第 3 章

创建虚拟机

第 2 章讲解了 KVM 相关组件的安装，本章将讲解虚拟机的创建。在 RHEL/CentOS 8 中创建虚拟机的方法主要有 3 种：Cockpit、virt-manager 和 virt-install。其中前两种方法是基于 GUI 的，virt-install 则是在命令行进行安装。

新用户总喜欢 GUI 而不喜欢命令行，因此我们会先从 Cockpit、virt-manager 开始来创建虚拟机，但同时也会尽可能地通过 virsh 来观察背后所发生的变化，这样才能深入地掌握所学习的知识。

本章要点：

- 使用 Cockpit 创建虚拟机。
- 使用 virt-manager 创建虚拟机。
- 使用 virt-install 创建虚拟机。
- 半虚拟化驱动 VirtIO。
- QEMU Guest Agent。
- 显示设备与协议原理。
- SPICE Agent。

3.1 使用 Cockpit 创建虚拟机

按 Fedora 社区和红帽公司的规划，将来会逐步用 Cockpit 替代 virt-manager 来管理虚拟机。虽然目前 Cockpit 的功能还比不上 virt-manager 丰富，但是基于 Web 的管理模式是大势所趋。

3.1.1 查看当前配置

在创建虚拟机之前，我们先通过 Cockpit 和 virsh 来查看当前的配置。

Cockpit 虚拟化管理的主界面中会显示虚拟机列表、存储池的数量及虚拟网络的数量，如图 3-1 所示。目前系统没有虚拟机，但是有一个默认的存储池和虚拟网络。

图 3-1　Cockpit 虚拟化管理的主界面

什么是存储池？libvirt 通过存储池(Storage Pool)和卷(Volume)在宿主机上提供存储管理。存储池是管理员(通常是专门的存储管理员)留出的一定空间的存储，供虚拟机使用。存储池由存储管理员或系统管理员划分为多个存储卷，并将这些卷作为块设备分配给虚拟机。在本书的第 6 章会详细讲解这些知识。

单击“存储池”链接，就会出现一个存储池清单，如图 3-2 所示。这个名为 default 的存储池的信息如下：

图 3-2　Cockpit 中的存储池

（1）存储名称：default。

（2）状态：已激活。

（3）已分配：3.85GB。

（4）容量：49.98GB。

（5）目标路径：/var/lib/libvirt/images。

（6）持久：是。

（7）自动启动：是。

（8）类型：dir。

virsh 是管理 KVM 虚拟化最主要的管理工具。它功能强大，除 KVM 之外，还可以管理 Xen、LXC、OpenVZ、VirtualBox 和 VMware ESXi/ESX 等多种虚拟化平台。

virsh 有很多种子命令，其中与存储池有关的子命令通常以 pool-开头。先简单了解一下 3 个子命令。

（1）pool-list：列出当前的存储池。默认仅显示处于激活状态的存储池。

（2）pool-info：返回特定存储池对象的基本信息。

（3）pool-dumpxml：返回特定存储池对象的详细信息，输出格式为 XML。

查看存储池 default 的命令如下：

```
#virsh pool-list
Name          State    Autostart
-------------------------------------------
default        active    yes

#virsh pool-info default
Name:         default
UUID:         b111d4c0-374c-4d8f-bf58-1050e4af953d
State:        running
Persistent:   yes
Autostart:    yes
Capacity:     49.98 GiB
Allocation:   3.85 GiB
Available:    46.13 GiB

#virsh pool-dumpxml default
<pool type='dir'>
  <name>default</name>
  <uuid>b111d4c0-374c-4d8f-bf58-1050e4af953d</uuid>
  <capacity unit='Bytes'>53660876800</capacity>
  <allocation unit='Bytes'>4129992704</allocation>
  <available unit='Bytes'>49530884096</available>
  <source>
  </source>
  <target>
```

```
    <path>/var/lib/libvirt/images</path>
    <permissions>
      <mode>0711</mode>
      <owner>0</owner>
      <group>0</group>
      <label>system_u:object_r:virt_image_t:s0</label>
    </permissions>
  </target>
</pool>
```

类似地，我们再使用 Cockpit 和 virsh 对照着查看虚拟网络的信息。现在，在 Cockpit 的虚拟网络中仅显示了一个名为 default 的虚拟网络，如图 3-3 所示。它的主要信息如下：

（1）名称：default。

（2）设备：virbr0。

（3）转发模式：NAT。

（4）状态：激活。

（5）IP 地址：192.168.122.1。

（6）DHCP 范围：192.168.122.2-192.168.122.254。

图 3-3　Cockpit 中的虚拟网络

virsh 中与宿主机网络有关的子命令通常以 net-开头。先简单了解一下 3 个子命令。

（1）net-list：列出当前的网络，默认仅显示处于激活状态的网络。

（2）net-info：返回特定网络对象的基本信息。

（3）net-dumpxml：返回特定网络对象的详细信息，输出格式为 XML。

查看虚拟网络 default 的命令如下：

```
#virsh net-list
Name         State    Autostart    Persistent
----------------------------------------------
default      active   yes          yes

#virsh net-info default
Name:          default
UUID:          52959885-1cb7-425e-ae2d-4de2f98f02bc
Active:        yes
Persistent:    yes
Autostart:     yes
Bridge:        virbr0

#virsh net-dumpxml default
<network>
  <name>default</name>
  <uuid>52959885-1cb7-425e-ae2d-4de2f98f02bc</uuid>
  <forward mode='nat'>
    <nat>
      <port start='1024' end='65535'/>
    </nat>
  </forward>
  <bridge name='virbr0' stp='on' delay='0'/>
  <mac address='52:54:00:fd:b2:60'/>
  <ip address='192.168.122.1' netmask='255.255.255.0'>
    <dhcp>
      <range start='192.168.122.2' end='192.168.122.254'/>
    </dhcp>
  </ip>
</network>
```

3.1.2 创建虚拟机

如果在嵌套虚拟化环境做实验，则建议使用 32 位版的操作系统以减少资源开销。下面将创建、安装一个 32 位版 CentOS 的虚拟机。

提示：由于从版本 7 开始，CentOS 就不再提供 32 位版的 ISO 文件了，所以我们下载 CentOS 6 的 32 位版本，例如 CentOS 6.10 的最小化版本。

从 CentOS 官方网站下载 CentOS-6.10-i386-minimal.iso，并保存在/iso 目录中，命令如下：

```
#mkdir /iso

#cd /iso

# wget \
http://mirrors.163.com/centos/6.10/isos/i386/CentOS-6.10-i386-minimal.iso

#ls -l
total 364544
-rw-r--r--. 1 root root 373293056 Jun 30 2018 CentOS-6.10-i386-minimal.iso

#file CentOS-6.10-i386-minimal.iso
CentOS-6.10-i386-minimal.iso: DOS/MBR boot sector; partition 1 : ID=0x17, active, start
-CHS (0x0,0,1), end-CHS (0x163,63,32), startsector 0, 729088 sectors
```

