一.kvm 虚拟化环境搭建准备

1.硬件环境

kvm 只能部署在物理机上面。

2. BIOS 开启 VT

Virtualization Tech [Enabled]

3. 查看 **cpu** 是否支持 **kvm** 全虚拟化

```
# grep "flags" /proc/cpuinfo |uniq
flags : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat
pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm syscall nx pdpe1gb rdtscp lm
constant_tsc nonstop_tsc arat pni monitor ds_cpl vmx smx est tm2 ssse3 cx16
xtpr sse4_1 sse4_2 popcnt lahf_lm
```

如果输出的结果包含VMX，它是Intel 处理器虚拟机技术标志，如果包含SVM，它是AMD 处
理器虚拟机技术的标志，看到VMX 或SVM，说明支持全虚拟化。另处linux 发行版本必须在64bit 环境中才能使用KVM。

二.虚拟化环境部署

1. 安装 **kvm** 和虚拟化管理软件包

```
yum -y install kvm virt-* libvirts bridge-utils qemu-img
```

kvm 软件包。kvm 软件包中含有 KVM 内核模块，它可在默认 Linux 内核中提供 KVM 管理程序。
libvirts 安装虚拟机管理工具，使用virsh 等命令来管理和控制虚拟机
bridge-utils 安装网络支持，设置桥接
virt-* 创建、克隆虚拟机等命令，以及图形化管理工具virt-manager
qemu-img 安装 qemu 组件，使用 qemu 命令来创建磁盘等

2. 加载 **kvm** 模块

```
# modprobe kvm-intel
```

3. 查看**kvm** 模块是否被加载

```
# lsmod |grep kvm
kvm_intel 85256 0
kvm 225952 1 kvm_intel
```

4.**reboot** 并确认

确认kvm 模块是否被加载

```
# lsmod |grep kvm
kvm_intel 85256 0
kvm 225952 2 ksm,kvm_intel
```

#如出现以上内容，说明 kvm 模块正确加载

kvm 相关模块的路径

cd /lib/modules/2.6.18-348.1.1.el5/weak-updates/kmod-kvm/

pwd

/lib/modules/2.6.18-348.1.1.el5/weak-updates/kmod-kvm

[root@wg-vm-s2 kmod-kvm]# ls

ksm.ko kvm-amd.ko kvm-intel.ko kvm.ko

5. 检查物理主机虚拟化完成情况

virsh list

Id Name State

#如出现以上提示说明虚拟化被正确安装

6.关闭防火墙和selinux

由于防火墙和selinux会阻止libvirt为虚拟机创建网络设备。

关闭防火墙

/etc/init.d/iptables stop

关闭selinux

[root@test22 ~]# cat /etc/selinux/config

This file controls the state of SELinux on the system.

SELINUX= can take one of these three values:

enforcing - SELinux security policy is enforced.

permissive - SELinux prints warnings instead of enforcing.

disabled - No SELinux policy is loaded.

SELINUX=disabled   #更改为disabled

SELINUXTYPE= can take one of these two values:

targeted - Targeted processes are protected,

mls - Multi Level Security protection.

SELINUXTYPE=targeted

然后命令行执行：setenforce 0

7.配置网络桥接

cat /etc/sysconfig/network-scripts/ifcfg-eth0

Broadcom Corporation NetXtreme BCM5720 Gigabit Ethernet PCIe

DEVICE=eth0

HWADDR=90:B1:1C:2A:A4:40

ONBOOT=yes

BOOTPROTO=none

TYPE=Ethernet

BRIDGE=br0

#注意不要写错。

cat /etc/sysconfig/network-scripts/ifcfg-br0

Broadcom Corporation NetXtreme BCM5720 Gigabit Ethernet PCIe

DEVICE=br0

#HWADDR=90:B1:1C:2A:A4:40

ONBOOT=yes
BOOTPROTO=static
TYPE=Bridge
IPADDR=192.168.3.200
NETMASK=255.255.255.0
GATEWAY=192.168.3.254

配置完 eth0 和 br0 重启网卡：serivce network restart
查看已有桥接：
brctl show
bridge name bridge id STP enabled interfaces
br0 8000.90b11c2aa440 no eth0　　（配置正确的桥接 是有mac地址的）
br1 8000.90b11c2aa441 no eth1
错误桥接：
brctl show
bridge name bridge id STP enabled interfaces
br0 8000.000000000000 no eth0

