

云平台部署与管理

NSD CLOUD

DAY01

内容

上午	09:00 ~ 09:30	KVM简介
	09:30 ~ 10:20	
	10:30 ~ 11:20	Virsh管理
	11:30 ~ 12:00	
下午	14:00 ~ 14:50	自定义虚拟机
	15:00 ~ 15:50	
	16:10 ~ 17:00	虚拟设备管理
	17:10 ~ 18:00	总结和答疑



KVM简介

KVM简介

搭建KVM服务器

虚拟化概念

安装虚拟化服务器平台

KVM虚拟机的组成

管理KVM平台

virsh 命令

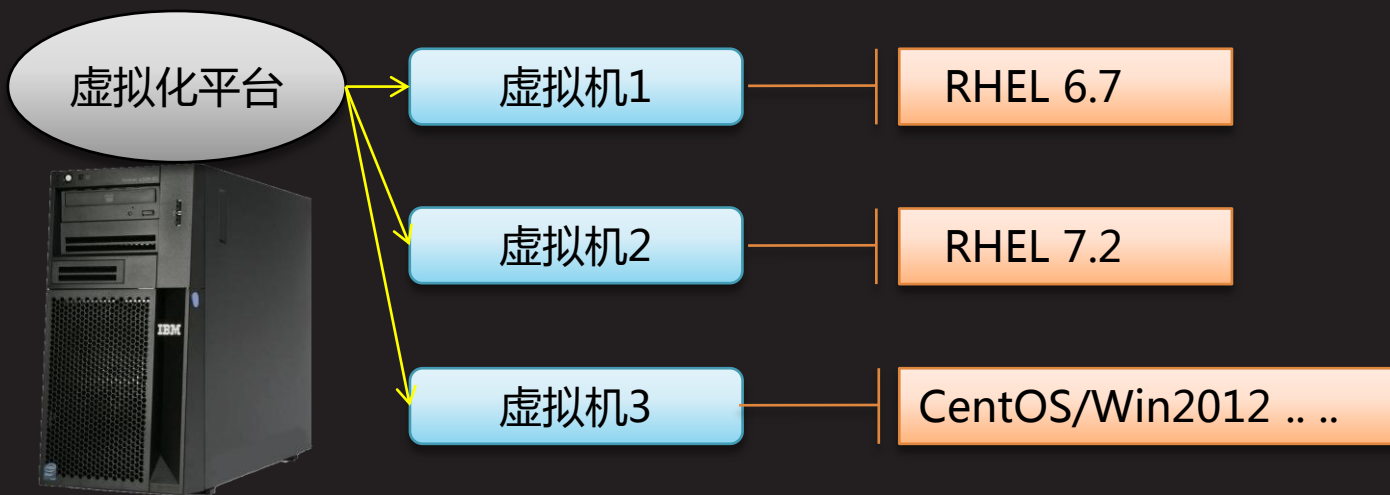
qcow2简介

qemu-img 命令

搭建KVM服务器

虚拟化概述

- virtualization 资源管理
 - x个物理资源 --> y个逻辑资源
 - 实现程度：完全、部分、硬件辅助（CPU）



虚拟化概述（续1）

- 虚拟化主要厂商及产品

系 列	PC/服务器版代表
VMware	VMware Workstation、vSphere
Microsoft	VirtualPC、Hyper-V
RedHat	KVM、RHEV
Citrix	Xen
Oracle	Oracle VM VirtualBox



安装虚拟化服务器平台

- KVM /QEMU /LIBVIRT
 - KVM是linux内核的模块，它需要CPU的支持，采用硬件辅助虚拟化技术 Intel-VT，AMD-V，内存的相关如 Intel的 EPT 和 AMD 的 RVI 技术
 - QEMU 是一个虚拟化的仿真工具，通过 ioctl 与内核 kvm 交互完成对硬件的虚拟化支持
 - Libvirt 是一个对虚拟化管理的接口和工具，提供用户端程序 virsh ,virt-install, virt-manager, virt-view 与用户交互



安装虚拟化服务器平台

- 必备软件
- qemu-kvm
 - 为 kvm 提供底层仿真支持
- libvirt-daemon
 - libvirtd 守护进程，管理虚拟机
- libvirt-client
 - 用户端软件，提供客户端管理命令
- libvirt-daemon-driver-qemu
 - libvirtd 连接 qemu 的驱动



安装虚拟化服务器平台

- 可选功能
 - virt-install # 系统安装工具
 - virt-manager # 图形管理工具
 - virt-v2v # 虚拟机迁移工具
 - virt-p2v # 物理机迁移工具

- 虚拟化平台的安装

```
yum install -y qemu-kvm \
    libvirt-daemon \
    libvirt-client \
    libvirt-daemon-driver-qemu
systemctl start libvirtd
```



虚拟机的组成

- 虚拟机的组成
 - 内核虚拟化模块 (KVM)
 - 系统设备仿真 (QEMU)
 - 虚拟机管理程序 (LIBVIRT)
 - 一个 XML 文件 (虚拟机配置声明文件)
 - 位置 /etc/libvirt/qemu/
 - 一个磁盘镜像文件 (虚拟机的硬盘)
 - 位置 /var/lib/libvirt/images/



管理KVM平台



virsh命令工具介绍

- 提供管理各虚拟机的命令接口
 - 支持交互模式，查看/创建/停止/关闭 ...
 - 格式：`virsh 控制指令 [虚拟机名称] [参数]`

```
[root@nova01 ~]# virsh
```

```
Welcome to virsh, the virtualization interactive terminal.
```

```
Type: 'help' for help with commands
```

```
'quit' to quit
```

```
virsh #
```