在虚拟机页面中单击“创建虚拟机”按钮，就会出现一个“创建新的虚拟机”窗口，如图 3-4 所示。

图 3-4　在 Cockpit 中创建新的虚拟机

创建新的虚拟机，需要提供这样一些选项：

1. 名称

虚拟机的名称，例如 centos6.10。

2. 安装类型

有这样几种选项：

(1) 下载 OS：从 Cockpit 的操作系统仓库中下载操作系统。

(2) 本地安装介质：使用已经下载好的安装介质。

(3) URL：从操作系统安装介质树的 URL 进行安装。

(4) 网络引导(PXE)：通过网络引导进行安装。

本次实验将选中“本地安装介质”，然后在安装源中指定/iso/目录中已经下载好的 ISO 文件。

3. 操作系统

指定要安装的操作系统类型。从下拉列表中选择 CentOS 6.10。

4. 存储来源及大小

从下拉列表中选择“创建新卷”选项。指定磁盘大小，保持默认的 9GiB 即可。

5. 内存

指定内存大小，保持默认的 1GB 即可。

选中“立即启动 VM”，然后单击“创建”按钮。这样便会创建一个新的虚拟机并启动，然后会自动切换到“控制台”界面，如图 3-5 所示。

图 3-5　Cockpit 中虚拟机的控制台

由于此时默认的引导次序是安装介质优先，所以会启动 CentOS 6.10 的安装程序。这时，要确保将“图形控制台(VNC)”作为控制台类型。接下来，就可以根据具体的提示来安装操作系统了。

安装结束后，单击 Reboot 按钮重新启动虚拟机，如图 3-6 所示。

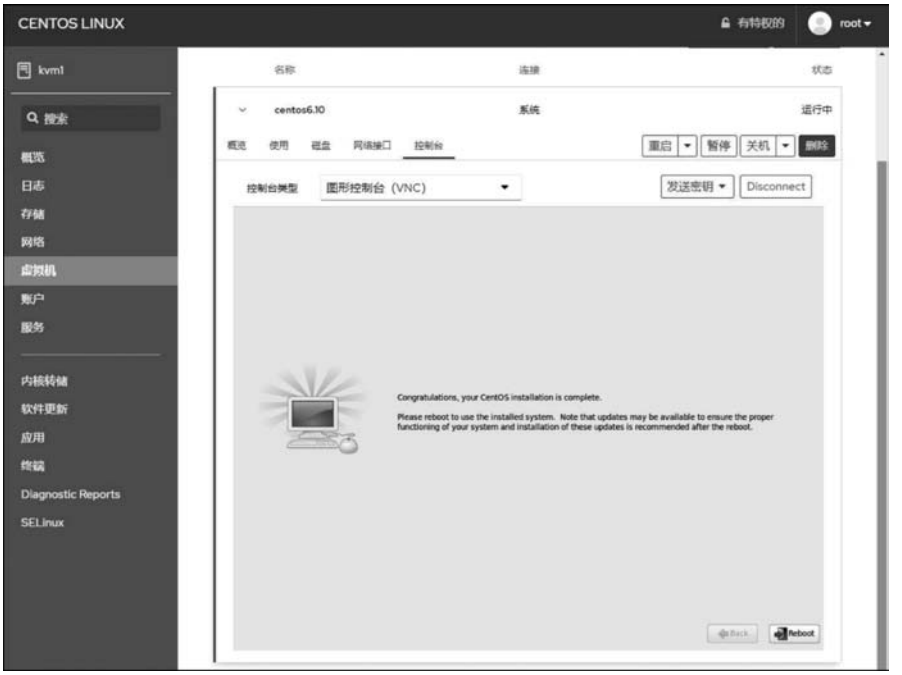


图 3-6　虚拟机安装完毕

3.1.3　查看虚拟机与环境的配置

下面，我们再通过 Cockpit 和 virsh 命令查看虚拟机与环境的配置。

在 Cockpit 的虚拟机主界面中会看到新创建的虚拟机，如图 3-7 所示。

图 3-7　虚拟机列表

virsh 的 list 子命令也可以获得虚拟机列表，登录命令如下：

```
#virsh list --all
Id    Name                State
----------------------------------------------------
1     centos6.10          running
```

执行命令后会列出全部虚拟机。如果不使用--all 选项，则仅列出已启动的虚拟机。在 libvirt 中，虚拟机对应的术语是域(domain)。

在输出中，ID 是虚拟机标识，Name 是虚拟机的名称。在 virsh 命令中，既可以通过 ID 也可以通过 Name 来引用虚拟机。State 是虚拟机的运行状态，共分为 7 种状态：

(1) running：运行中。

(2) idle：空闲。

(3) paused：已暂停。

(4) in shutdown：正在关机。

(5) shut off：已关闭。

(6) crashed 崩溃。

(7) pmsuspended 暂停。

在 Cockpit 中，单击虚拟机名称左边的“˅”链接，会显示此虚拟机的概要信息，如图 3-8 所示。

图 3-8 虚拟机的详细信息

对应的 virsh 子命令是 dominfo。在使用时，需要为 dominfo 子命令提供虚拟机的 ID 或名称，命令如下：

```
#virsh dominfo 1
Id:             1
Name:           centos6.10
UUID:           edcba2b1 - afa9 - 4de5 - 8de4 - b42cc1bc66c9
OS Type:        hvm
State:          running
CPU(s):         2
CPU time:       40.7s
Max memory:     1048576 KiB
Used memory:    1048576 KiB
Persistent:     yes
Autostart:      disable
Managed save:   no
Security model: seLinux
Security DOI:   0
Security label: system_u:system_r:svirt_t:s0:c168,c621 (enforcing)
```

提示：virsh 的子命令名称及格式很有规律，例如：dom 是 domain 的缩写，info 是 information 的缩写，掌握这个规律可以大大提高效率。

与 Cockpit 中“使用”选项卡对应的子命令是 domstats，stats 是 statistics 的缩写，示例命令如下：

```
#virsh domstats 1
```

与 Cockpit 中“磁盘”选项卡对应的子命令是 domblklist，blk 是 blocks 的缩写，示例命令如下：

```
#virsh domblklist 1
```

与 Cockpit 中“网络接口”选项卡对应的子命令是 domiflist，if 是 interfaces 的缩写，示例命令如下：

```
#virsh domiflist 1
```

提示：可以通过 virsh help 命令来获得联机帮助，例如 virsh help domiflist。另外，virsh 命令有类似于 Bash Shell 的自动补全功能，使用 Tab 键可以对子命令、选项和参数进行自动补全。

查看完虚拟机，下面查看宿主机的存储。

在 Cockpit 中，单击 default 存储池中的“存储卷”，会发现有一个新的存储卷，这就是新

创建的虚拟机的虚拟磁盘，如图 3-9 所示。

图 3-9　存储池 default 中的存储卷

对应的 virsh 子命令是 vol-list，在使用时需要提供存储池的标识，命令如下：

```
#virsh vol-list default
Name                 Path
-----------------------------------------------------
centos6.10.qcow2     /var/lib/libvirt/images/centos6.10.qcow2
```

