KVM 虚拟化技术

惨绿少年 Linux运维, 云计算, 虚拟化 0评论 来源：本站原创 79℃ 字体： 小 中 大

1.1 前言

1.1.1 什么是虚拟化?

在计算机技术中，虚拟化（技术）或虚拟技术（英语：Virtualization）是一种资源管理技术，是将计算机的各种实体资源（CPU、内存、磁盘空间、网络适配器等），予以抽象、转换后呈现出来并可供分区、组合为一个或多个电脑配置环境。

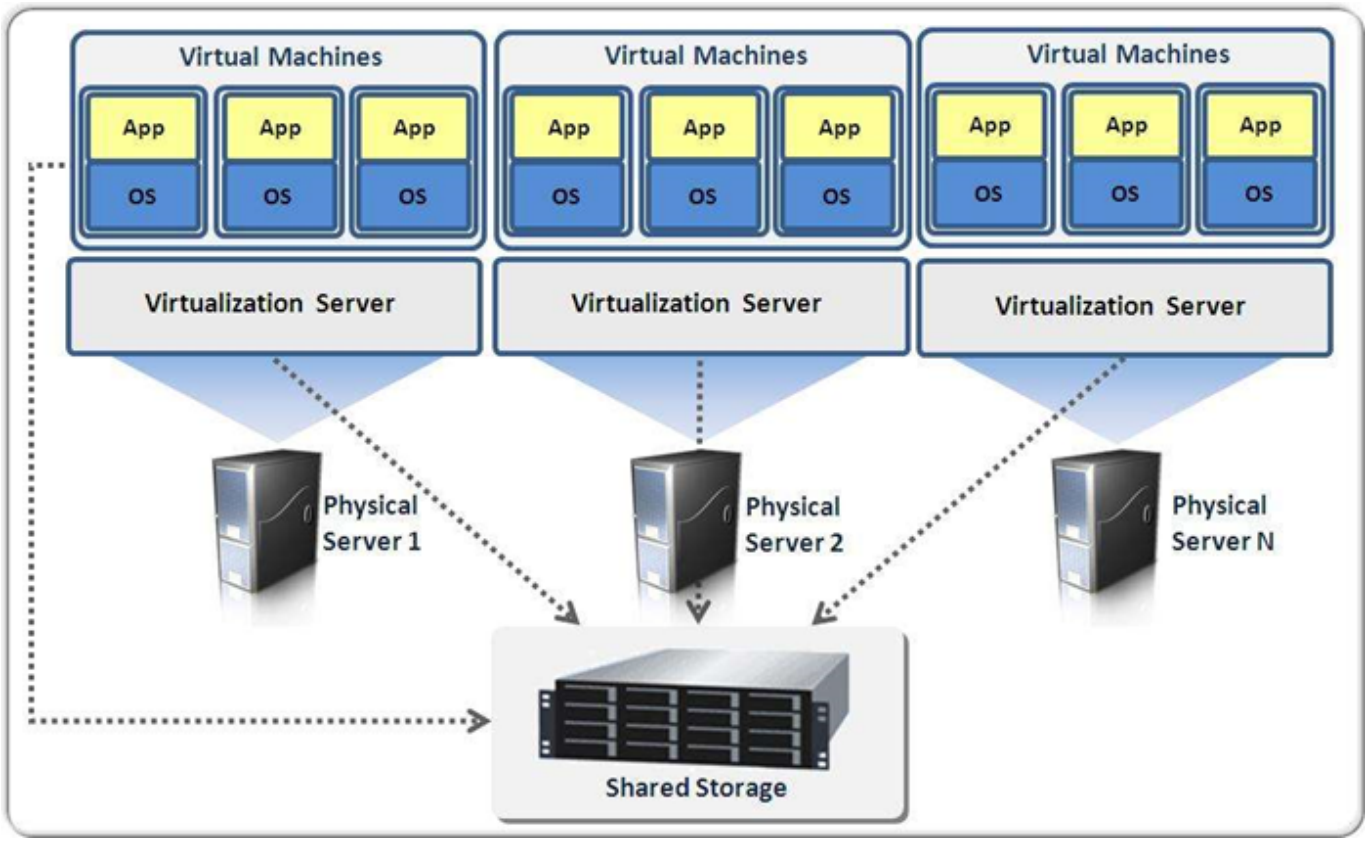


图 – 虚拟化示意图

由此，打破实体结构间的不可切割的障碍，使用户可以比原本的配置更好的方式来应用这些电脑硬件资源。这些资源的新虚拟部分是不受现有资源的架设方式，地域或物理配置所限制。

一般所指的虚拟化资源包括计算能力和数据存储。

由于目前信息技术领域的很多企业都曾在宣传中将该企业的某种技术称为虚拟化技术，这些技术涵盖的范围可以从Java虚拟机技术到系统管理软件，这就使得准确的界定虚拟技术变得困难。因此各种相关学术论文在谈到虚拟技术时常常提到的便是如前面所提到的那个不严格的定义。