查看虚拟化信息

- 查看KVM节点（服务器）信息
 - `virsh nodeinfo`
- 列出虚拟机
 - `virsh list [--all]`
- 列出虚拟网络
 - `virsh net-list [--all]`
- 查看指定虚拟机的信息
 - `virsh dominfo 虚拟机名称`



开关机操作

- 运行|重启|关闭指定的虚拟机
 - `virsh start|reboot|shutdown` 虚拟机名称
- 强制关闭指定的虚拟机
 - `virsh destroy` 虚拟机名称
- 将指定的虚拟机设为开机自动运行
 - `virsh autostart [--disable]` 虚拟机名称



案例3：virsh基本管理操作

1. 列出当前正在运行的虚拟机
2. 查看虚拟机的信息
3. 启动/重启/关机/强制关机操作
4. 设置虚拟机开机自动运行



常用镜像盘类型

- 虚拟机的磁盘镜像文件格式

特点\类型	RAW	QCOW2
KVM默认	否	是
I/O效率	高	较高
占用空间	大	小
压缩	不支持	支持
后端盘复用	不支持	支持
快照	不支持	支持



qemu-img

- qemu-img 是虚拟机的磁盘管理命令
- qemu-img 支持非常多的磁盘格式，例如 raw、qcow2、vdi、vmdk 等等
- qemu-img 命令格式
 - qemu-img 命令 参数 块文件名称 大小
 - 常用的命令有
 - create 创建一个磁盘
 - convert 转换磁盘格式
 - info 查看磁盘信息
 - snapshot 管理磁盘快照



qemu-img

- 创建新的镜像盘文件
 - `qemu-img create -f 格式 磁盘路径 大小`
 - `qemu-img create -f qcow2 disk.img 50G`
- 查询镜像盘文件的信息
 - `qemu-img info 磁盘路径`
 - `qemu-img info disk.img`
- -b 使用后端模板文件
 - `qemu-img create -b disk.img -f qcow2 disk1.img`



COW技术原理

- Copy On Write, 写时复制
 - 直接映射原始盘的数据内容
 - 当原始盘的旧数据有修改时, 在修改之前自动将旧数据存入前端盘
 - 对前端盘的修改不回写到原始盘



virsh管理

KVM简介

virsh虚拟机管理

连接本地/远程

虚拟机远程管理

创建虚拟交换机

网络管理

xml管理

导出虚拟机

导入虚拟机

删除虚拟机

编辑xml设置

virsh虚拟机管理

连接本地/远程KVM

- 使用 virsh 客户端工具
 - 连接本地
 - virsh
 - virsh# connect qemu:///system （默认选项）
 - 连接远程
 - virsh# connect
qemu+ssh://user@ip.xx.xx.xx:port/system



虚拟机远程管理

- 使用 virt-manager 客户端工具
 - virt-manager 也可以通过 add connection 管理其它机器上的虚拟机，一般通过"remote tunnel over ssh"就可以了
 - 需要注意的是 virt-manager 需要使用 ssh 免密码登录如果没有免密码登录需要单独安装 python 的 ssh 相关模块，这里我们可以使用部署 key 的方法解决



创建虚拟交换机

- libvirt 网络接口
 - 原理：调用 dnsmasq 提供DNS、DHCP等功能
 - 创建配置文件 /etc/libvirt/qemu/networks/vbr.xml

```
<network>  
  <name>vbr</name>  
  <bridge name="vbr"/>  
  <forward mode="nat"/>  
  <ip address="192.168.1.254" netmask="255.255.255.0">  
    <dhcp>  
      <range start="192.168.1.100" end="192.168.1.200"/>  
    </dhcp>  
  </ip>  
</network>
```



网络管理

- virsh 管理虚拟网络
 - net-list 查看虚拟网络
 - net-define vbr.xml 创建虚拟网络
 - net-undefine vbr 删除虚拟网络
 - net-start vbr 启动虚拟网络
 - net-destroy vbr 停止虚拟网络
 - net-edit vbr 修改 vbr 网络的配置
 - net-autostart vbr 设置 vbr 虚拟网络开机自启动



课堂练习

- virsh 管理
 - 创建一个虚拟网络 vbr
 - 设置 vbr 的 ip 为 192.168.1.254
 - 配置 vbr 虚拟网络的 dhcp 分配地址范围 100-200
 - 启动 vbr 虚拟网络并用 ifconfig 验证
 - 设置 vbr 虚拟网络开机自启动



xml管理

导出虚拟机

- xml配置文件
 - 定义了一个虚拟机的名称、CPU、内存、虚拟磁盘、网卡等各种参数设置
 - 默认位于 `/etc/libvirt/qemu/虚拟机名.xml`
- 导出xml配置文件
 - 查看：`virsh dumpxml 虚拟机名`
 - 备份：`virsh dumpxml 虚拟机名 > 虚拟机名.xml`



编辑虚拟机设置

- 对虚拟机的配置进行调整
 - 编辑：`virsh edit 虚拟机名`
 - 若修改 name、memory、disk、network，可自动保存为新虚拟机配置

```
[root@kvmshr ~]# virsh edit rhel-207
<domain type='kvm'>
  <name>rhel-207</name>
  <uuid>76d5dc2c-5eef-4e30-8b6c-e58851814f84</uuid>
  <disk type='file' device='disk'>
    <source file='/var/lib/libvirt/images/rhel7.2.qcow2'/>
  .. ..
  <interface type='network'>
    <mac address='52:54:00:91:52:e4'/>
  .. ..
```