在 Cockpit 中，还会发现 1 个名为 iso 的新存储池，如图 3-10 所示。

在这个存储池中有 1 个存储卷，就是在安装 centos6.10 虚拟机时使用的 ISO 文件，如图 3-11 所示。

virsh 子命令 pool-list 和 pool-info 可以获得存储池的列表和特定存储池的信息，命令如下：

```
#virsh pool-list
Name          State        Autostart
-------------------------------------------
default       active       yes
iso           active       yes

#virsh pool-info iso
Name:         iso
UUID:         51c3aa8a-6df2-4a46-b121-81bd44f04298
```

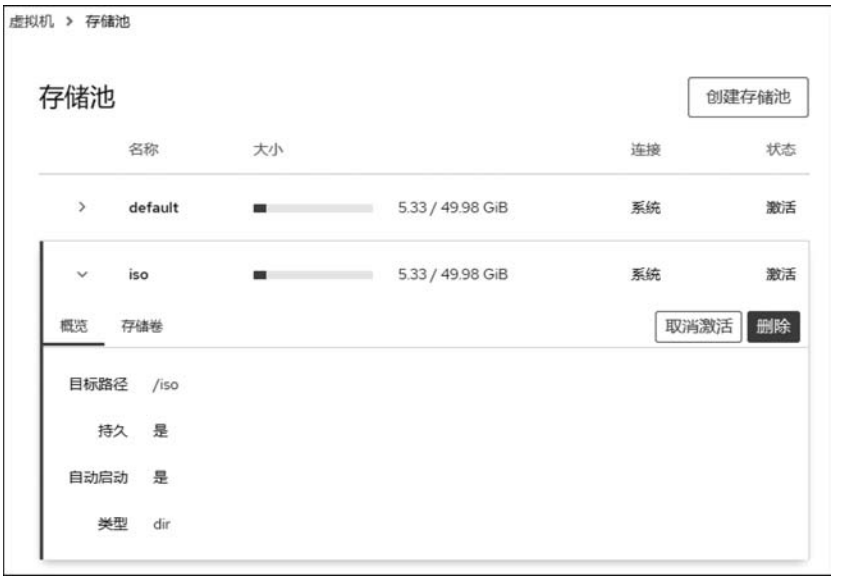


图 3-10　新增加的存储池

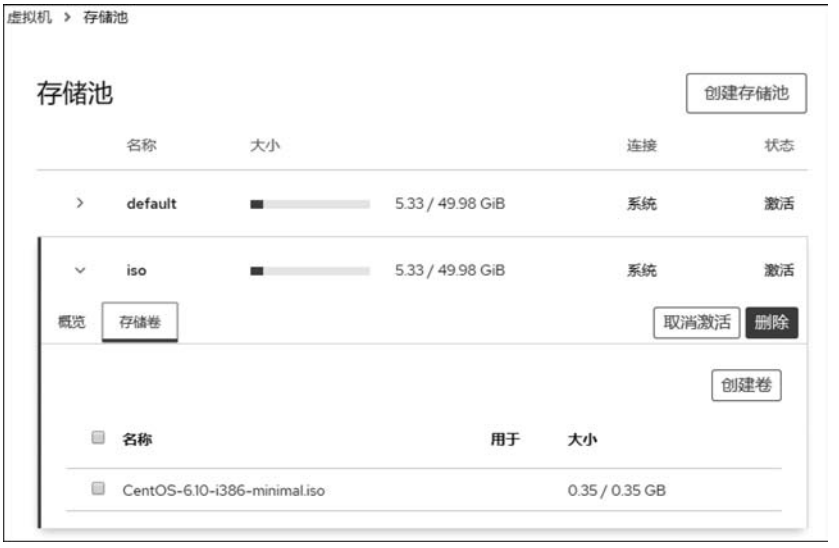


图 3-11　存储池 iso 中的存储池

```
State:          running
Persistent:     yes
Autostart:      yes
Capacity:       49.98 GiB
Allocation:     5.33 GiB
Available:      44.65 GiB
```

virsh 子命令 vol-list 可以查看指定存储池中的存储卷列表。在使用时需要指定存储池的名称，命令如下：

```
#virsh vol-list iso
Name                 Path
------------------------------------------------------------
CentOS-6.10-i386-minimal.iso /iso/CentOS-6.10-i386-minimal.iso
```

最后，我们使用 nmcli 观察宿主机上网络连接的变化，命令如下：

```
#nmcli connection show
NAME     UUID                                  TYPE        DEVICE
ens32    0b1638a6-add5-4057-9bce-575efc3d5bf2  ethernet    ens32
virbr0   fc81f0db-11eb-4471-84b6-3bb6b0c44f7b  bridge      virbr0
vnet0    27ebc3da-dc3d-4a9c-8880-6ce52af0cf8f  tun         vnet0
```

在输出中，我们会看到新增了一个名为 vnet0 的网络设备，它与虚拟机的网卡相连接。

3.2 使用 virt-manager 创建虚拟机

虚拟机管理器 virt-manager 是一个管理宿主机和虚拟机的 GUI 管理工具。虽然 RHEL/CentOS 8 还包含这个软件，但是会在将来用 Cockpit 替换它。可是目前比较“尴尬”的地方是 Cockpit 功能并不完整，有些功能只能通过 virt-manager 或 virsh 来完成，所以我们还必须掌握 virt-manager 的使用。

3.2.1 使用 virt-manager 查看当前配置

第 2 章介绍了在 VNC、XRDP 和 X-Window 等 3 种环境中启动 virt-manager 的方法，不管采用哪种方法启动，都会先看到一个虚拟机列表，如图 3-12 所示。