1.1.2 为什么要用虚拟化

- 同一台物理机运行多个不同版本应用软件
- 硬件依赖性较低和便于数据迁移

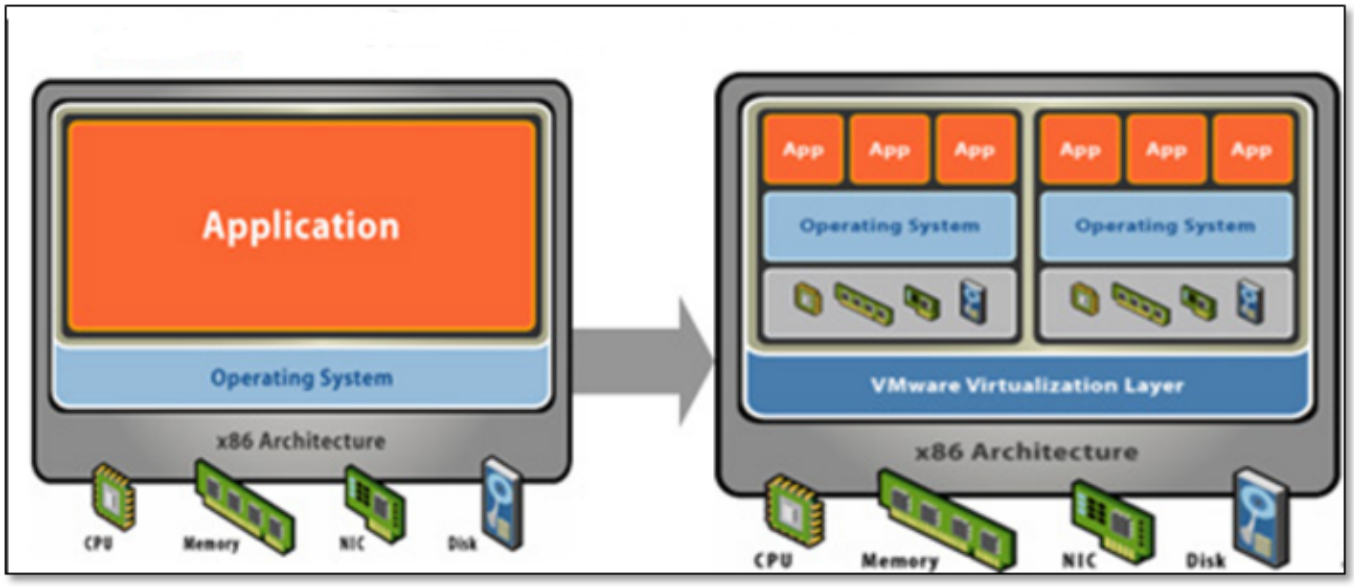


图 – 虚拟化前后对比

详情参考<1.1.3 虚拟化技术的优势>。

1.1.3 虚拟化技术的优势

1.降低运营成本

服务器虚拟化降低了IT基础设施的运营成本，令系统管理员摆脱了繁重的物理服务器、OS、中间件及兼容性的管理工作，减少人工干预频率，使管理更加强大、便捷。

2.提高应用兼容性

服务器虚拟化提供的封装性和隔离性使大量应用独立运行于各种环境中，管理人员不需频繁根据底层环境调整应用，只需构建一个应用版本并将其发布到虚拟化后的不同类型平台上即可。

3.加速应用部署

采用服务器虚拟化技术只需输入激活配置参数、拷贝虚拟机、启动虚拟机、激活虚拟机即可完成部署，大大缩短了部署时间，免除人工干预，降低了部署成本。

4.提高服务可用性

用户可以方便地备份虚拟机，在进行虚拟机动态迁移后，可以方便的恢复备份，或者在其他物理机上运行备份，大大提高了服务的可用性。

5.提升资源利用率

通过服务器虚拟化的整合，提高了CPU、内存、存储、网络等设备的利用率，同时保证原有服务的可用性，使其安全性及性能不受影响。

6.动态调度资源

在服务器虚拟化技术中，数据中心从传统的单一服务器变成了统一的资源池，用户可以即时地调整虚拟机资源，同时数据中心管理程序和数据中心管理员可以灵活根据虚拟机内部资源使用情况灵活分配调整给虚拟机的资源。

7.降低能源消耗

通过减少运行的物理服务器数量，减少CPU以外各单元的耗电量，达到节能减排的目的。

1.1.4 KVM简介

KVM，基于内核的虚拟机（英语：*Kernel-based Virtual Machine*，缩写为 KVM），是一种用于Linux内核中的虚拟化基础设施，可以将Linux内核转化为一个hypervisor。KVM在2007年2月被导入Linux 2.6.20核心中，以可加载核心模块的方式被移植到FreeBSD及illumos上。

KVM在具备Intel VT或AMD-V功能的x86平台上运行。它也被移植到S/390，PowerPC与IA-64平台上。在Linux内核3.9版中，加入ARM架构的支持。

KVM目前由Red Hat等厂商开发，对CentOS/Fedora/RHEL等Red Hat系发行版支持极佳。

1.1.5 关于KVM

1. KVM是开源软件，全称是kernel-based virtual machine（基于内核的虚拟机）。
2. 是x86架构且硬件支持虚拟化技术（如 intel VT 或 AMD-V）的Linux全虚拟化解决方案。
3. 它包含一个为处理器提供底层虚拟化 可加载的核心模块kvm.ko（kvm-intel.ko或kvm-AMD.ko）。
4. KVM还需要一个经过修改的QEMU软件（qemu-kvm），作为虚拟机上层控制和界面。
5. KVM能在不改变linux或windows镜像的情况下同时运行多个虚拟机，（它的意思是多个虚拟机使用同一镜像）并为每一个虚拟机配置个性化硬件环境（网卡、磁盘、图形适配器……）同时KVM还能够使用ksm技术帮助宿主服务器节约内存。
6. 在主流的Linux内核，如2.6.20以上的内核均已包含了KVM核心。

1.1.6 关于Virtual Machine Manager

在电脑运算中，红帽公司的Virtual Machine Manager是一个虚拟机管理员，可以让用户管理多个虚拟机。

基于内核的虚拟机libvirt与Virtual Machine Manager。

Virtual Machine Manager可以让用户:

- 创建、编辑、引导或停止虚拟机。
- 查看并控制每个虚拟机的控制台。
- 查看每部虚拟机的性能以及使用率。
- 查看每部正在运行中的虚拟机以及主控端的即时性能及使用率信息。