导入虚拟机

- 根据修改后的独立xml文件定义新虚拟机
 - `virsh define XML描述文件`

```
[root@kvmsvr ~]# virsh define /root/rhel-207.xml
```

定义域 rhel-207 (从 /root/rhel-207.xml)

```
[root@kvmsvr ~]# virsh list --all
```

Id	名称	状态

-	rhel-207	关闭
-	rhel7.2	关闭



删除虚拟机

- 必要时可去除多余的xml配置
 - 比如虚拟机改名的情况
 - 避免出现多个虚拟机的磁盘或MAC地址冲突
 - `virsh undefine 虚拟机名`

```
[root@kvmsvr ~]# virsh undefine rhel7.2  
域 rhel7.2 已经被取消定义
```



自定义虚拟机

自定义虚拟机

自定义虚拟机安装

网络 yum 源的安装配置

virt-manager 安装虚拟机

虚拟机模板制作

软件包安装及yum源配置

网卡及配置文件配置

Console及磁盘分区配置

去除个性化信息

自定义虚拟机安装

网络yum源的安装和配置

- 快速配置网络 yum 源
 - 配置ftp

```
yum install vsftp
```
 - 修改配置文件 /etc/vsftpd/vsftpd.conf
 - listen=YES
 - listen_ipv6=NO
 - systemctl enable vsftpd
 - systemctl start vsftpd
 - vsftp 默认根目录为 /var/ftp



网络yum源的安装和配置

- 快速配置网络 yum 源
 - 在 ftp 跟目录创建文件夹 centos7
 - 把 CentOS7 的光盘挂载到刚刚创建的目录上

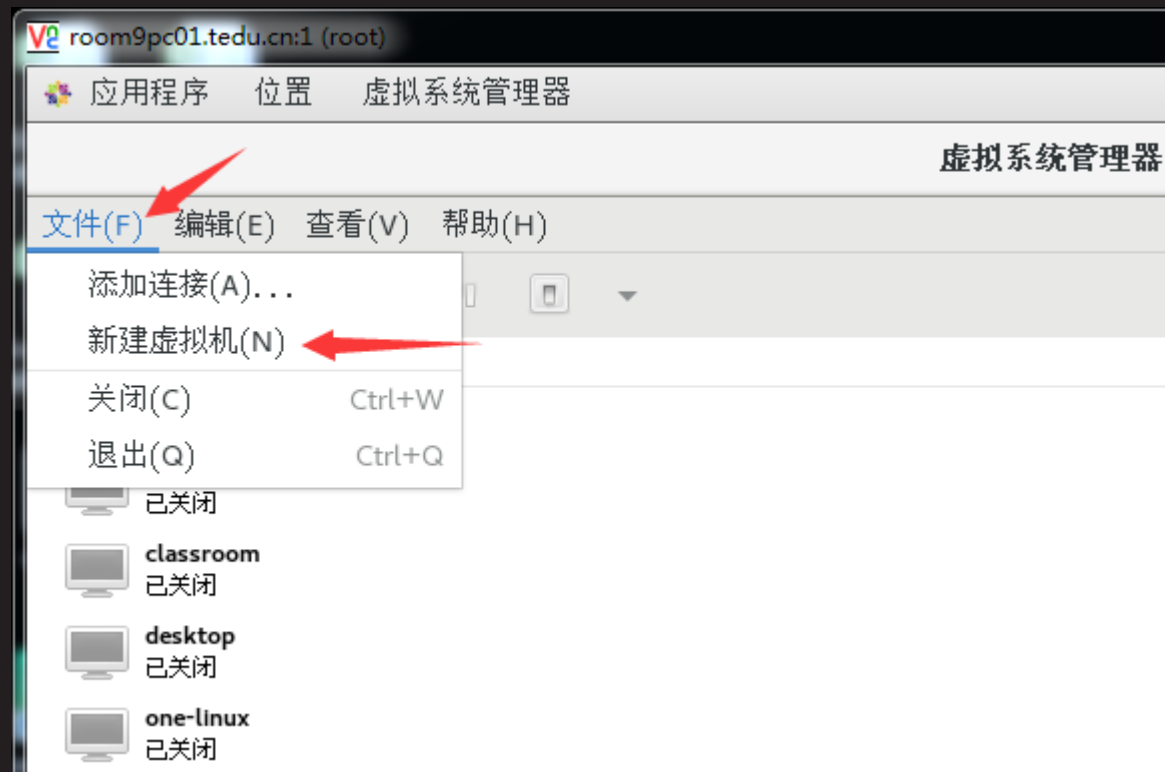
```
mount -t iso9660 -o loop,ro /xx/xx.iso /var/ftp/centos7
```
 - 在客户机里面配置 /etc/yum.repos.d/xxx.repo

```
[Centos_repo]
name= CentOS packet
baseurl=ftp://xx.xx.xx.xx/centos7
enabled=1
gpgcheck=0
```
 - yum repolist