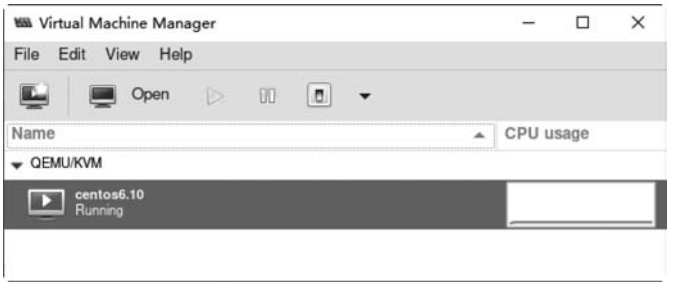


图 3-12 virt-manager 的主界面

双击虚拟机的名称或者单击工具栏中的图标，就会显示此虚拟机控制台和细节信息，可以在控制台中管理这台虚拟机，如图 3-13 所示。

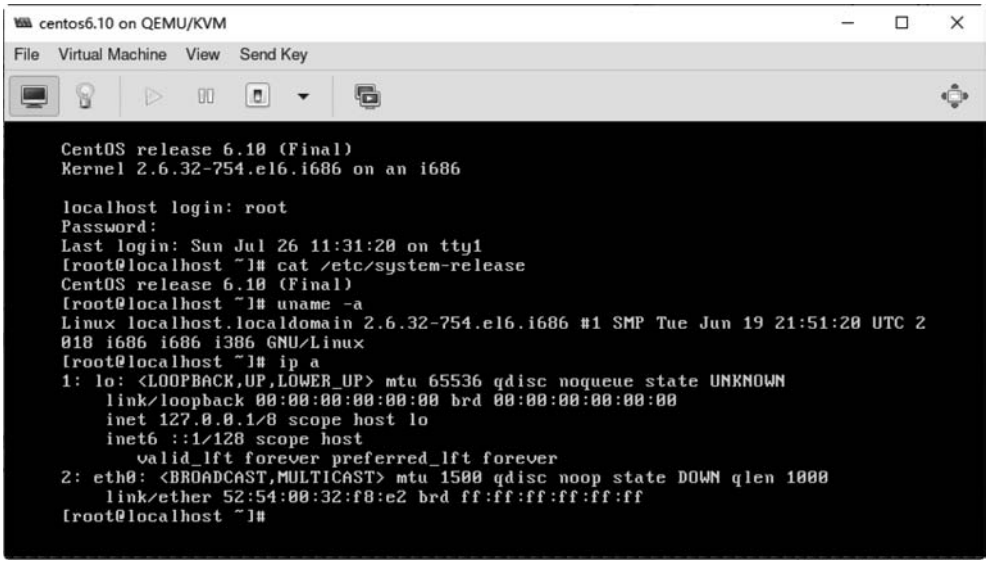


图 3-13　virt-manager 中的虚拟机控制台

单击工具栏中的图标，会显示此虚拟机的详细配置，可以在这里查看和修改虚拟机的配置，如图 3-14 所示。

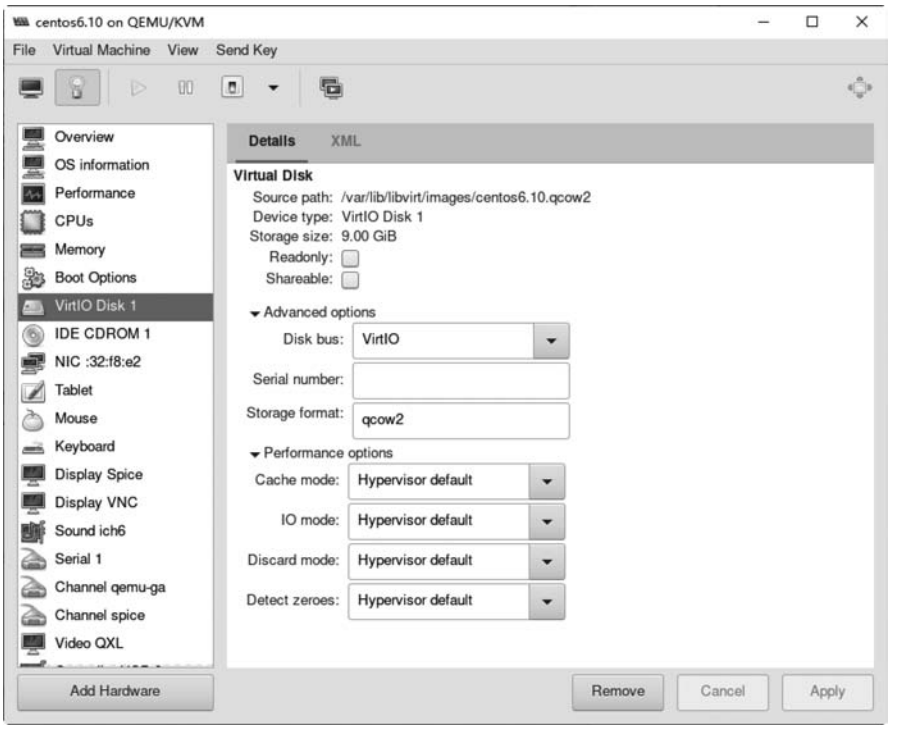


图 3-14　virt-manager 中的虚拟机配置管理

在 virt-manager 主界面中单击 Edit 菜单中的 Connection Details，如图 3-15 所示，就会显示宿主机的配置，包括整体运行状态、虚拟网络和存储，如图 3-16～图 3-18 所示。

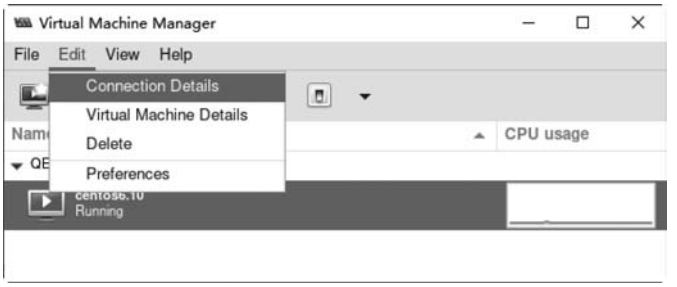


图 3-15　在 virt-manager 中打开连接的细节

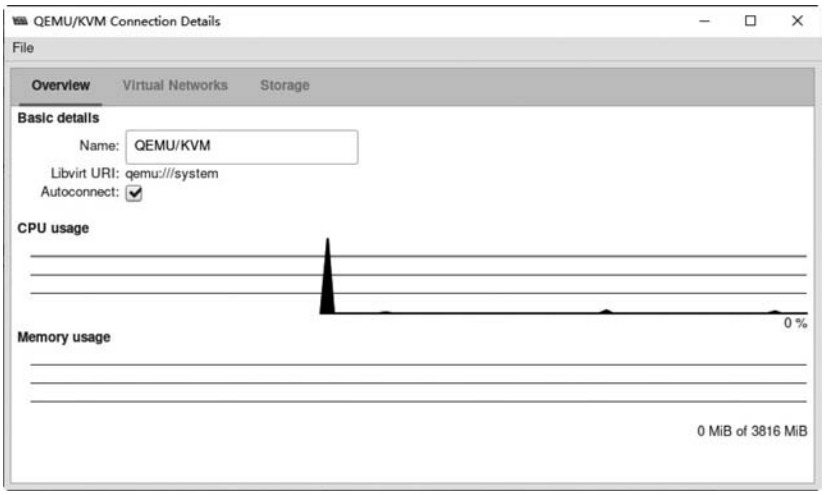


图 3-16　宿主机的整体运行状态

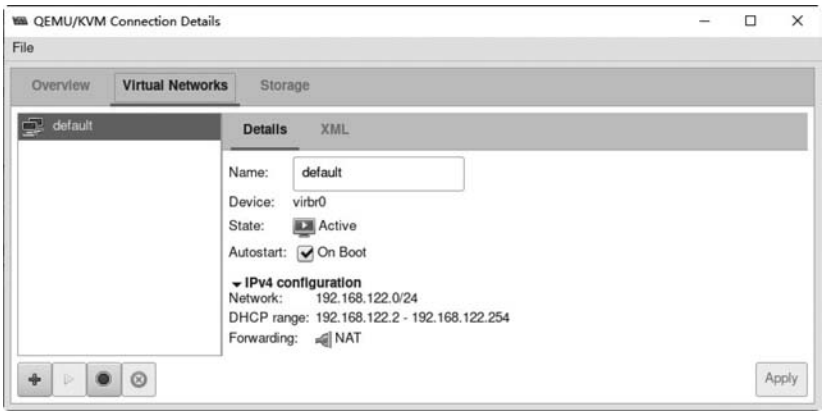


图 3-17　宿主机的虚拟网络

图 3-18　宿主机的存储池与存储卷

3.2.2　创建虚拟机

与前面的实验类似，为了减少资源开销我们将安装一个 32 位版的 Windows Server 2003 虚拟机。首先将 ISO 文件通过 WinSCP 类软件上传到宿主机/iso 目录中，如图 3-19 所示。