不论是在本机或远程，皆可使用KVM、Xen、QEMU。

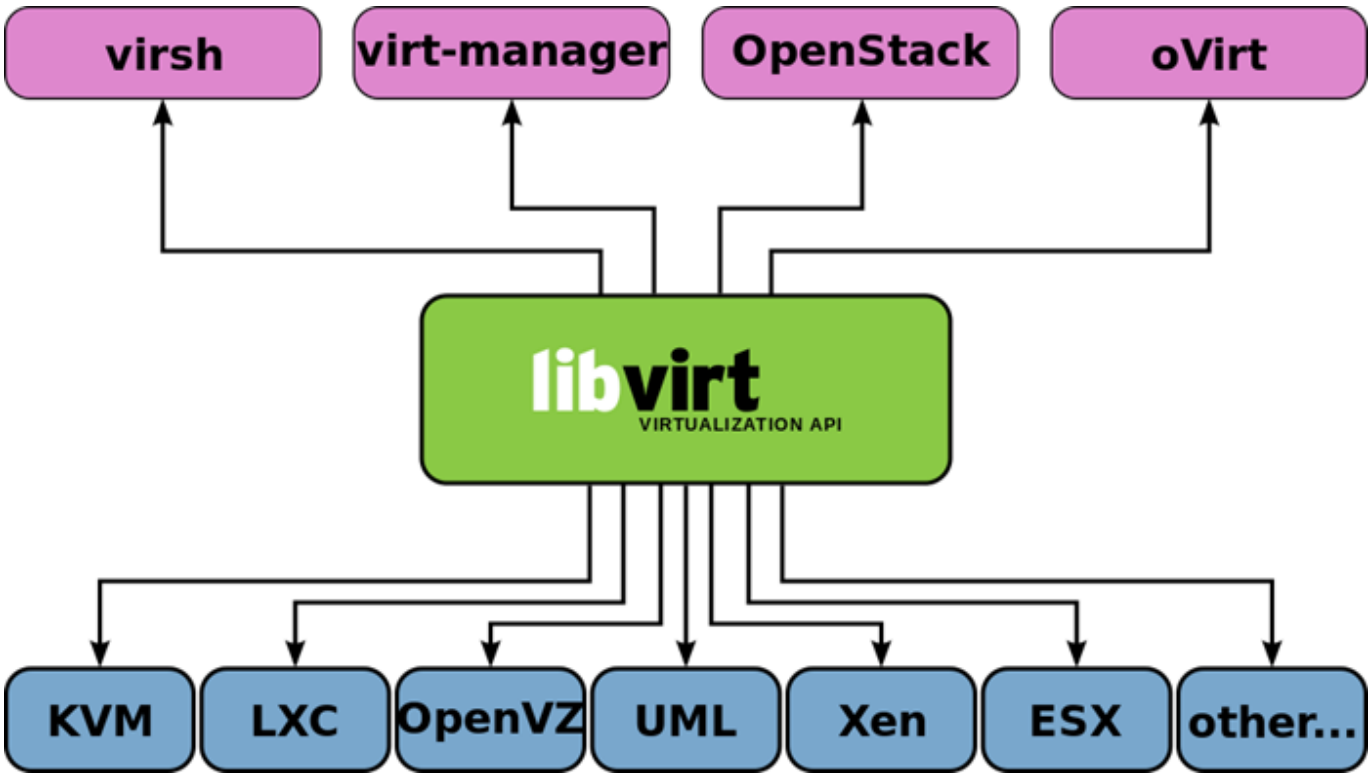


图 – libvirt服务

1.1.7 其他虚拟化软件

Xen

Xen是一个开放源代码虚拟机监视器，由XenProject开发。它打算在单个计算机上运行多达128个有完全功能的操作系统。

在旧（无虚拟硬件）的处理器上执行Xen，操作系统必须进行显式地修改（“移植”）以在Xen上运行（但是提供对用户应用的兼容性）。这使得Xen无需特殊硬件支持，就能达到高性能的虚拟化。

QEMU

QEMU是一套由Fabrice Bellard所编写的模拟处理器的自由软件。它与Bochs，PearPC近似，但其具有某些后两者所不具备的特性，如高速度及跨平台的特性。经由KVM（早期为kqemu加速器，现在kqemu已被KVM取代）这个开源的加速器，QEMU能模拟至接近真实电脑的速度。QEMU有两种主要运作模式：

User mode模拟模式，亦即是用户模式。

QEMU能引导那些为不同中央处理器编译的Linux程序。而Wine及Dosemu是其主要目标。

System mode模拟模式，亦即是系统模式。

QEMU能模拟整个电脑系统，包括中央处理器及其他周边设备。它使得为系统源代码进行测试及除错工作变得容易。其亦能用来在一部主机上模拟数部不同虚拟电脑。

1.2 KVM部署与使用

系统环境说明

```
[root@kvm ~]# cat /etc/redhat-release
CentOS Linux release 7.4.1708 (Core)
[root@kvm ~]# uname -r
3.10.0-693.el7.x86_64
[root@kvm ~]# sestatus
SELinux status:                 disabled
[root@kvm ~]# systemctl status firewalld.service
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset: enabled)
   Active: inactive (dead)
     Docs: man:firewalld(1)
[root@kvm ~]# hostname -I
172.16.1.240 10.0.0.240
# kvm主机内存不能低于4GB
```

1.2.1 安装KVM虚拟化软件

安装依赖包(可以使用本地yum源)

```
yum install libvirt* virt-* qemu-kvm* -y
```

安装软件说明内容:

```
libvirt    # 虚拟机管理
virt       # 虚拟机安装克隆
qemu-kvm   # 管理虚拟机磁盘
```

启动服务

```
[root@kvm ~]# systemctl start libvirtd.service
[root@kvm ~]# systemctl status libvirtd.servic
```

安装VNC软件:

下载vnc软件方法，tightvnc官网：http://www.tightvnc.com

VNC软件，用于VNC（Virtual Network Computing），为一种使用RFB协议的显示屏画面分享及远程操作软件。此软件借由网络，可发送键盘与鼠标的动作及即时的显示屏画面。

VNC与操作系统无关，因此可跨平台使用，例如可用Windows连接到某Linux的电脑，反之亦同。甚至在没有安装客户端程序的电脑中，只要有支持JAVA的浏览器，也可使用。

安装VNC时，使用默认安装即可，无需安装server端。

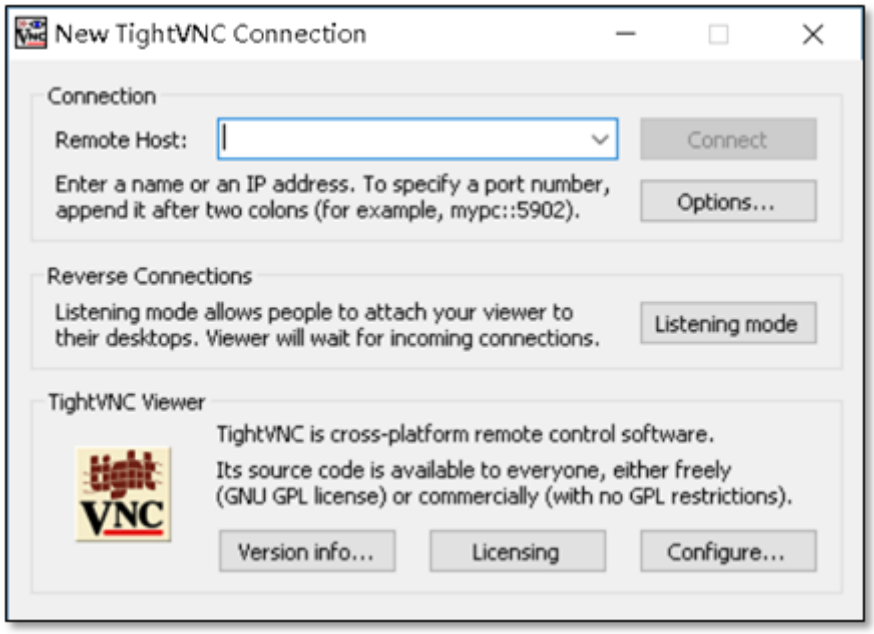


图 – vnc软件

1.2.2 配置第一台KVM虚拟机

使用命令

```
[root@kvm ~]# virt-install --virt-type kvm --os-type=linux --os-variant rhel7 --name centos7 --m
```

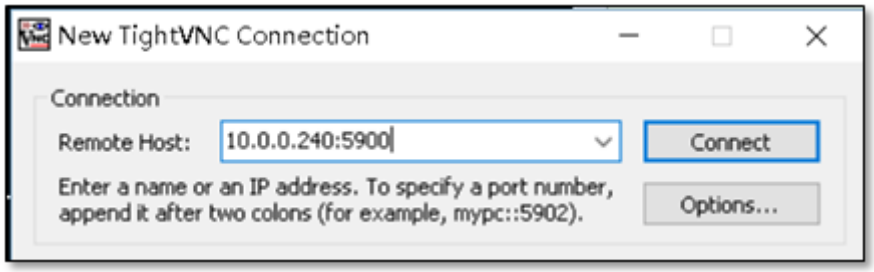
注意：需要先将镜像文件拷贝到 /data/CentOS-7-x86_64-DVD-1511.iso 。

使用参数说明：

参数	参数说明
–virt-type HV_TYPE	要使用的管理程序名称 (kvm, qemu, xen, …)
–os-type	系统类型
–os-variant DISTRO_VARIANT	在客户机上安装的操作系统，例如：‘fedora18’、‘rhel6’、‘winxp’ 等。
-n NAME, –name NAME	客户机实例名称
–memory MEMORY	配置客户机虚拟内存大小
–vcpus VCPUS	配置客户机虚拟 CPU(vcpu) 数量。
–disk DISK	指定存储的各种选项。
-cdrom CDROM	光驱安装介质