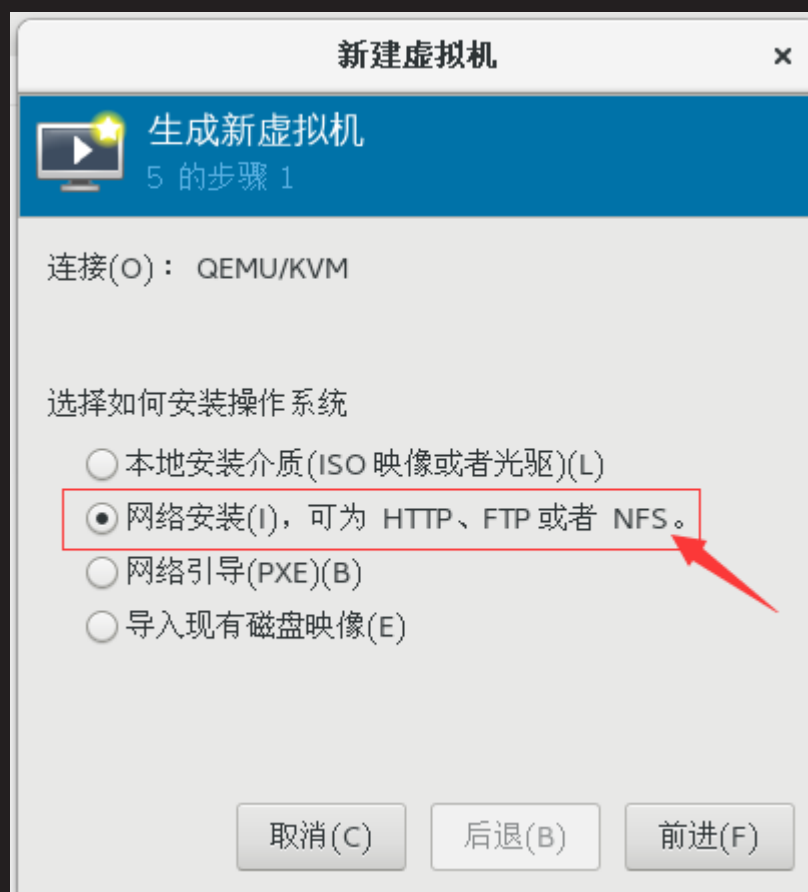
virt-manager安装虚拟机

- 启动 virt-manager 软件，选择新建虚拟机



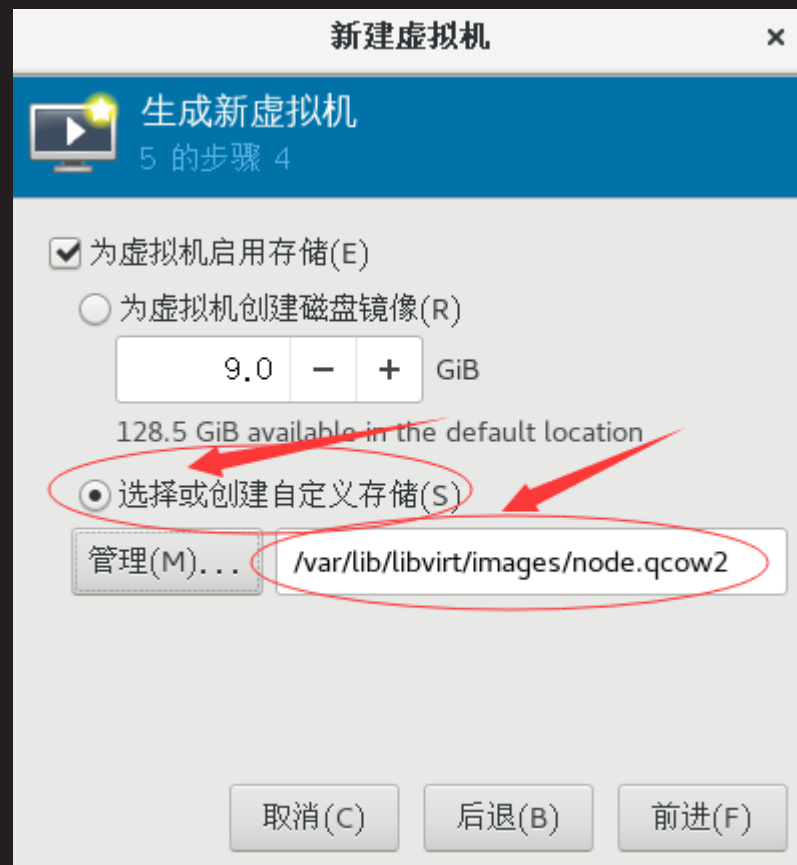
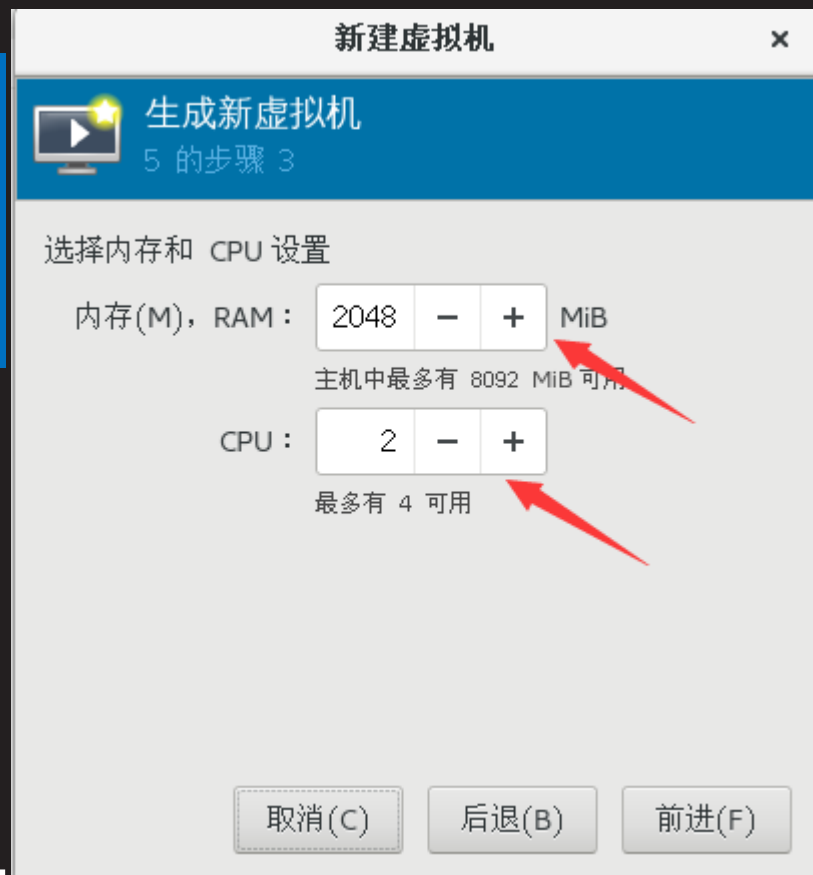
virt-manager安装虚拟机