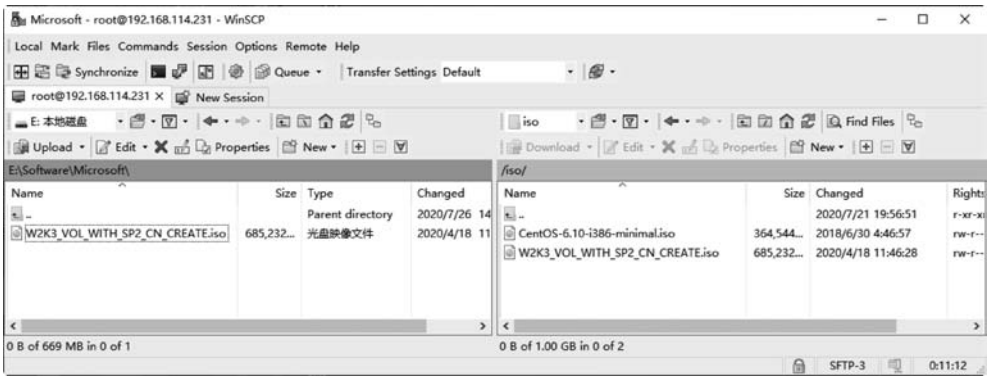


图 3-19　将 Windows Server 2003 的 ISO 文件上传到宿主机的/iso 目录中

单击 virt-manager 的 File 菜单中的 New Virtual Machine 或工具栏中的图标，就会出现一个名为 New VM 的向导。它将虚拟机的创建过程分为 5 步：

(1) 选择安装类型。

(2) 查找和配置安装介质。

(3) 配置内存和 CPU 选项。

(4) 配置虚拟机的存储。

(5) 配置虚拟机名称、网络、体系结构和其他硬件设置。

第 1 步需要在 4 种安装操作系统的方式中选择一种,如图 3-20 所示。

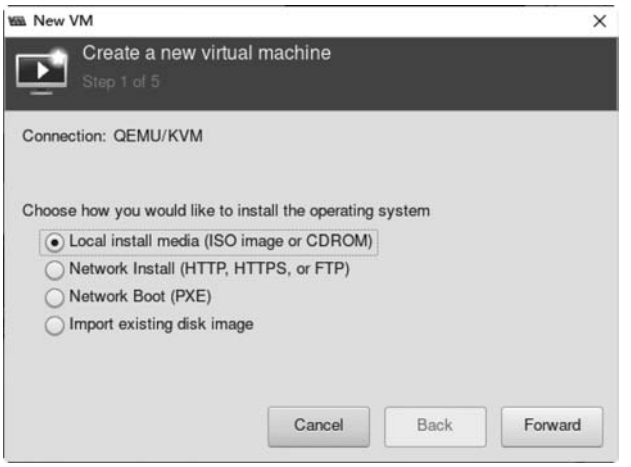


图 3-20 选择安装方式

1) Local install media (ISO image or CDROM)

此方法使用 ISO 格式的映像文件。虽然写有 CDROM,但是目前还是无法通过宿主机上的 CD-ROM 或 DVD-ROM 设备进行安装。

2) Network Install (HTTP, HTTPS, or FTP)

通过保存在 HTTP、HTTPS 或 FTP 服务器上的操作系统安装文件来安装。如果选择此项,还需要提供安装文件的 URL 及内核选项。

3) Network Boot (PXE)

采用预引导执行环境(PXE)服务器来安装虚拟机。

4) Import existing disk image

创建新的虚拟机,并将现有的磁盘映像(包含预安装的可引导操作系统)导入该虚拟机。

现在我们选中 Local install media (ISO image or CDROM)并单击 Forward 按钮继续。

第 2 步需要查找和配置安装介质,如图 3-21 所示。单击 Browse 按钮打开 Choose Storage Volume 窗口,如图 3-22 所示。

由于之前通过 Cockpit 创建虚拟机时已经创建过一个名为 iso 的存储池,所以可以在其中找到 Windows Server 20003 的 ISO 文件,然后单击 Choose Volume 按钮,将返回 New VM 向导。

接下来,需要设置操作系统类型和版本信息。为虚拟机设置适当的操作系统类型及版本,是其流畅工作的前提。如果选中了 Automatically detect from the installation media/source 复选框,则向导会自动检测安装介质并设置操作系统类型与版本,如图 3-23 所示。如果没有检测出或者检测的结果不正确,我们还可以手动指定。单击 Forward 按钮继续。

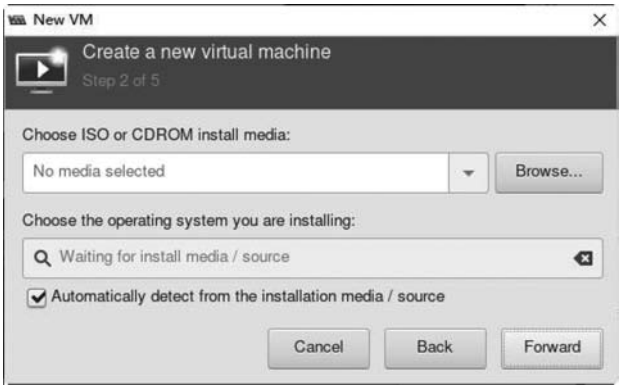


图 3-21　从本地 ISO 映像安装

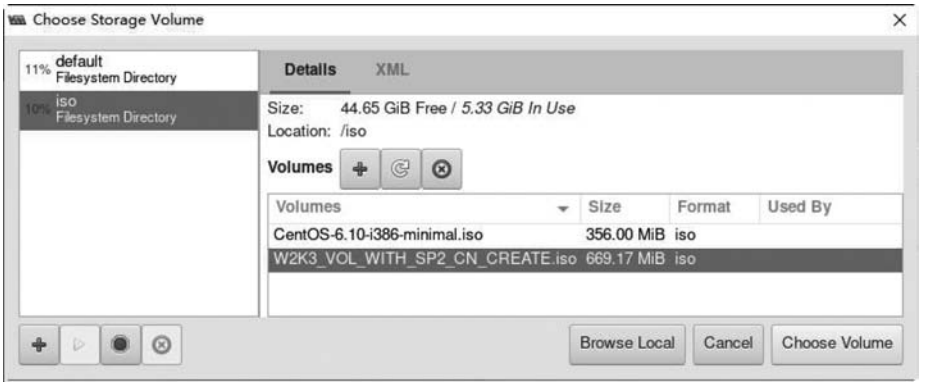


图 3-22　选择存储卷

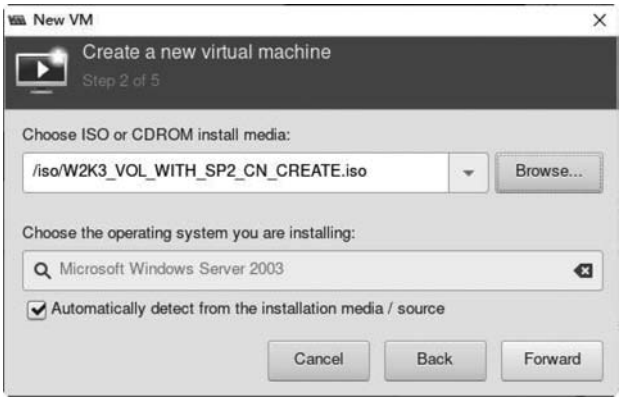


图 3-23　从本地 ISO 映像安装

第 3 步需要配置内存和 CPU 选项，这些值会影响宿主机和虚拟机的性能。向导提供默认的内存大小和 CPU 数量，我们也可以根据实际的需求进行调整，如图 3-24 所示。单击 Forward 按钮继续。

