述**

**什么是容器**

**• 容器技术已经成为应用程序封装和交付的核心技术**

**• 容器技术的核心有以下几个内核技术组成:**

**– CGroups(Control Groups)-资源管理**

**– NameSpace-进程隔离**

**– SELinux安全**

**• 由于是在物理机上实施隔离,启动一个容器,可以像**

**启动一个进程一样快速**

**什么是Docker**

**• Docker是完整的一套容器管理系统**

**• Docker提供了一组命令,让用户更加方便直接地使**

**用容器技术,而丌需要过多关心底层内核技术**

**Docker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**

**Docker的缺点**

**• 容器的隔离性没有虚拟化强**

**• 共用Linux内核,安全性有先天缺陷**

**• SELinux难以驾驭**

**• 监控容器和容器排错是挑战**

**安装前准备**

**• 需要64位操作系统**

**• 至少RHEL6.5以上的版本,强烈推荐RHEL7**

**• 关闭防火墙(丌是必须)**

**安装Docker**

**• 软件包列表:**

**– docker-engine**

**– docker-engine-selinux**

**[root@jacob ~]# rpm -ivh docker-engine-1.12.1-1.el7.centos.x86\_64.rpm**

**[root@jacob ~]# rpm -ivh docker-engine-selinux-1.12.1-**

**1.el7.centos.noarch.rpm**

**[root@jacob ~]# systemctl start docker**

**[root@jacob ~]# systemctl enable docker**

**什么是镜像**

**• 在Docker中容器是基于镜像启动的**

**• 镜像是启动容器的核心**

**• 镜像采用分层设计**

**• 使用快照的COW技术,确保底层数据丌丢失**

**Docker hub镜像仓库**

**• https://hub.docker.com**

**• Docker官方提供公共镜像的仓库(Registry)**

**下载、上传镜像**

**• 下载镜像(从镜像仓库中下载镜像)**

**[root@server0 ~]# docker help pull**

**docker pull [OPTIONS] NAME[:TAG|@DIGEST]**

**[root@server0 ~]# docker pull rhel7**

**• 上传镜像(上传镜像到仓库)**

**[root@server0 ~]# docdocker help push**

**Usage: docker push [OPTIONS] NAME[:TAG]**

**[root@server0 ~]# docker push rhel7**

**导入、导出镜像**

**• 导入镜像(通过tar包文件导入镜像)**

**[root@server0 ~]# docker load < xx.tar**

**[root@server0 ~]# docker images**

**• 导出镜像(将本地镜像导出为tar文件)**

**[root@server0 ~]# docker images**

**[root@server0 ~]# docker save image\_name > xx.tar**

**启动镜像**

**• 启动centos镜像生成一个容器**

**[root@server0 ~]# docker images**

**[root@server0 ~]# docker run -it centos bash**

**• 开启另一个终端(查看容器信息)**

**[root@server0 ~]# docker ps**

**镜像常用命令**

**命令列表**

**• 命令列表**

**– docker images //查看镜像列表**

**– docker history //查看镜像制作历史**

**– docker inspect //查看镜像底层信息**

**– docker pull //下载镜像**

**– docker push //上传镜像**

**– docker rmi //删除本地镜像**

**– docker save //镜像另存为tar包**

**– docker load //使用tar包导入镜像**

**– docker search //搜索镜像**

**– docker tag //修改镜像名称和标签**

**docker images**

**• 查看镜像列表**

**– 镜像仓库名称**

**– 镜像标签**

**– 镜像ID**

**– 创建时间**

**– 大小**

**[root@jacob ~]# docker images**

**docker history**

**• 查看镜像历史**

**– 了解镜像制作过程**

**– 详绅参考后面的dockerfile内容**

**[root@jacob ~]# docker history centos**

**docker inspect**

**• 查看镜像底层信息**

**– 了解镜像环境变量、存储卷、标签等信息**

**[root@jacob ~]# docker inspect centos**

**docker rmi**

**• 删除本地镜像**

**– 注意:启动容器时删除镜像会提示错误**

**[root@jacob ~]# docker rmi centos**

**docker save|load**

**• 保存本地镜像另存为tar文件**

**– 方便其他人使用tar包导入镜像**

**[root@jacob ~]# docker save centos > centos.tar**

**• 使用tar包文件导入镜像**

**[root@jacob ~]# docker load < centos.tar**

**docker tag**

**• 重命名镜像名称(复制)**

**[root@jacob ~]# docker images**

**[root@jacob ~]# docker tag centos:latest test:v1**

**[root@jacob ~]# docker images**

**容器常用命令**

**命令列表**

**• 命令列表**

**– docker run //运行容器**

**– docker ps //查看容器列表**

**– docker stop //关闭容器**

**– docker start //启动容器**

**– docker restart //重启容器**

**– docker attach|exec //进入容器**

**– docker inspect //查看容器底层信息**

**– docker top //查看容器进程列表**

**– docker rm //删除容器**

**docker run**

**• 使用镜像启动容器**

**[root@jacob ~]# docker run -it centos bash**

**[root@bbb51ac87b08 /]#exit**

**[root@jacob ~]# docker run -itd centos bash**

**b8f218f2341c12655e6092d7d2e7fd5229824fdefce84075fb3a9569ebf82079**

**[root@jacob ~]# docker ps**

**[root@docker01 ~]# docker run -d nginx # -d 启动的是服务**

**docker ps**

**• 列出容器列表**

**– docker ps 查看正在运行的容器**

**– docker ps -a 查看所有容器列表**

**– docker ps -aq 仅显示容器id**

**[root@server0 ~]# docker ps**

**docker stop|start|restart**

**• 管理容器**

**– docker stop 关闭容器**

**– docker start 开启容器**

**– docker restart 重启容器**

**[root@server0 ~]# docker ps**

**[root@server0 ~]# docker stop b8f**

**docker attach|exec**

**• 进入容器**

**– docker attach 进入容器,exit会导致容器关闭**

**– docker exec 进入容器,退出时丌会关闭容器**

**[root@server0 ~]# docker ps**

**[root@server0 ~]# docker attach b8f**

**[root@server0 ~]# docker exec -it 7e4e bash**

**docker inspect**

**• 查看容器底层信息**

**[root@server0 ~]# docker ps**

**[root@server0 ~]# docker inspect 7e4e**

**docker top**

**• 查看容器进程列表**

**[root@server0 ~]# docker ps**

**[root@server0 ~]# docker top 7e4e**

**docker rm**

**• 删除容器**

**– 注意,删除正在运行的容器时会提示错误**

**[root@server0 ~]# docker rm 7e4e**

**[root@server0 ~]# docker rm `docker ps -aq`**

**故障分析及排除**

**• 原因分析**

**– 问题1:提示error:You cannot remove a running container**

**• 解决办法**

**– 问题1:先关闭容器,才能删除容器**