• 选择网络安装源



virt-manager安装虚拟机

– qemu-img create -f qcow2 node.qcow2 16G



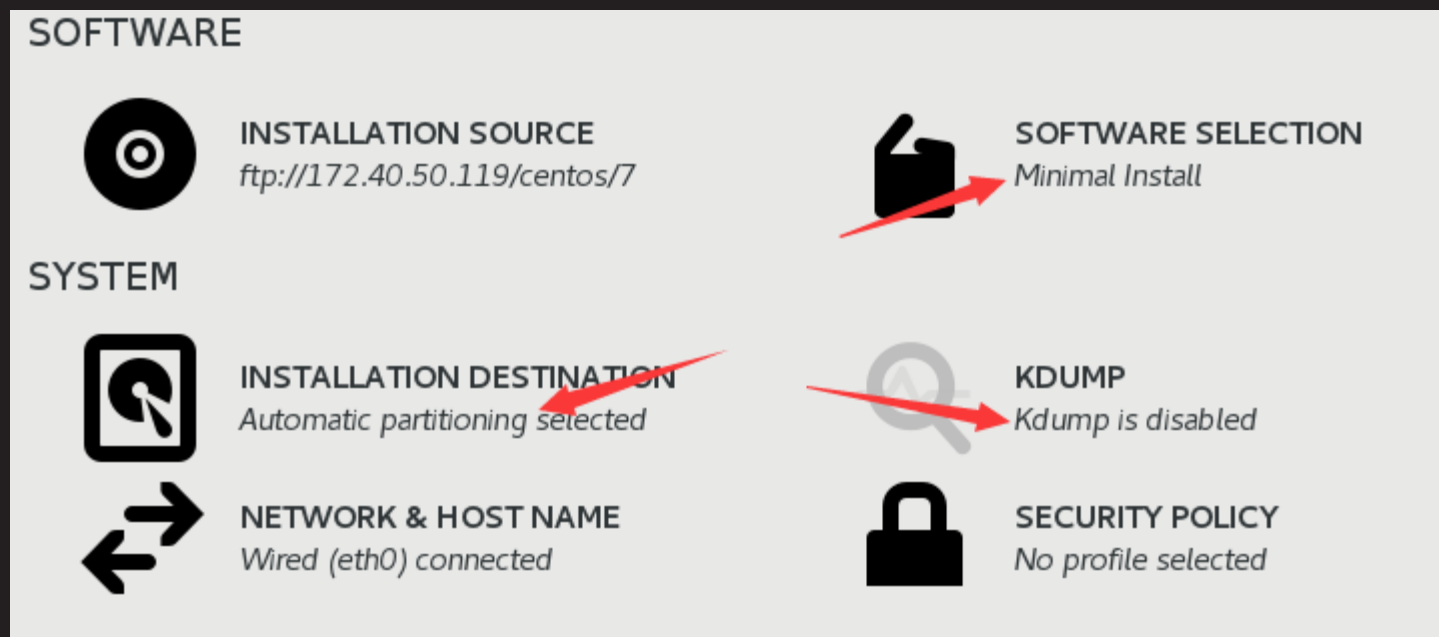
virt-manager安装虚拟机

- 网络选择自定义的 vbr



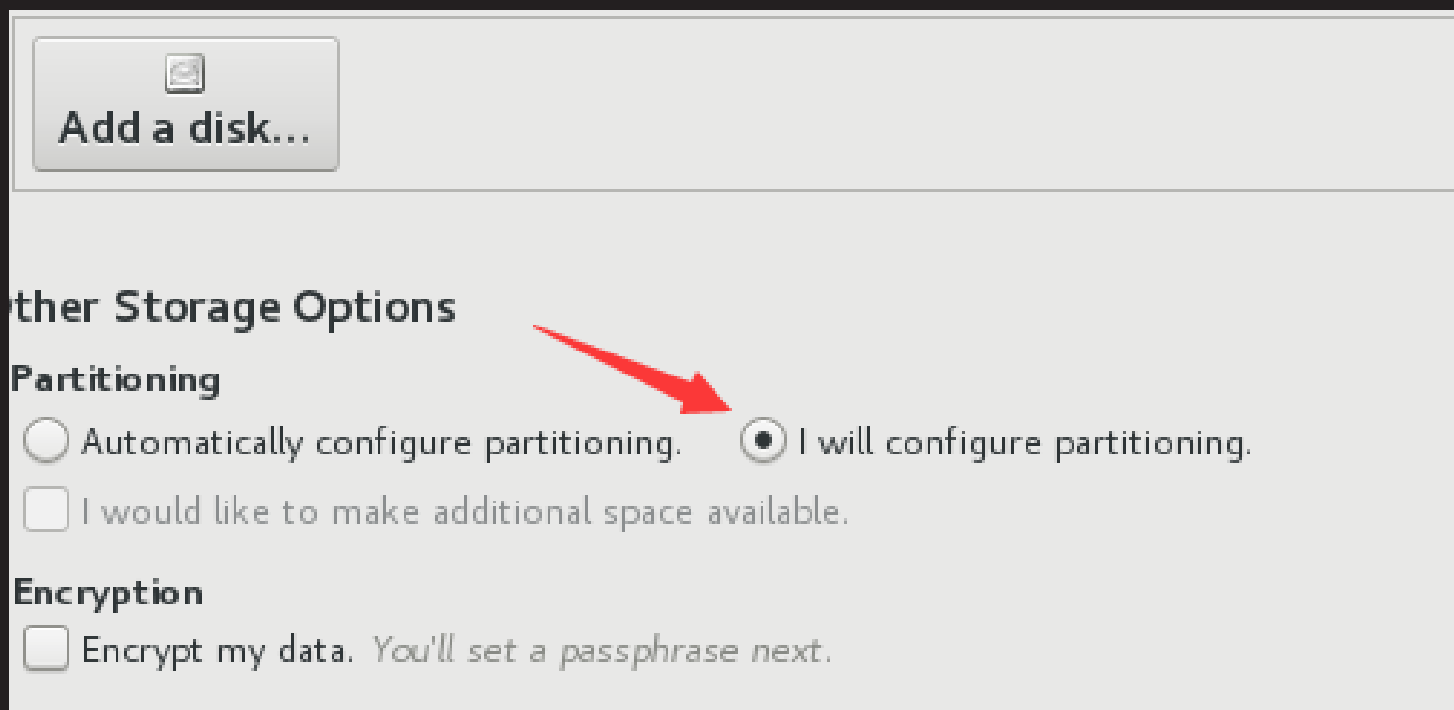
virt-manager安装虚拟机

- 软件选择 minimal , 关闭 KDUMP
- 选择手工分区



virt-manager安装虚拟机

- 手动分区



virt-manager安装虚拟机

- 使用标准分区格式，只分一个根分区