-w NETWORK, –network NETWORK	配置客户机网络接口。
–graphics GRAPHICS	配置客户机显示设置。
虚拟化平台选项:	
-v, –hvm	这个客户机应该是一个全虚拟化客户机
-p, –paravirt	这个客户机应该是一个半虚拟化客户机
–container	这个客户机应该是一个容器客户机
–virt-type HV_TYPE	要使用的管理程序名称 (kvm, qemu, xen, …)
–arch ARCH	模拟 CPU 架构
–machine MACHINE	机器类型为仿真类型
其它选项:	
–noautoconsole	不要自动尝试连接到客户端控制台
–autostart	主机启动时自动启动域。
–noreboot	安装完成后不启动客户机。
以上信息通过 ” virt-install –help ” 获得。	

在启动的同时使用vnc连接



下面就进入到安装系统的操作，关于系统安装的方法参考：http://www.cnblogs.com/clsn/p/7489784.html

1.2.3 KVM虚拟机管理操作

virsh命令常用参数总结

参数	参数说明
基础操作	
list	查看虚拟机列表，列出域
start	启动虚拟机，开始一个（以前定义的）非活跃的域
shutdown	关闭虚拟机，关闭一个域
destroy(危险)	强制关闭虚拟机，销毁（停止）域
vncdisplay	查询虚拟机vnc端口号
配置管理操作	
dumpxml	导出主机配置信息
undefine	删除主机
define	导入主机配置
domrename	对虚拟机进行重命名
挂起与恢复	
suspend	挂起虚拟机
resume	恢复虚拟机
自启动管理	
autostart	虚拟机开机启动
autostart –disable	取消虚拟机开机启动
以上参数通过 “virsh –help” 获得。	

操作过程：

KVM虚拟机配置文件位置

```
[root@kvm ~]# ll /etc/libvirt/qemu/centos7.xml
```

修改KVM虚拟机配置的方法

```
[root@kvm ~]# virsh edit centos7
```

使用该命令修改可以对文件进行语法校验。

备份与恢复

备份虚拟机配置(关机时备份):

```
[root@kvm ~]# virsh dumpxml centos7  > centos7.xml
```

删除虚拟机配置

```
# 查看
[root@kvm ~]# virsh list --all
 Id    名称                         状态
----------------------------------------------------
 -     centos7                        关闭
 # 删除
[root@kvm ~]# virsh undefine centos7
域 centos7 已经被取消定义
[root@kvm ~]# virsh list --all
 Id    名称                         状态
----------------------------------------------------
```

导入虚拟机

```
# 导入
[root@kvm ~]# virsh define centos7-off.xml
定义域 centos7（从 centos7-off.xml）
# 查看
[root@kvm ~]# virsh list --all
 Id    名称                         状态
----------------------------------------------------
 -     centos7                        关闭
```

修改虚拟机名称

```
# 重命名
[root@kvm ~]# virsh domrename centos7 clsn7
Domain successfully renamed
# 查看
[root@kvm ~]# virsh list
 Id    名称                         状态
----------------------------------------------------
 9     clsn7                          关闭
```

虚拟机挂起与恢复

```
# 挂起虚拟机
[root@kvm ~]# virsh suspend clsn7
域 clsn7 被挂起
# 查看状态
[root@kvm ~]# virsh list --all
 Id    名称                         状态
----------------------------------------------------
 9     clsn7                          暂停
```

恢复虚拟机

```
[root@kvm ~]# virsh resume clsn7
域 clsn7 被重新恢复
```

查询虚拟机vnc端口

```
[root@kvm ~]# virsh vncdisplay clsn7
:0
# :0 即 为 5900 端口，以此类推 :1为5901 。
```

开机自启动设置

```
# 设置 libvirtd 服务开机自启动。
[root@kvm ~]# systemctl is-enabled libvirtd.service
enabled
```

设置宿主机开机虚拟机在其他

```
[root@kvm ~]# virsh autostart clsn7
域 clsn7标记为自动开始
# 实质为创建软连接
[root@kvm ~]# ll /etc/libvirt/qemu/autostart/clsn7.xml
lrwxrwxrwx 1 root root 27 1月  22 12:17 /etc/libvirt/qemu/autostart/clsn7.xml -> /etc/libvirt/qe
```

取消开机自启动

```
[root@kvm ~]# virsh autostart --disable clsn7
域 clsn7取消标记为自动开始
```

1.3 kvm虚拟机console登录

1.3.1 CentOS 7.X 版本console登录

配置console登录

在clsn7虚拟机内操作(该操作**仅限centos7**):

```
[root@kvm ~]# grubby --update-kernel=ALL --args="console=ttyS0,115200n8"
[root@kvm ~]# reboot
# 115200n8：能显示虚拟机的启动过程
```

重启完成后，使用virsh console 连接虚拟机。

```
[root@kvm ~]# virsh console clsn7
连接到域 clsn7
换码符为 ^]
CentOS Linux 7 (Core)
Kernel 3.10.0-327.el7.x86_64 on an x86_64

clsn7 login: root
Password:
Last login: Mon Jan 22 12:24:48 from 192.168.122.1
```

```
[root@clsn7 ~]# w
 12:26:11 up 0 min,  1 user,  load average: 0.09, 0.03, 0.01
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
root     ttyS0                     12:26    3.00s  0.02s  0.01s w
```

1.3.2 CentOS 6.X 版本console登录

使用virsh console连接CentOS 6虚拟主机方法：

安装一台centos6的kvm虚拟机

```
virt-install --virt-type kvm --os-type=linux --os-variant rhel6 \
--name clsn6 --memory 1124 --vcpus 1 \
--disk /data/clsn6/clsn-6.raw,format=raw,size=10 \
--cdrom /data/CentOS-6.9-x86_64-bin-DVD1.iso \
--network network=default --graphics vnc,listen=0.0.0.0,port=5901 \
--noautoconsole
```

新安装一台虚拟机后，是无法通过virsh console 命令连入虚拟机中的，这时我们需要开启虚拟机的console功能。

以下操作都在虚拟机中进行

1、添加ttyS0的许可，允许root登陆

```
[root@clsn6 ~]# echo "ttyS0" >> /etc/securetty
```

2、编辑/etc/grub.conf中加入console=ttyS0

在该文件的第16行。kernel选项后添加

```
[root@clsn6 ~]# sed -i '/\tkernel/s#.*#& console=ttyS0#g' /etc/grub.conf
[root@clsn6 ~]# sync  # 同步配置到 /boot/grub/grub.conf
[root@clsn6 ~]# cat -n  /etc/grub.conf
     1    # grub.conf generated by anaconda
     2    #
     3    # Note that you do not have to rerun grub after making changes to this file
     4    # NOTICE:  You have a /boot partition.  This means that
     5    #          all kernel and initrd paths are relative to /boot/, eg.
     6    #          root (hd0,0)
     7    #          kernel /vmlinuz-version ro root=/dev/vda3
     8    #          initrd /initrd-[generic-]version.img
     9    #boot=/dev/vda
    10    default=0
    11    timeout=5
    12    splashimage=(hd0,0)/grub/splash.xpm.gz
    13    hiddenmenu
    14    title CentOS 6 (2.6.32-696.el6.x86_64)
    15        root (hd0,0)
    16        kernel /vmlinuz-2.6.32-696.el6.x86_64 ro root=UUID=48532582-c271-4c0a-b55f-395fe16
    17        initrd /initramfs-2.6.32-696.el6.x86_64.img
```