8. 修改**vnc** 监听端口
cd /etc/libvirt/
ls
libvirtd.conf nwfilter qemu qemu.conf
cp qemu.conf qemu.conf.panglu-$(date +%F)
vi qemu.conf +12
11 #
12 # vnc_listen = "0.0.0.0" #把前面的#注释去掉，开启vnc 监听
13
14
15 # Enable use of TLS encryption on the VNC server. This requires
16 # a VNC client which supports the VeNCrypt protocol extension.
17 # Examples include vinagre, virt-viewer, virt-manager and vencrypt

9. 开启相应的服务
/etc/init.d/messagebus start #linux ICP 服务
Starting system message bus: [OK]

/etc/init.d/libvirtd restart #启动virbr0
libvirtd (pid 3665) is running...
添加到开机启动
chkconfig messagebus on
echo “/etc/init.d/libvirtd start”>>/etc/rc.local

到此，kvm 服务端安装完成。

三. 安装虚拟机
1. 创建虚拟机磁盘目录

mkdir /data/vmdisk -p

2.创建 iso 文件目录

mkdir /iso

把光盘放入到光驱，dd 命令把iso 文件拷贝到/iso 目录下

dd if=/dev/cdrom of=/iso/centos5.iso

3.创建虚拟磁盘文件

qemu-img create -f qcow2 -o preallocation=metadata dev_5931.img 300G

重要参数：-o preallocation=metadata 预分配磁盘，硬盘空间不会立即占用

注意：ext3 不支持此参数，ext4 支持。

4.安装虚拟机命令及参数

virt-install --name=dev_5934_44 --ram 8192 --vcpus=2 -f dev_5934_44.img --cdrom ../iso/CentOS-6.4-x86_64-bin-DVD1to2/CentOS-6.4-x86_64-bin-DVD1.iso --graphics vnc,listen=0.0.0.0,port=5934, --force --autostart

注意端口号设定值，用 vnc 连接时需要用。

5. 接下来客户端 vnc 去连接。

#192.168.3.200 是宿主机 IP，5900 是安装命令指定的 vnc 端口

四. 虚拟机硬件配置更改篇

1.解决 kvm 环境下可以使用 shutdown 命令让虚拟机关机,但不生效。

进入不能关机的虚拟机中：

[root@localhost ~]# yum -y install acpid ；安装 acpid 服务

[root@localhost ~]# service acpid restart ；启动服务

[root@localhost ~]# chkconfig acpid on ；加入开机启动

通过以上步骤后就可以让虚拟机响应 shutdown 和 reboot 命令了

2.修改 kvm 中虚拟机的内存大小及 cpu 数量。

a.查看虚拟机

virsh # list --all

Id	名称	状态
2	test_centos	running
4	qishi	running
13	qishi2	running
14	cloud_monitor_5921	running

20 test22 running
- test33 关闭

b.更改要修改的配置文件

[root@nfs ~]# virsh edit test22 ##注意 vi 直接编辑不生效

<domain type='kvm'>
<name>test22</name>
<uuid>eb342f67-e70c-194b-5291-e91010ed996f</uuid>
<memory unit='KiB'>5120000</memory>
<currentMemory unit='KiB'>5120000</currentMemory> #512000→为内存大小，单位为 K 修改此数字，例如要调整为 8G 则设置为 8192000
<vcpu placement='static'>4</vcpu> #4→为 cpu 数量，更改为 5 个则将 4 修改为 5
<os>
<type arch='x86_64' machine='rhel6.5.0'>hvm</type>
<boot dev='hd'/>
</os>

修改完保存→关闭虚拟机→启动虚拟机 （注意直接重启不生效）

[root@nfs ~]# virsh shutdown test22
Domain vm01 is being shutdown
[root@nfs ~]# virsh start test22