▼ New CentOS 7 Installation

You haven't created any mount points for your CentOS 7 installation yet.
You can:

- [Click here to create them automatically.](#)
- Create new mount points by clicking the '+' button.

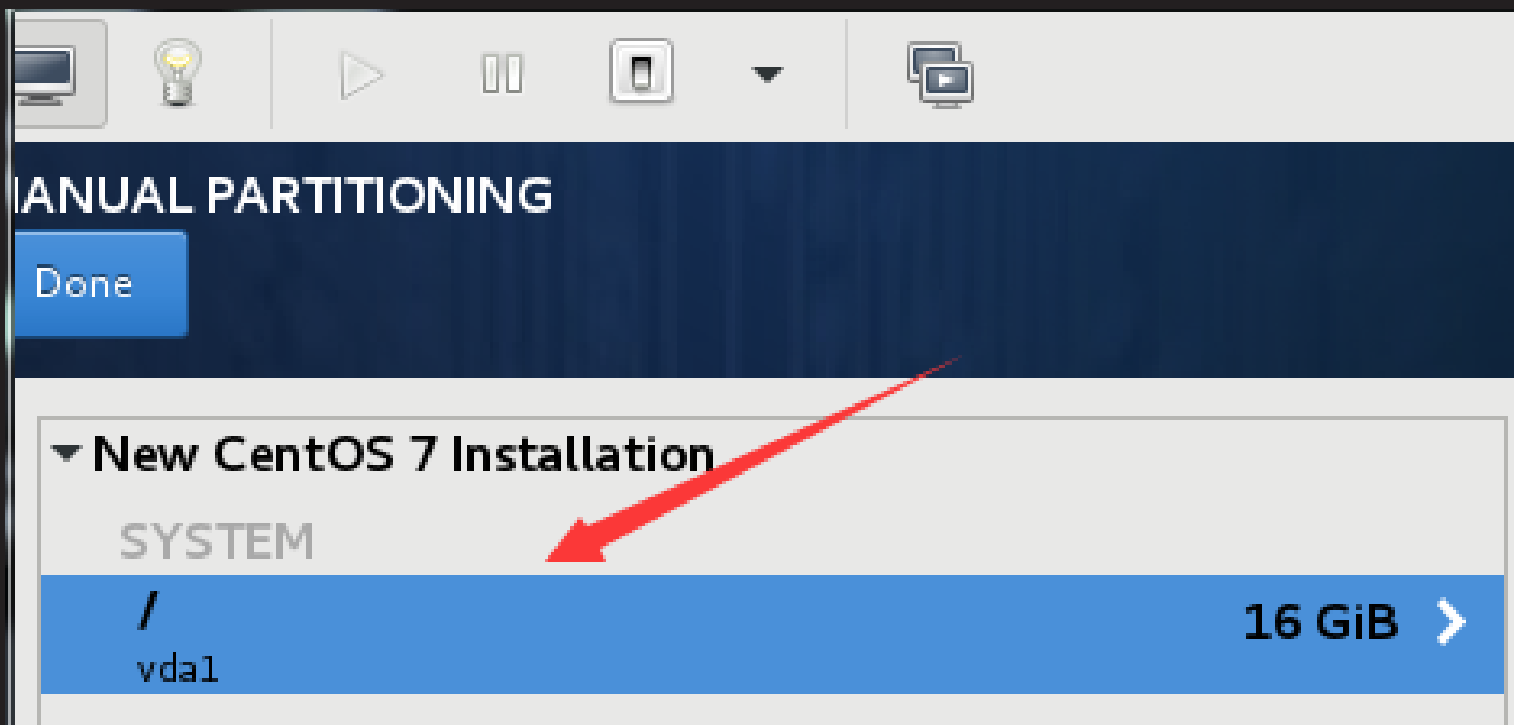
New mount points will use the following partitioning scheme:

Standard Partition ▶



virt-manager安装虚拟机

- 选取一个根分区，开启安装系统



虚拟机模板制作

软件包安装及yum配置

- 把刚刚安装好的系统初始化
 - 1、禁用 selinux /etc/selinux/config
`SELINUX=disabled`
 - 2、卸载防火墙与NetworkManager
`yum remove -y NetworkManager-* firewall-* python-firewall`
 - 3、配置 yum 源
`[local_repo]`
`name=CentOS-$releasever - Base`
`baseurl="ftp://192.168.1.254/centos7"`
`enabled=1`
`gpgcheck=0`



软件包安装及yum配置

- 续上页

- 安装软件

- `yum install -y lftp`

- 1、yum 源导入公钥验证配置

- `gpgcheck=1`

- 2、导入 gpg key

- `rpm --import ftp://192.168.1.254/centos7/RPM-GPG-KEY-CentOS-7`

- 3、常用系统命令安装

- `yum install -y net-tools vim-enhanced bridge-utils psmisc`



网卡及配置文件设置

- 删除网络配置里的个性化信息
 - /etc/sysconfig/network-scripts/ifcfg-eth0

```
DEVICE="eth0"
ONBOOT="yes "
IPV6INIT="no"
TYPE="Ethernet"
BOOTPROTO= "dhcp "
```
 - 禁用空路由
 - /etc/sysconfig/network

```
NOZEROCONF="yes"
```



Console及磁盘分区配置

- 添加 Console 配置及删除磁盘分区里的个性化信息
 - /etc/default/grub

```
GRUB_CMDLINE_LINUX="biosdevname=0 net.ifnames=0  
console=ttyS0,115200n8"  
GRUB_DISABLE_LINUX_UUID="true"  
GRUB_ENABLE_LINUX_LABEL="true "
```
 - 重新生成 grub.cfg

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```
 - /etc/fstab 文件中到 UUID 手工修改成系统设备
 - blkid 查看 uuid 对应的磁盘设备，修改 fstab 文件



去除个性化信息

- 安装分区扩展软件

```
yum install -y cloud-utils-growpart
```

- 设置第一次开机自动扩容根目录

```
chmod 755 /etc/rc.local
```

- 在 rc.local 里加入如下配置

```
###
```

```
/usr/bin/growpart /dev/vda 1
```

```
/usr/sbin/xfs_growfs /
```

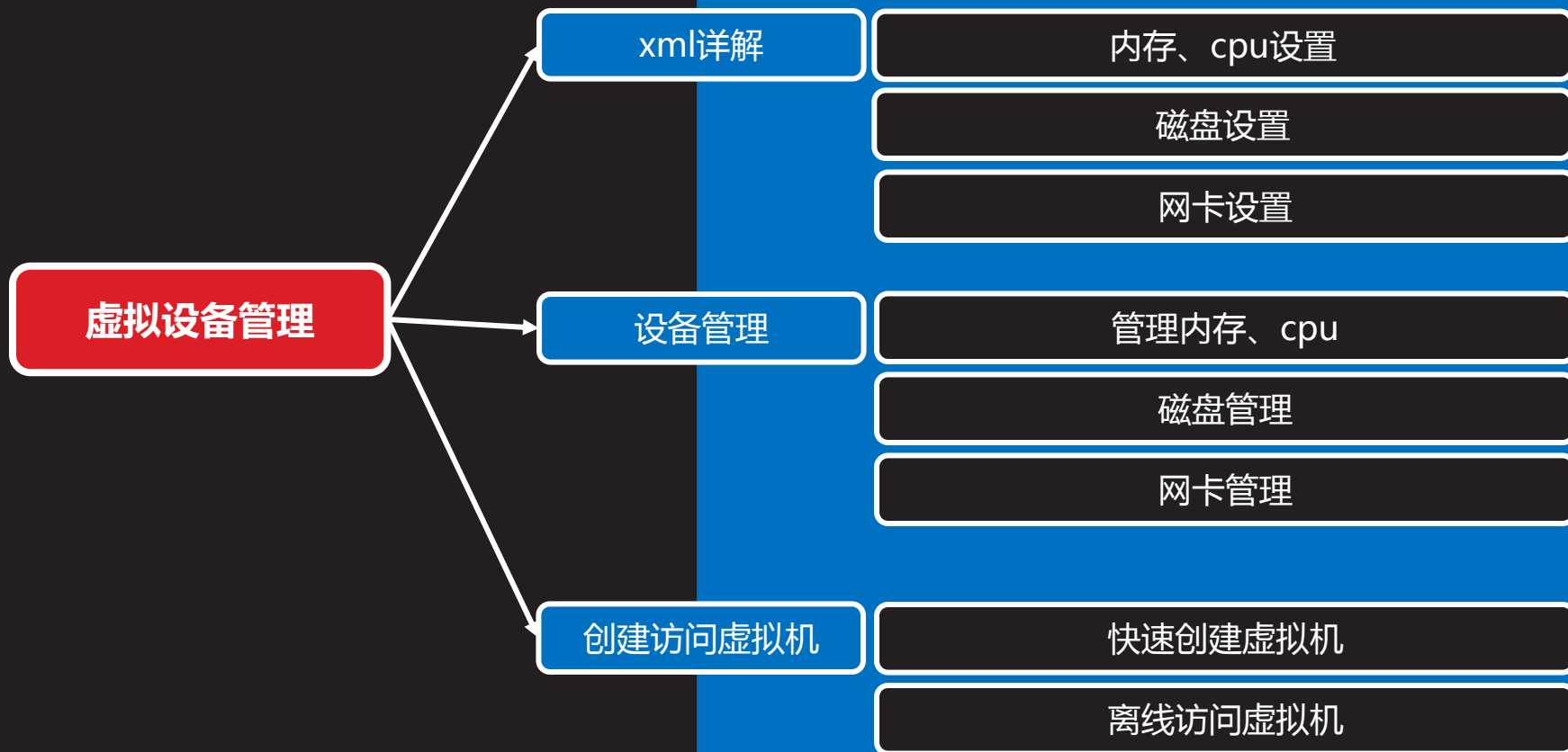
```
/usr/bin/sed '/^###/, $d' -i /etc/rc.d/rc.local
```