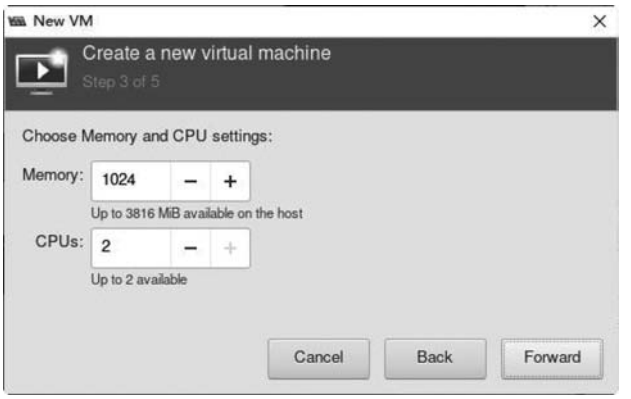


图 3-24　配置虚拟机的内存和 CPU

第 4 步需要配置虚拟机的存储，如图 3-25 所示。选中 Enable storage for this virtual machine 可以为新虚拟机启用存储，然后选中 Create a disk image for the virtual machine 单选按钮，会在宿主机上创建新磁盘映像文件，准确来讲就在那个名为 default 的存储池所对应的/var/lib/libvirt/images/目录中创建新的映像文件。根据需要来指定文件的大小，本次实验保持默认磁盘映像的大小。单击 Forward 按钮。

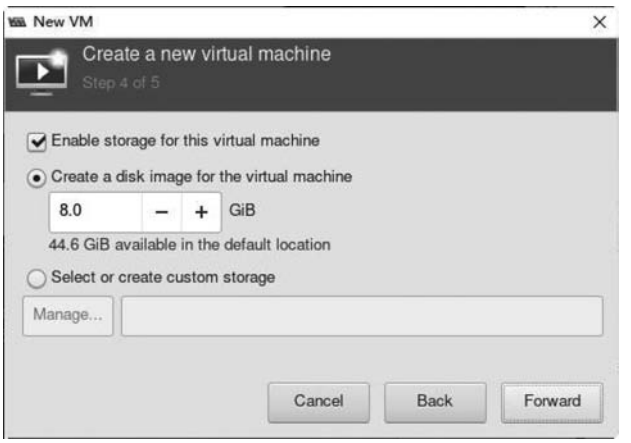


图 3-25　配置虚拟机的存储

第 5 步需要设置虚拟机的名称和检查最终配置，如图 3-26 所示。虚拟机名称可以包含字母、数字、下画线、句点和连字符，建议起一个有意义的虚拟机名称。默认情况下，新虚拟机将使用那个名为 default 的虚拟网络。

检查并验证一下虚拟机的设置，单击 Finish 按钮，这将创建具有指定网络设置、虚拟化类型和体系结构的虚拟机。

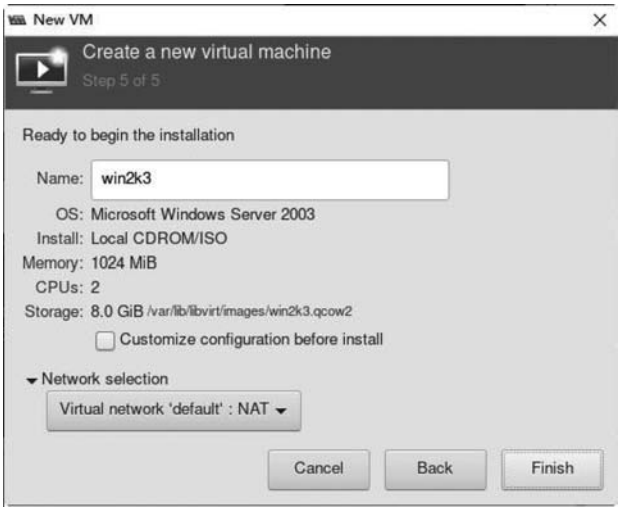
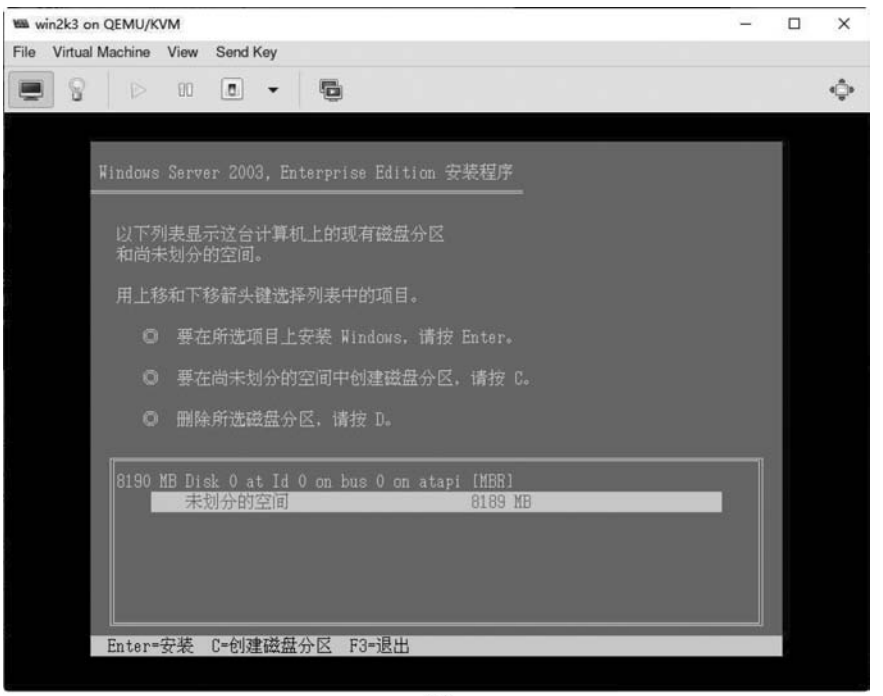


图 3-26　设置虚拟机的名称、网络并审核配置

在虚拟机中安装 Windows Server 2003 操作系统与物理机上安装过程类似，如图 3-27(a)和图 3-27(b)所示。



(a)

图 3-27　Windows Server 2003 虚拟机的安装

(b)