3、编辑/etc/inittab

在最后一行加入内容 S0:12345:respawn:/sbin/agetty ttyS0 115200

```
[root@clsn6 ~]# echo 'S0:12345:respawn:/sbin/agetty ttyS0 115200' >>/etc/inittab
```

4、以上操作都完成后，重启虚拟机

```
[root@clsn6 ~]# reboot
```

以下操作在kvm宿主机上执行

1、检查虚拟机的状态

```
[root@kvm ~]# virsh list --all
 Id    名称                         状态
----------------------------------------------------
 11    clsn7                          running
 21    clsn6                          running
```

2、进行连接测试

```
[root@kvm ~]# virsh console clsn6
连接到域 clsn6
换码符为 ^]  # 注：退出virsh console连接的方法，使用组合键Ctrl+]即可

CentOS release 6.9 (Final)
Kernel 2.6.32-696.el6.x86_64 on an x86_64

clsn6 login: root
Password:
Last login: Mon Jan 22 05:44:25 on ttyS0
[root@clsn6 ~]# who
root     ttyS0        2018-01-22 05:50
# 登陆成功，查看登陆接口为之前设置的ttyS0
```

1.4 KVM虚拟机磁盘、快照与克隆

1.4.1 磁盘管理

KVM qcow2、raw、vmdk等镜像格式说明：http://blog.csdn.net/zhengmx100/article/details/53887162

```
# 创建一块qcow2的虚拟硬盘(仅测试使用，无实际意义)
[root@kvm data]# qemu-img create -f qcow2 clsn.qcow2 2G
[root@kvm data]# ls -l
```

查看当前虚拟机硬盘信息

```
[root@kvm ~]# qemu-img info /data/clsn.raw
image: /data/clsn.raw
file format: raw
virtual size: 10G (10737418240 bytes)
disk size: 1.1G
```

raw转qcow2格式：

```
# 参数说明
[root@kvm data]# qemu-img  --help |grep convert
qemu-img convert [-f fmt] [-O output_fmt] filename  output_filename
```

转换原有磁盘格式

```
[root@kvm data]# qemu-img convert -f raw -O qcow2 clsn.raw clsn.qcow2
```

修改clsn7 虚拟机配置文件

```
[root@kvm data]# virsh edit clsn7
修改前：
    'file' device='disk'>
      'qemu' type='raw'/>
      '/data/clsn.raw'/>
      'vda' bus='virtio'/>

'pci' domain='0x0000' bus='0x00' slot='0x06' function='0x0'/>

修改后：
    'file' device='disk'>
      'qemu' type='qcow2'/>
      '/data/clsn.qcow2'/>
      'vda' bus='virtio'/>

'pci' domain='0x0000' bus='0x00' slot='0x06' function='0x0'/>
```

删除原磁盘文件

```
[root@kvm data]# \rm clsn.raw
```

启动虚拟机

```
[root@kvm data]# virsh start clsn7

[root@kvm data]# virsh list --all
 Id    名称                         状态
----------------------------------------------------
 22    clsn7                          running
```

1.4.2 KVM虚拟机添加硬盘

进入硬盘存放目录

```
[root@kvm ~]# cd /data
```

创建一块新的硬盘

```
[root@kvm data]# qemu-img create -f qcow2 clsn7-add01.qcow2 5G
Formatting 'clsn7-add01.qcow2', fmt=qcow2 size=5368709120 encryption=off cluster_size=65536 lazy
```

查看创建的硬盘信息

```
[root@kvm data]# qemu-img info clsn7-add01.qcow2
image: clsn7-add01.qcow2
file format: qcow2
virtual size: 5.0G (5368709120 bytes)
disk size: 196K
cluster_size: 65536
Format specific information:
    compat: 1.1
    lazy refcounts: false
```

为虚拟机添加硬盘

```
[root@kvm data]# virsh attach-disk clsn7 /data/clsn7-add01.qcow2 vdb --live --cache=none --subdr
# 成功附加磁盘
```

参数说明:

参数	参数说明
vdb	第二块硬盘
–live	热添加
–subdriver	驱动类型

调整已添加硬盘的大小

```
[root@kvm data]# virsh --help |grep disk
    attach-disk                    #附加磁盘设备
    detach-disk                    #分离磁盘设备
```

将已挂载的磁盘卸载下来

```
[root@kvm data]# virsh detach-disk clsn7  vdb
成功分离磁盘
```

调整磁盘大小

```
# 使用参数
[root@kvm data]# qemu-img --help |grep resize
  resize [-q] filename [+ | -]size
```

增加1G容量

```
[root@kvm data]# qemu-img resize clsn7-add01.qcow2 +1G
Image resized.
[root@kvm data]# qemu-img info clsn7-add01.qcow2
image: clsn7-add01.qcow2
file format: qcow2
```

```
virtual size: 6.0G (6442450944 bytes)
disk size: 260K
cluster_size: 65536
Format specific information:
    compat: 1.1
    lazy refcounts: false
```

重新讲磁盘添加到虚拟机

```
[root@kvm data]# virsh attach-disk clsn7 /data/clsn7-add01.qcow2 vdb --live --cache=none --subdr
```

以下在虚拟机中操作

格式化磁盘

```
[root@clsn7 ~]# mkfs.xfs /dev/vdb
```

挂载磁盘

```
[root@clsn7 ~]# df -h |grep /dev/vdb
/dev/vdb        6.0G   33M  6.0G   1% /opt
```

使用 **xfs_growfs** 刷新磁盘的信息

```
[root@clsn7 ~]# xfs_growfs --help
xfs_growfs: invalid option -- '-'
Usage: xfs_growfs [options] mountpoint
```

1.4.3 快照管理

注意:raw格式的磁盘无法创建快照

创建快照

```
[root@kvm data]# virsh snapshot-create clsn7
已生成域快照 1516607756
```

查看主机快照列表

```
[root@kvm data]# virsh snapshot-list  clsn7
 名称                 生成时间                 状态
------------------------------------------------------------
 1516607756           2018-01-22 15:55:56 +0800 running
# 注：该名称为unix时间戳(格林威治时间)
```

查看快照信息

```
[root@kvm data]# virsh snapshot-info  clsn7 --snapshotname 1516607756
```

登陆虚拟机，进行删除操作

```
[root@clsn7 /]# ls -1|egrep -v 'proc|sys|run' |rm -rf
```

还原快照

```
[root@kvm data]# virsh snapshot-revert clsn7 --snapshotname 1516607756
```

删除快照

```
[root@kvm data]# virsh snapshot-delete  clsn7 --snapshotname 1516607756
```