- 关闭虚拟机后执行信息清理工作

```
virt-sysprep -d node
```



虚拟设备管理



xml详解

XML详解

- XML 配置文件

- 保存 node 虚拟机配置文件

```
virsh dumpxml node >demo.xml
```

- 清除模板配置

```
virsh undefine node
```

- xml 模板

- <!-- 是注释的开始

- --> 是注释的结尾

- xml 标签必须成对出现

- <keyword> </keyword>



XML详解

- XML 配置文件

```
<domain type='kvm' id='4'>
  <name>centos7.0</name>
  <uuid>8413f30a-c978-4796-a28f-8ba9fe74b759</uuid>
  <memory unit='KiB'>2097152</memory>
  <currentMemory unit='KiB'>2097152</currentMemory>
```

- id=4 个性化设置，去掉
- uuid 去掉
- memory unit 虚拟机最大使用内存，可以手动调整
- currentmemory 创建虚拟机使用内存



XML详解

- XML 配置文件

- cpu 设置，可以调整

`<vcpu placement='static'>2</vcpu>`

- 资源设置，可以删除

`<resource> </resource>`

- 系统配置及引导设备，不需要修改

`<os>`

`<boot dev='hd'/>`

`</os>`

- 电源相关配置无需修改

`<features> </features>`



XML详解

- XML 配置文件
 - cpu配置，可以修改成使用真机cpu


```
<cpu ... ... />
<cpu mode='host-passthrough'> </cpu>
```
 - 时钟相关配置，可以删除


```
<clock ... ... </clock>
```
 - 重启，关机，强制关闭对应命令


```
<on_poweroff>destroy</on_poweroff>
<on_reboot>restart</on_reboot>
<on_crash>destroy</on_crash>
```



XML详解

- XML 配置文件
 - 内存及硬盘休眠相关设置，可以删除
`<pm> </pm>`
 - 仿真设备相关配置
`<devices> </devices>`
 - 其他配置
 - 驱动安全标签，可删除
`<seclabel> </seclabel>`



设备管理



XML详解

- 仿真设备配置

- 总线地址，别名配置，都可以删除

`<address`

`<alias ...`

- 硬盘配置，需要修改

`<emulator>/usr/libexec/qemu-kvm</emulator>`

`<disk </disk>`

- usb 相关设备配置，可以删除

`<controller type='usb'`



XML详解

- 仿真设备配置
 - type='pci' pci总线设备，可删除
 - type='virtio-serial' 串口设备需要保留串口终端
 - type='network' 网络配置需要修改
 - type='pty' 串口终端，需要保留
 - type='unix' 虚拟机通讯控制接口
 - type='spicevmc' 图形图像相关配置可以删除
 - type='tablet' 数位板，可以删除



XML详解

- 仿真设备配置
 - type='mouse' 鼠标，保留
 - type='keyboard' 键盘保留
 - graphics、video 图形图像显卡相关配置，可以删除
 - sound 声卡相关配置，可以删除
 - redirdev 设备重定向，可以删除
 - memballoon 内存气泡，可以动态调整内存



创建访问虚拟机

快速创建虚拟机

- 如何快速创建虚拟机
 - 1、 xml 文件配置
 - 把我们的 xml 模板文件进行复制
 - 修改名称及磁盘文件
 - 2 img 文件创建
 - 以我们刚刚装系统的模板为后端文件创建虚拟机
- ```
qemu-img create -b node.qcow2 -f qcow2 node1.img
```



# 快速创建虚拟机

- 如何快速创建虚拟机
  - 完成虚拟机注册及启动虚拟机
  - `virsh define node1.xml`
  - `virsh start node1`
  - `virsh console node`



# 配置虚拟机系统有奇招

- 使用 **guestmount** 工具
    - 支持离线挂载 raw、qcow2 格式虚拟机磁盘
    - 可以在虚拟机关机的情况下，直接修改磁盘中的文档
    - 方便对虚拟机定制、修复、脚本维护
- !!! 需要注意 SELinux 机制的影响





# 如何挂载虚拟盘

- 基本用法

`guestmount -a 虚拟机磁盘路径 -i /挂载点`

```
[root@kvmsvr ~]# mkdir /mnt/kdisk
[root@kvmsvr ~]# guestmount -a node1.qcow2 -i /mnt/kdisk
[root@kvmsvr ~]# ls /mnt/kdisk
bin home media opt sbin tmp
boot lib misc proc selinux usr
...
```