图 3-27 （续）

3.2.3 查看虚拟机与环境的配置

下面使用 virsh 命令查看虚拟机与环境的配置。查看当前虚拟机的列表，命令如下：

```
#virsh list
Id    Name                           State
----------------------------------------------------
1     centos6.10                     running
3     win2k3                         running
```

查看虚拟机 w2k3 的信息，命令如下：

```
#virsh dominfo win2k3
Id:             3
Name:           win2k3
UUID:           b547eefa-37af-42af-97c2-730f9ac5288a
OS Type:        hvm
State:          running
CPU(s):         2
CPU time:       1276.9s
Max memory:     1048576 KiB
```

```
Used memory:              1048576 KiB
Persistent:               yes
Autostart:                disable
Managed save:             no
Security model:           seLinux
Security DOI:             0
Security label: system_u:system_r:svirt_t:s0:c437,c725 (enforcing)
```

查看存储池 ISO 中的存储卷，命令如下：

```
#virsh vol - list iso
Name            Path
-----------------------------------------------------------
CentOS - 6.10 - i386 - minimal.iso /iso/CentOS - 6.10 - i386 - minimal.iso
W2K3_VOL_WITH_SP2_CN_CREATE.iso /iso/W2K3_VOL_WITH_SP2_CN_CREATE.iso
```

查看存储池 default 中的存储卷，命令如下：

```
#virsh vol - list default
Name                  Path
-----------------------------------------------------------
CentOS6.10.qcow2      /var/lib/libvirt/images/CentOS6.10.qcow2
win2k3.qcow2          /var/lib/libvirt/images/win2k3.qcow2
```

查看宿主机上网络连接的信息，命令如下：

```
#nmcli connection show
NAME    UUID                                   TYPE      DEVICE
ens32   0b1638a6 - add5 - 4057 - 9bce - 575efc3d5bf2   ethernet  ens32
virbr0  c847ab96 - 1add - 47f2 - b365 - d2a882eab9f1   bridge    virbr0
vnet0   e606acdb - d7e4 - 475e - 929c - f234a8164ffd   tun       vnet0
vnet1   8873aeef - 12d4 - 4e79 - a788 - 49df0fe17ccf   tun       vnet1
```

从输出的信息中可以看出来，新的网络连接名称很有规律，第 1 个虚拟机是 vnet0，第 2 个虚拟机是 vnet1，以此类推。

3.3 使用 virt-install 创建虚拟机

与前面两种方法相比，通过 virt-install 来创建新虚拟机是效率最高的方法，同时也是最复杂的方法。

通过 virt-install 来创建虚拟机需要满足两个先决条件：

1. 可访问的保存在本地或网络上的操作系统安装源

可以是以下其中之一：

(1) 安装介质的 ISO 映像文件。

(2) 现有操作系统的虚拟磁盘映像。

2. 可实现的安装模式

(1) 如果是交互式安装,则需要 virt-viewer 软件。

(2) 如果是非交互式安装,就需要为操作系统安装程序提供回答文件,例如 kickstart 文件。

3.3.1 创建虚拟机并通过交互模式安装

使用 virt-install 创建虚拟机,必须提供以下参数。

(1) --name: 虚拟机的名称。

(2) --memory: 虚拟机的内存。

(3) --vcpus: 虚拟机虚拟 CPU(vCPU)的数量。

(4) --disk: 虚拟机磁盘类型和大小。

(5) 操作系统安装源的类型和位置,可以由--location、--cdrom、--pxe、--import 和--boot 选项来指定。

查看 virt-install 命令的帮助信息,命令如下:

```
#virt-install --help
```

查看 virt-install 选项属性的完整列表,命令如下:

```
#virt install --option=?
```

例如,查看磁盘存储和选项的命令如下:

```
#virt-install --disk=?
```

当然,最完整的帮助信息在 virt-install 手册页中,除了有每个命令选项之外,还有重要的提示和丰富的示例。

在运行 virt-install 之前,有可能还需要使用 qemu-img 命令来配置存储选项。有关 qemu-img 的使用,可参见后续的章节。

下面,我们使用 virt-install 命令在交互模式下安装一台 CentOS 6.10 的虚拟机。首先,要保证 virt-viewer 命令可以正常地启动。

```
#virt-viewer -V
virt-viewer version 7.0-9.el8 (OS ID: rhel8)
```

提示: 如果系统中没有安装 virt-viewer,则可以通过命令 dnf -y install virt-viewer 进行安装。

下面准备创建了一个名为 centos6.10vm2 的虚拟机，给虚拟机分配了 1024MB 内存、1 个 vCPU、8GB 虚拟磁盘。将通过存储在本地/iso/目录中的 CentOS-6.10-i386-minimal.iso 在虚拟机中安装操作系统。另外，通过--os-variant 指定虚拟机操作系统的版本为 centos6.10。示例命令如下：

```
#virt-install --name centos6.10vm2 \
--memory 1048 --vcpus 1 --disk size=8 \
--cdrom /iso/centos-6.10-i386-minimal.iso \
--os-variant centos6.10

Starting install...
Allocating 'centos6.10vm2.qcow2'          | 8.0 GB 00:00
```

由于仅指定虚拟磁盘的大小而未指定其存储位置和名称，所以 virt-install 会自动在默认的 default 存储池中创建一个名为 CentOS6.10vm2.qcow2 的存储卷，然后分配给虚拟机。

当虚拟创建完毕之后，virt-install 会自动启动 virt-viewer，然后就可以在其中通过交互模式进行安装了，如图 3-28 所示。

图 3-28　virt-viewer 是一个可与虚拟机交互的重要界面

3.3.2 查看虚拟机与环境的配置

安装完成之后，查看一下虚拟机的信息及宿主机存储的变化。

```
#virsh list
Id    Name                 State
----------------------------------------------------
4     centos6.10vm2        running

#virsh dominfo centos6.10vm2
Id:             4
Name:           centos6.10vm2
UUID:           e81c4a75-1d80-4c68-9951-bd16038b8bec
OS Type:        hvm
State:          running
CPU(s):         1
CPU time:       21.1s
Max memory:     1048576 KiB
Used memory:    1048576 KiB
Persistent:     yes
Autostart:      disable
Managed save:   no
Security model: seLinux
Security DOI:   0
Security label: system_u:system_r:svirt_t:s0:c206,c238 (enforcing)

#virsh domblklist CentOS6.10vm2
Target     Source
------------------------------------------------------
vda        /var/lib/libvirt/images/centos6.10vm2.qcow2
hda        -

#virsh vol-list default
Name                 Path
-----------------------------------------------------------------------------
centos6.10.qcow2     /var/lib/libvirt/images/centos6.10.qcow2
centos6.10vm2.qcow2  /var/lib/libvirt/images/centos6.10vm2.qcow2
win2k3.qcow2         /var/lib/libvirt/images/win2k3.qcow2
```