快照配置文件位置

```
[root@kvm data]# cd  /var/lib/libvirt/qemu/snapshot/
[root@kvm snapshot]# tree
.
└── clsn7
    └── 1516607756.xml
```

1.4.4 kvm虚拟机克隆

复制一个虚拟机，需修改如 MAC 地址，名称等所有主机端唯一的配置。

虚拟机的内容并没有改变：virt-clone 不修改任何客户机系统内部的配置，它只复制磁盘和主机端的修改。所以像修改密码，修改静态 IP 地址等操作都在本工具复制范围内。如何修改此类型的配置，请参考 virt-sysprep。

克隆常用命令：

```
[root@kvm ~]# virt-clone --auto-clone -o clsn7
WARNING  设置图形设备端口为自动端口，以避免相互冲突。
正在分配 'clsn-clone.ra 4% [-                ] 1.5 MB/s | 464 MB  01:50:18 ETA
```

参数说明:

参数	参数说明
–auto-clone	从原始客户机配置中自动生成克隆名称和存储路径。
-o ORIGINAL_GUEST, **–original ORIGINAL_GUEST**	原始客户机名称；必须为关闭或者暂停状态。

1.5 kvm虚拟机网络管理

1.5.1 桥接网络配置

1、设置桥接网络

```
[root@kvm ~]# virsh iface-bridge eth0 br0
使用附加设备 br0 生成桥接 eth0 失败
已启动桥接接口 br0
```

查看网卡配置文件

```
# 查看 eth0 配置文件
[root@kvm ~]# cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
ONBOOT=yes
BRIDGE="br0"
# 查看 br0 配置文件
[root@kvm ~]# cat /etc/sysconfig/network-scripts/ifcfg-br0
DEVICE="br0"
ONBOOT="yes"
TYPE="Bridge"
BOOTPROTO="none"
IPADDR="10.0.0.240"
NETMASK="255.255.255.0"
GATEWAY="10.0.0.254"
IPV6INIT="yes"
IPV6_AUTOCONF="yes"
DHCPV6C="no"
STP="on"
DELAY="0"
```

2、修改虚拟机网络配置

```
[root@kvm ~]# virsh edit clsn7
修改前：
    'network'>
      '52:54:00:42:bf:bc'/>
      'default'/>
      'virtio'/>

'pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>

修改后：
    'bridge'>
      '52:54:00:42:bf:bc'/>
      'br0'/>
      'virtio'/>

'pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>
```

查看宿主机网桥

```
[root@kvm ~]# brctl show
bridge name    bridge id        STP enabled    interfaces
br0        8000.000c294d551b    yes        eth0
virbr0     8000.5254006aaa40    yes        virbr0-nic
                                vnet0
                                vnet1
```

查看防火墙规则：

```
[root@kvm ~]# iptables -t nat  -nvL
Chain PREROUTING (policy ACCEPT 195 packets, 24665 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain INPUT (policy ACCEPT 131 packets, 16209 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 272 packets, 24045 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain POSTROUTING (policy ACCEPT 272 packets, 24045 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 RETURN     all  --  *      *       192.168.122.0/24     224.0.0.0/24
    1   328 RETURN     all  --  *      *       192.168.122.0/24     255.255.255.255
   29  1740 MASQUERADE  tcp  --  *      *       192.168.122.0/24    !192.168.122.0/24     ma
    0     0 MASQUERADE  udp  --  *      *       192.168.122.0/24    !192.168.122.0/24     ma
    3   252 MASQUERADE  all  --  *      *       192.168.122.0/24    !192.168.122.0/24
```

3、修改kvm虚拟机网卡配置文件

```
[root@clsn7 ~]# cat /etc/sysconfig/network-scripts/ifcfg-eth0
TYPE=Ethernet
BOOTPROTO=static
NAME=eth0
DEVICE=eth0
ONBOOT=yes
IPADDR=10.0.0.110
NETMASK=255.255.255.0
GATEWAY=10.0.0.254
DNS1=223.5.5.5
```

测试网络连通性

```
[root@clsn7 ~]# ping 223.5.5.5 -c1
PING 223.5.5.5 (223.5.5.5) 56(84) bytes of data.
64 bytes from 223.5.5.5: icmp_seq=1 ttl=128 time=94.4 ms
```

1.6 KVM虚拟机冷/热迁移

在进行迁移之前需要准备一台与KVM配置相同的机器（KVM02）,部署好kvm环境。

1.6.1 虚拟机冷迁移

在kvm02中安装kvm组件

```
[root@kvm02 ~]# yum install libvirt* virt-* qemu-kvm* -y
```

配置桥接网络

```
[root@kvm02 ~]# virsh iface-bridge eth0 br0
[root@kvm02 ~]# mkdir -p /data
```

将虚拟机关机，导出配置文件

```
[root@kvm data]# virsh dumpxml clsn7 >clsn7.xml
```

将虚拟机文件传输到kvm02上

```
[root@kvm data]# scp -rp  clsn7.xml clsn.qcow2  10.0.0.201:/data
```

导入配置文件

```
[root@kvm02 ~]# virsh define clsn7.xml
```

启动虚拟机

```
[root@kvm02 ~]# virsh start clsn7
```

查看虚拟机状态

```
[root@kvm02 ~]# virsh list --all
 Id    名称                         状态
----------------------------------------------------
 5     clsn7                          running
```

至此，一次KVM冷迁移就完成了

1.6.2 virt-manager和kvm虚拟机热迁移（准备）

实现kvm虚拟机热迁移核心：共享存储。在这里使用的时NFS共享存储，关于nfs的详情参考：http://www.cnblogs.com/clsn/p/7694456.html

1、安装virt-manager所需桌面及vnc-server

```
[root@kvm ~]# yum groupinstall "GNOME Desktop" -y
# vnc-server端
[root@kvm ~]# yum install tigervnc-server -y
# virt-manager需要软件
[root@kvm ~]# yum install openssh-askpass -y
```

2、配置vnc服务

复制vnc配置文件

```
[root@kvm ~]# vi /usr/lib/systemd/system/vncserver@.services
[root@kvm ~]# \cp /usr/lib/systemd/system/vncserver@.service  /usr/lib/systemd/system/vncserver@
```

修改配置文件，主要修改参数。

```
[root@kvm ~]# egrep -v "^#|^$" /usr/lib/systemd/system/vncserver@\:1.service
[Unit]
Description=Remote desktop service (VNC)
```

```
After=syslog.target network.target
[Service]
Type=forking
User=root
ExecStartPre=-/usr/bin/vncserver -kill %i
ExecStart=/usr/bin/vncserver %i
PIDFile=/root/.vnc/%H%i.pid
ExecStop=-/usr/bin/vncserver -kill %i
[Install]
WantedBy=multi-user.target
# 用户为root，家目录为root
```

官方提供修改方法

```
# Quick HowTo:
# 1. Copy this file to /etc/systemd/system/vncserver@.service
# 2. Replace  with the actual user name and edit vncserver
#    parameters appropriately
#   ("User=" and "/home//.vnc/%H%i.pid")
# 3. Run `systemctl daemon-reload`
# 4. Run `systemctl enable vncserver@:.service`
```