检查：

启动后登录虚拟机，查看内存及 cpu,发现已经更改。

查看内存：# free -m

查看 cpu 数量 # cat /proc/cpuinfo

3.虚拟机网卡 mac 冲突的解决办法：

a.连接上虚拟机。

b.关掉冲突主机。

ifdown eth0

c.修改 eth0 的 mac

#sbin/ifconfig eth0 hw ether 00：AA:BB:CC:11：22 ##-->分一个尽量不易出现的 mac

d.修改 eth0 配置文件中的 mac

vi /etc/sysconfig/network-scripts/ifcfg-eth0

Hwaddr: 00：AA:BB:CC:11：22

e.重启网卡

service network restart

f.我们用命令行更改的 mac 在当前生效，重启后失效，所以需要加入到开机启动。

echo “ifdown eth0”>>/etc/rc.local

echo “sbin/ifconfig eth0 hw ether 00：AA:BB:CC:11：22”

echo “ifup eth0”

好了，网卡 mac 冲突解决了！

4.给虚拟机增加硬盘。

1、给当前磁盘扩容

a.找到需要扩容的磁盘文件。

```
[root@qishi_test qemu]# cd /data/VHOST/
[root@qishi_test VHOST]# ll
总用量 318393772
-rw-r--r-- 1 qemu qemu 322171961344 6 月   2 12:33 cloud_monitor_5921.img
-rw-r--r-- 1 root root  32212254720 5 月  30 16:54 jia_qishi2.img
-rw-r--r-- 1 qemu qemu  64432963584 6 月   2 12:33 test2.img
-rwxr-xr-x 1 root root  53695545344 6 月   2 10:05 test33.img
-rw-r--r-- 1 qemu qemu  21474836480 5 月  30 18:53 test_jia.img
```

b.我们现在给 test22 增加 10G 空间,他的磁盘文件对应的 test2.img。

```
qemu-img resize test2 +10G
```

c.关闭虚拟机 test22→启动虚拟机 test22(注意：重启不生效)

检查：

登陆虚拟机，fdisk -l 发现磁盘增加了 10G 的空间，我们可以对其进行分区挂载等操作了。

2、给虚拟机添加一块磁盘。

a. 创建文件硬盘的镜像

```
qemu-img create -f raw /disk/sdb6/c1d6.img 10G #指定路径、名称、及大小。
Formatting '/disk/sdb6/c1d6.img', fmt=raw size=10737418240
```

b.打开虚拟机列表

```
virsh # list --all
 Id    名称                         状态
----------------------------------------------------
 2     test_centos                    running
 4     qishi                          running
 13    qishi2                         running
 14    cloud_monitor_5921             running
 20    test22                         running
 -     test33                         关闭
```

c.在此 test22 硬盘扩容

```
virsh # edit test22
#找到硬盘配置(原来的系统硬盘)
<disk type='file' device='disk'>
      <driver name='qemu' type='raw' cache='none'/>
      <source file='/data/VHOST/test2.img'/>
      <target dev='hda' bus='ide'/>
      <address type='drive' controller='0' bus='0' target='0' unit='0'/>
 </disk>
#新加磁盘配置
    <disk type='block' device='cdrom'>
      <driver name='qemu' type='raw'/>
      <target dev='hdc' bus='ide'/>
      <readonly/>
```

```
      <address type='drive' controller='0' bus='1' target='0' unit='0'/>
    </disk>
```

注意:dev 名称设定不能一样。bus 不能一样。一般源为“0”则加一个数字+1,即 1,2,3....

#保存退出

d.关闭虚拟机→启动虚拟机（注意：重启不能生效）

```
[root@nfs ~]# virsh  shutdown test22
Domain vm01 is being shutdown
[root@nfs ~]# virsh  start test22
```

检查：

虚拟机启动后，登陆进去使用命令 #fdisk -l 查看新加的磁盘。

以下为磁盘分区挂载步骤，若不关注忽略！

在 c1 中，进行硬盘查检并分区

```
~
 sudo fdisk -l

Disk /dev/vda: 42.9 GB, 42949672960 bytes
16 heads, 63 sectors/track, 83220 cylinders, total 83886080 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x000516aa

   Device Boot      Start         End      Blocks   Id  System
/dev/vda1   *        2048      499711      248832   83  Linux
/dev/vda2          501758    83884031    41691137    5  Extended
/dev/vda5          501760    83884031    41691136   8e  Linux LVM

Disk /dev/vdb: 10.7 GB, 10737418240 bytes
16 heads, 63 sectors/track, 20805 cylinders, total 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/vdb doesn't contain a valid partition table

Disk /dev/mapper/u1210-root: 38.4 GB, 38394658816 bytes
255 heads, 63 sectors/track, 4667 cylinders, total 74989568 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000
```

```
Disk /dev/mapper/u1210-root doesn't contain a valid partition table

Disk /dev/mapper/u1210-swap_1: 4294 MB, 4294967296 bytes
255 heads, 63 sectors/track, 522 cylinders, total 8388608 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/mapper/u1210-swap_1 doesn't contain a valid partition table
/dev/vdb 已经被识别，接下来 分区,格式化,挂载,使用
```

