1. 云计算初步与kvm虚拟化实践

安装kvm yum install qeum-kvm qeum-kvm-tools libvirt

启动libvirt systemctl start libvirtd

Systemctl libvirtd on

Windows机器安装 tightVNC  
创建虚拟机镜像 qemu-img create -f raw /opt/CentOS-7-x86\_64.raw 10G

安装virt yum install virt-install

创建虚拟机 virt-install --virt-type kvm --name CentOS-7-x86\_64 --ram 2048 \

--crom=/root/CentOS-7-x86\_64-DVD.iso --disk path=/opt/CentOS-7-x86\_64.raw \

--network network=defalt --graphics vnc,listen=0.0.0.0 --noautoconsole

Vnc连接 在安装选项按e 后面加quiet net.ifnames=0 biosdevname=0

查看虚拟机 virsh list --all

启动虚拟机 virsh start 刚命令查询的名字

关闭虚拟机 virsh shutdown 虚拟机名字

启动虚拟机后设置网卡 PEERDNS=no 注释 uuid 注释ipv6相关

安装网络工具 yum install net-tools

虚拟机配置文件在 /etc/libvirt/qemu目录下虚拟机同名xml文件

编辑虚拟机xml文件 virsh edit 虚拟机名字

备份xml virsh dumpxml 虚拟机名字 > backup.xml

删除虚拟机 virsh undefine 虚拟机名字

恢复虚拟机 virsh define 虚拟机名字

挂起虚拟机 virsh suspend 虚拟机名字

恢复虚拟机 virsh resume 虚拟机名字

修改cpu个数

编辑虚拟机xml文件 virsh edit 虚拟机名字

修改<vcpu placement=’auto’ current=’1’>4</vcpu>

动态增加内存 需小于最大内存 virsh qemu-monitor-command 虚拟机名字 --hmp --cmd balloon 512

查看镜像文件 qemu-img info Centos-7-x86\_64.raw 镜像文件名字

转换镜像为 qcow2 qemu-img convert –f raw –O qcow2 CentOS-7-x86\_64.raw test.qcow2

查看网络 brctl show

增加网卡 brctl addbr br0

桥接网卡 brctl addif br0 eth0

删除网卡ip ip addr del dev eth0 192.168.56.11/24

设置br0 ip addr bro 192.168.56.11/24 up

增加路由 route add default gw 192.168.56.1

修改虚拟机网卡为桥接 virsh edit 虚拟机名字 修改 <interface type=’bridge’>

<source network=’default’>改为<source bridge=’br0’>

虚拟化优化

Pid绑定cpu执行 task set –cp 0 6573

内存设置ept技术

开启大页内存 cat /sys/kernel/mm/transparent\_hugepage/enabled

磁盘算法 cat /sys/block/sda/queue/scheduler

Echo noop > /sys/block/sda/queue/scheduler

1. DOCKER

安装docker yum install docker

启动docker systemctl start docker

查看docker状态 systemctl status docker

查看镜像 docker images

查找镜像 docker search centos

在线下载镜像 docker pull centos

Docker save –o centos.tar centos

导入镜像 docker load –input centos.tar

删除镜像 docker rmi 镜像id

启动容器 docker run centos /bin/echo ‘Hello world’

查看 docker ps –a

docker run --name mydocker -t -i centos /bin/bash

启用刚命名的 docker start mydocker

进入启动的 docker attach mydocker 这种进入方式不安全

安装 yum install util-linux

获取容器第一个进程pid docker inspect -f "{{ .State.Pid}}" mydocker

进入启动的nsenter -t 5974 -m -u -i -n –p

Vim docker\_in.sh

#!/bin/bash

#use nsenter to access docker

Docker\_in(){

NAME\_ID=$1

PID=$( docker inspect -f "{{ .State.Pid}}" $NAME\_ID)

nsenter -t $PID -m -u -i -n –p

}

Docker\_in $1

执行命令 docker exec mydocker whoami

进入 docker exec –it mydocker /bin/bash

删除容器 docker rm mydocker

后台运行容器 docker run –d nginx

查看容器日志 docker log id/名字

随机绑定端口 docker run –d –P nginx

iptables -t nat –vnL

固定端口绑定 docker run –d –p 192.168.56.11:81:80 –name mynginx nginx

查看绑定端口 docker port mynginx