设置vnc连接时的密码，

```
[root@kvm ~]# vncpasswd
Password:
Verify:
Would you like to enter a view-only password (y/n)? n
# y为创建只读用户，n为非只读用户。
```

启动vnc服务，设置开机自启动

```
[root@kvm ~]# systemctl start vncserver@\:1.services
[root@kvm ~]# systemctl enable vncserver@\:1.services
```

查看密码文件及其他配置文件位置

```
[root@kvm ~]# ll ~/.vnc/
```

3、配置NFS存储

安装软件

```
[root@kvm ~]# yum install nfs-utils rpcbind -y
```

修改配置文件

```
[root@kvm ~]# cat /etc/exports
/data   172.16.1.0/24(rw,sync,all_squash,anonuid=0,anongid=0)
```

启动nfs程序

```
[root@kvm ~]# systemctl restart rpcbind
[root@kvm ~]# systemctl restart nfs
# 设置开机自启动
[root@kvm ~]# systemctl enable rpcbind
[root@kvm ~]# systemctl enable nfs
```

在kvm02上安装nfs

```
[root@kvm02 ~]#  yum install nfs-utils rpcbind -y
```

查看共享信息

```
[root@kvm02 ~]# showmount -e 172.16.1.240
Export list for 172.16.1.240:
/data 172.16.1.0/24
```

挂载目录

```
[root@kvm02 ~]# mount.nfs 172.16.1.240:/data /data
# 加入开机自启动
[root@kvm02 ~]# echo  'mount.nfs 172.16.1.240:/data /data' >>/etc/rc.local
[root@kvm02 ~]# chmod +x /etc/rc.d/rc.local
```

1.6.3 KVM虚拟机热迁移（实现）

vnc连接KVM宿主机：

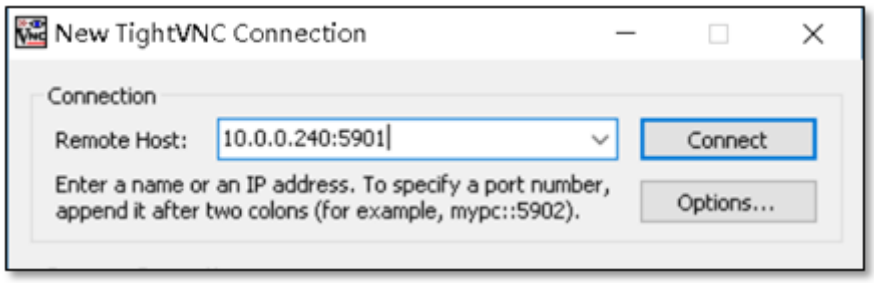


图 – 连接地址

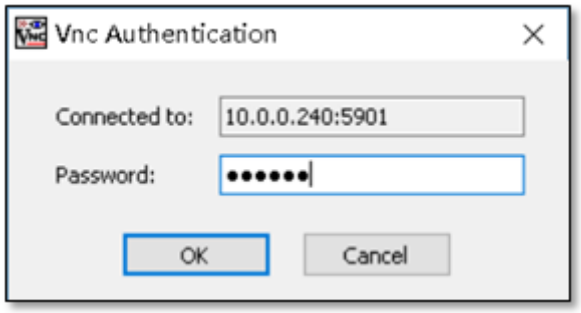


图 – 输入vnc密码

图 – 使用vmm 虚拟系统管理器

添加KVM02宿主机

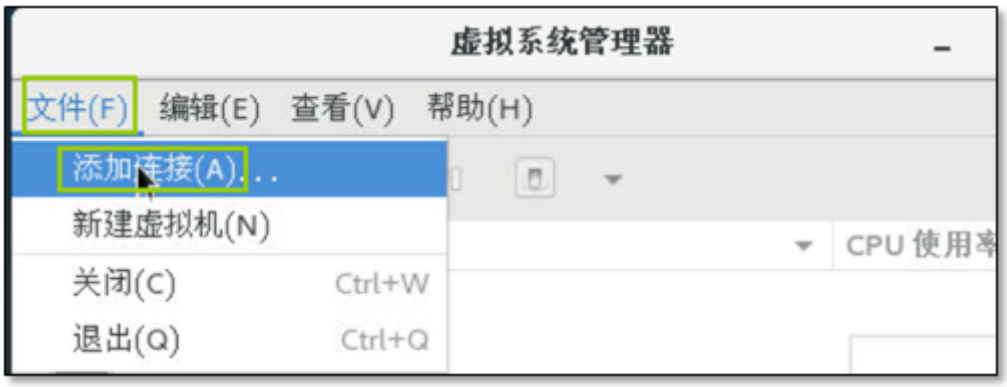


图 – 添加新连接

注：连接上KVM02机器即可

图 – 添加上kvm02主机

图 – 主机添加完成

主机热迁移

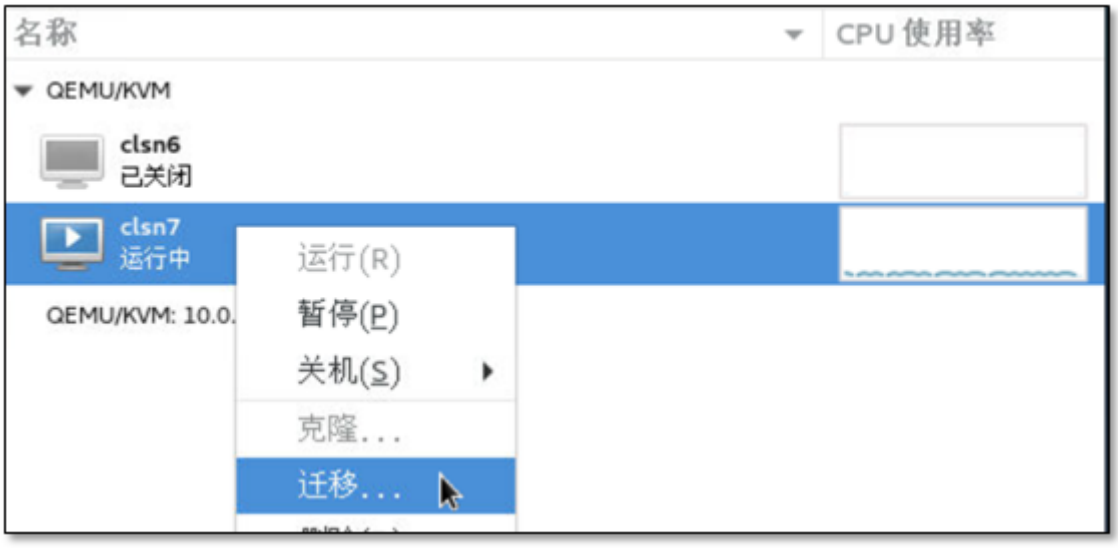


图 – 迁移1

图 – 迁移2，选择要迁移到目的主机

图 – 迁移过程

图 – 迁移完成

在kvm02上查看虚拟机状态

```
[root@kvm02 ~]# virsh list --all
 Id    名称                         状态
----------------------------------------------------
 7     clsn7                          running
```

虚拟机配置查看方法:

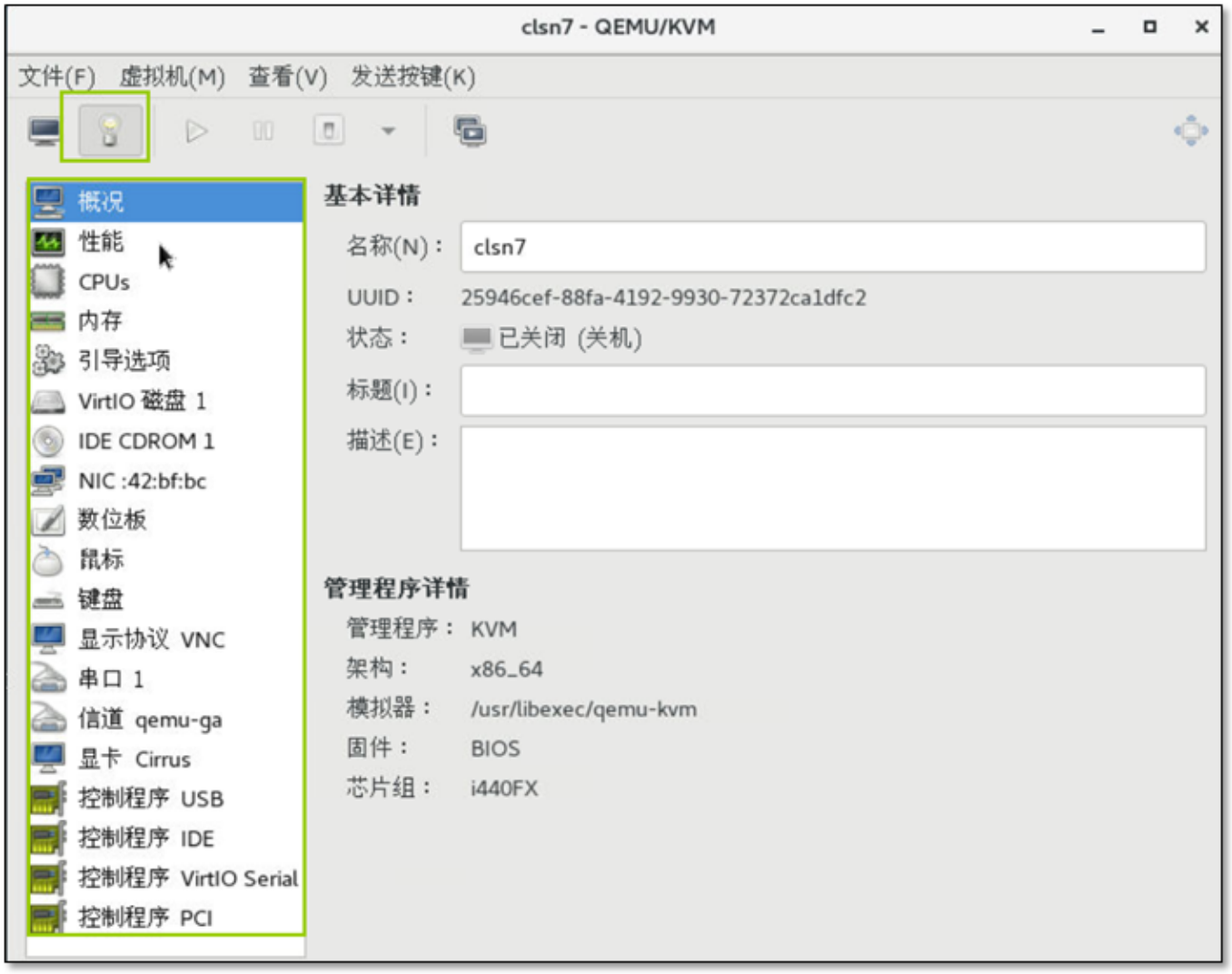


图 – clsn7 虚拟机配置信息

说明:在热迁移的过程中可能会参数丢包的情况,一般不会超过1个包。

```
[C:\~]$ ping 10.0.0.110 -t
来自 10.0.0.110 的回复: 字节=32 时间=1ms TTL=64
来自 10.0.0.110 的回复: 字节=32 时间=13ms TTL=64
来自 10.0.0.110 的回复: 字节=32 时间=11ms TTL=64
请求超时。
来自 10.0.0.110 的回复: 字节=32 时间=4ms TTL=64
来自 10.0.0.110 的回复: 字节=32 时间<1ms TTL=64
来自 10.0.0.110 的回复: 字节=32 时间<1ms TTL=64
```

至此,一次热迁移就完成了

1.7 KVM链接克隆

链接克隆脚本

```
#!/bin/bash
# kvm link clone scripts
# user clsn
# blog: https://www.nmtui.com
# 2018-02-06
###

# init
if [ $# -ne 2 ]
  then
    echo "Usage: $0 OLD_VMNAME NEW_VMNAME"
    exit 2
fi
LOG=/var/log/messages
old_vm=$1
new_vm=$2
new_xml="/tmp/${new_vm}.xml"
. /etc/init.d/functions

# dump old xmlfile
virsh dumpxml $old_vm >$new_xml
old_disk=`awk -F "'" '/source file/{print $2}'  $new_xml`
tmp_dir=`dirname $old_disk`
new_disk=${tmp_dir}/${new_vm}.qcow2

# make link disk
qemu-img create -f qcow2 -b $old_disk $new_disk &>> $LOG

# make over xml info
sed -i '/uuid/d' $new_xml
sed -i '/mac address/d' $new_xml
sed -i '2s#'$old_vm'#'$new_vm'#' $new_xml
sed -i "s#$old_disk#$new_disk#g" $new_xml
sed -i '/source mode/d' $new_xml

# import new xml file
virsh define $new_xml &>> $LOG

# start new vm
virsh start $new_vm &>> $LOG
if [ $? -eq 0 ]
  then
   action "vmhost $new_vm start"  /bin/true
else
   action "vmhost $new_vm start"  /bin/false
   echo "log info : $LOG"
fi

# END
\rm $new_xml
```

说明:

1.7.1 手动克隆

第一步：复制虚拟磁盘文件

第二步：修改xml配置文件

1）name

2）uuid

3）虚拟磁盘存储路径

4）mac地址

1.7.2脚本实现思路

```
1) 备份old_vm的配置文件，并重定向生成一个新的虚拟机配置文件
2）取出old_vm的磁盘路径
3）创建新的链接磁盘文件
4) 修改xml配置文件
5) 导入新虚拟机
6）启动测试
```

1.8 参考文献

[1] https://zh.wikipedia.org/wiki/

[2] http://virtual.51cto.com/art/201303/386133.htm

[3] https://virt-manager.org

[4] https://zh.wikipedia.org/wiki/基于内核的虚拟机

[5] https://libvirt.org

[6] [kvm] 四种简单的网络模型 https://www.cnblogs.com/hukey/p/6436211.html

[7] https://www.cnblogs.com/xieshengsen/p/6215168.html

[8] http://wiki.ubuntu.org.cn/Kvm教程

惨绿少年Linux www.nmtui.com