硬盘分区

```
~ sudo fdisk /dev/vdb

Command (m for help): p

Disk /dev/vdb: 161.1 GB, 161061274112 bytes
16 heads, 63 sectors/track, 312076 cylinders, total 314572801 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x3b49c6a0

  Device Boot      Start         End      Blocks   Id  System

Command (m for help): n
Partition type:
   p   primary (0 primary, 0 extended, 4 free)
   e   extended
Select (default p): p
Partition number (1-4, default 1):
Using default value 1
First sector (2048-314572800, default 2048):
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-314572800, default 314572800):
Using default value 314572800

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.

#分区生效
```

```
~ sudo partprobe

~ sudo fdisk -l
Disk /dev/vdb: 10.7 GB, 10737418240 bytes
2 heads, 17 sectors/track, 616809 cylinders, total 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xf0432cd6

   Device Boot      Start         End      Blocks   Id  System
/dev/vdb1            2048    20971519    10484736   83  Linux
```

格式化

```
~ sudo mkfs -t ext4 /dev/vdb1
mke2fs 1.42.5 (29-Jul-2012)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
9830400 inodes, 39321344 blocks
1966067 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=4294967296
1200 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
    4096000, 7962624, 11239424, 20480000, 23887872

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
```

挂载

```
~ sudo mount /dev/vdb1 /home/cos/hadoopb
~ df -h
Filesystem              Size  Used Avail Use% Mounted on
/dev/mapper/u1210-root   36G  1.1G   33G   4% /
udev                    2.0G  4.0K  2.0G   1% /dev
tmpfs                   791M  232K  791M   1% /run
```

```
none                    5.0M     0  5.0M   0% /run/lock
none                    2.0G     0  2.0G   0% /run/shm
none                    100M     0  100M   0% /run/user
/dev/vda1               228M   29M  188M  14% /boot
/dev/vdb1               9.9G  151M  9.2G   2% /home/cos/hadoopb
```

五 **centos 6.0** 克隆虚拟机步骤
宿主机：192.168.3.100
模版机web-1：192.168.3.102
克隆机web-5：192.168.3.110
1） 关闭/暂停模版虚拟机进行克隆，（注意新克隆的虚拟磁盘文件的名字不能与模版机相同）
virt-clone -o web-1 -n web-5 -f /data/vmdisk/web-5.qcow2
2）然后手动更改xml 配置文件的vnc 端口。
vi /etc/libvirt/qemu/web-5.xml
3）重启libvirtd 服务使更改的端口生效。
/etc/init.d/libvirtd restart
4）启动新克隆的虚拟机
virsh start web-5
5）然后客户端vnc 去连。
6）更改web-5 的eth0 的ip
7）重启 network 服务
六拷贝**xml** 文件方式安装新虚拟机
安装新虚机web-10
1)制作虚拟机镜像
cd /data/vmdisk/
cp web-1.qcow2 web-10.qcow2
2)创建xml 文件
Cd /etc/libvirt/qemu
cp web-1.xml web-10.xml
修改相应的参数
uuid mac vnc 硬盘文件位置 虚拟机名字
uuid：随意更改，保证唯一
mac：随意更改，保证唯一
vnc：指定端口号，别和其他虚拟机冲突
虚拟磁盘文件位置：/data/vmdisk/web-10.qcow2
#<source file='/data/vmdisk/web-10.qcow2'/>
虚拟机名字：web-10
vi web-10.xml
:%S/web-1/web-10/g
3)创建虚拟机
virsh define web-10.xml
4)启动虚拟机
virsh start web-10
七 设置虚拟机自启动
#virsh autostart 虚拟机名（例如web-1）
在/etc/libvirt/qemu/network/autostart 目录下会出现 web-1.xml
此时，如果重启宿主机，web-1 虚拟机是不会自动启动的，需要重启libvirtd 服务
所以记得把 libvirtd 服务设置成开机自启动，这样宿主机重启，虚拟机就会自启动。