3.3.3 virt-install 高级用法示例

virt-install 有很丰富的选项参数以适用不同的场景，下面我们看两个示例。

示例 3-1：通过回答文件进行非交互式的自动化安装

在手工安装 RHEL/CentOS 时，需要设置多个选项参数，很烦琐。我们可以使用

Kickstart 文件实现自动化安装。Kickstart 文件是一个包含安装程序所需要的选项参数的文本文件。

默认情况下 RHEL/CentOS 的安装程序会在/root 目录生成一个名为 anaconda-ks. cfg 的文件,我们可以以这个文件的内容为“起点”快速生成一个回答文件,命令如下:

```
#cp anaconda-ks.cfg centos6.9.txt

#vi centos6.9.txt
  #可以根据需求进行修改,以下参数适合于 CentOS 6
autostep
install
cdrom
lang en_US.UTF-8
keyboard us
text
network --onboot yes --device eth0 --bootproto dhcp --noipv6
rootpw 123456
#firewall --service=ssh
firewall --disabled
authconfig --enableshadow --passalgo=sha512
#seLinux --enforcing
seLinux --disabled
timezone --utc Asia/Shanghai
bootloader --location=mbr --driveorder=vda --append="crashKernel=auto rhgb quiet"
#The following is the partition information you requested
#Note that any partitions you deleted are not expressed
#here so unless you clear all partitions first, this is
#not guaranteed to work
#clearpart --all --drives=vda
#volgroup VolGroup --pesize=4096 pv.253002
#logvol / --fstype=ext4 --name=lv_root --vgname=VolGroup --grow --size=1024 --
maxsize=51200
#logvol swap --name=lv_swap --vgname=VolGroup --grow --size=921 --maxsize=921
#part /boot --fstype=ext4 --size=500
#part pv.253002 --grow --size=1


#clearpart --all --initlabel --drives=vda
#part /boot --fstype=ext4 --size=200
#part pv.1 --grow --size 1
#volgroup vg0 pv.1
#logvol / --fstype=ext4 --name=lv_root --vgname=vg0 --grow
zerombr
clearpart --all --drives=vda
part /boot --fstype=ext4 --size=500
```

```
part pv.253002 --grow --size=1
volgroup VolGroup --pesize=4096 pv.253002
logvol swap --name=lv_swap --vgname=VolGroup --grow --size=921 --maxsize=921
logvol / --fstype=ext4 --name=lv_root --vgname=VolGroup --grow --size=1024 --
maxsize=51200

%packages --nobase
@core
%end
reboot
```

Kickstart 文件的参数设置可参见官方文档：https://access.redhat.com/documentation/en-us/red_hat_enterprise_Linux/6/html/installation_guide/s1-kickstart2-options。

除了这个回答文件，还需要为 virt-install 命令提供 3 个选项参数。

（1）--location：为 virt-install 指定 Linux 发行版本的安装源。既可以是网络位置，也可以是本地 ISO 文件，例如/iso/centos-6.10-i386-minimal.iso。

（2）--initrd-inject：需要与--location 选项一起使用，它指定一个保存在宿主机本地的 Kickstart 文件，例如/root/centos6.9.txt。

（3）--extra-args：需要与--location 选项一起使用，它传递给安装程序额外的参数，例如通过"ks=file:/centos6.9.txt"指定 Kickstart 文件。

示例命令如下：

```
#virt-install --name centos6.10vm3 \
--memory 1048 --vcpus 1 --disk size=8 \
--os-variant centos6.10 \
--location /iso/centos-6.10-i386-minimal.iso \
--initrd-inject /root/centos6.9.txt \
--extra-args="ks=file:/centos6.9.txt"

Starting install...
Retrieving file vmlinuz...                                    | 4.0 MB 00:00
Retrieving file initrd.img...                                 | 37 MB 00:00
Allocating 'centos6.10vm3.qcow2'                              | 8.0 GB 00:00
PuTTY X11 proxy: unable to connect to forwarded X server: Network error: Connection refused
Unable to init server: Could not connect: Connection refused

(virt-viewer:2057): Gtk-WARNING **: 22:29:23.035: cannot open display: localhost:10.0
Domain installation still in progress. You can reconnect to
the console to complete the installation process.
```

当虚拟机创建完毕之后，上述这个 virt-install 命令并不会启动 virt-viewer。只要 Kickstart 文件内容正确，就会在非交互模式下进行全自动的安装。当然，也可以通过

Cockpit 或 virt-manager 中的控制台查看安装的过程。

示例 3-2：对虚拟机性能影响很大的 os-variant 选项

Cockpit 和 virt-manager 都会对安装介质进行检测并提供一个操作系统类型的值。虽然这个值不是必需的，但是强烈建议设置最接近的值，因为 libvirt 会根据它来对虚拟机进行有针对性的优化。

可以通过--os-variant 选项为 virt-install 命令指定操作系统类型，例如 fedora32、rhel8，Windows 10。可以使用命令 osinfo-query 获取可接受的操作系统的列表。例如获得 CentOS 6、7、8 版本的字符串的命令如下：

```
#osinfo-query os | egrep centos[678]
centos6.0     | CentOS 6.0 | 6.0     | http://centos.org/centos/6.0
centos6.1     | CentOS 6.1 | 6.1     | http://centos.org/centos/6.1
centos6.10    | CentOS 6.10 | 6.10   | http://centos.org/centos/6.10
centos6.2     | CentOS 6.2 | 6.2     | http://centos.org/centos/6.2
centos6.3     | CentOS 6.3 | 6.3     | http://centos.org/centos/6.3
centos6.4     | CentOS 6.4 | 6.4     | http://centos.org/centos/6.4
centos6.5     | CentOS 6.5 | 6.5     | http://centos.org/centos/6.5
centos6.6     | CentOS 6.6 | 6.6     | http://centos.org/centos/6.6
centos6.7     | CentOS 6.7 | 6.7     | http://centos.org/centos/6.7
centos6.8     | CentOS 6.8 | 6.8     | http://centos.org/centos/6.8
centos6.9     | CentOS 6.9 | 6.9     | http://centos.org/centos/6.9
centos7.0     | CentOS 7 | 7         | http://centos.org/centos/7.0
centos8       | CentOS 8 | 8         | http://centos.org/centos/8
```

3.4 半虚拟化驱动 VirtIO

为了提高虚拟机的硬盘、网络及显卡设备的性能，需要在虚拟机中安装半虚拟化驱动程序 VirtIO 以替换普通的驱动程序。

3.4.1 半虚拟化驱动 VirtIO 原理

通过 virt-manager 可以查看虚拟机硬件的详细信息，例如虚拟磁盘总线接口，会看到的有 IDE、SATA、SCSI、USB 和 VirtIO 5 种类型，如图 3-29 所示。

类似地，虚拟机网卡有 e100、rtl8139 和 VirtIO 3 种类型，虚拟显卡有 Bochs、QXL、VGA 和 VirtIO 4 种类型。

上述 VirtIO 之外的虚拟设备均是全虚拟化(Full Virtualization)类型的设备。这种类型的设备的优点是适应性强，可以适合任何虚拟化操作系统，但是由于访问路径长，所以性能会比较差，如图 3-30 所示。

虚拟机操作系统为这些全虚拟化设备安装的是普通的驱动程序(通常在安装操作系统时会自动进行安装)，它是不知道自己运行在虚拟机中。操作对设备的请求被 Hypervisor

拦截，然后转发给 QEMU，QEMU 翻译之后再转发给宿主机上的驱动程序，最后才到达真实的物理硬件设备。

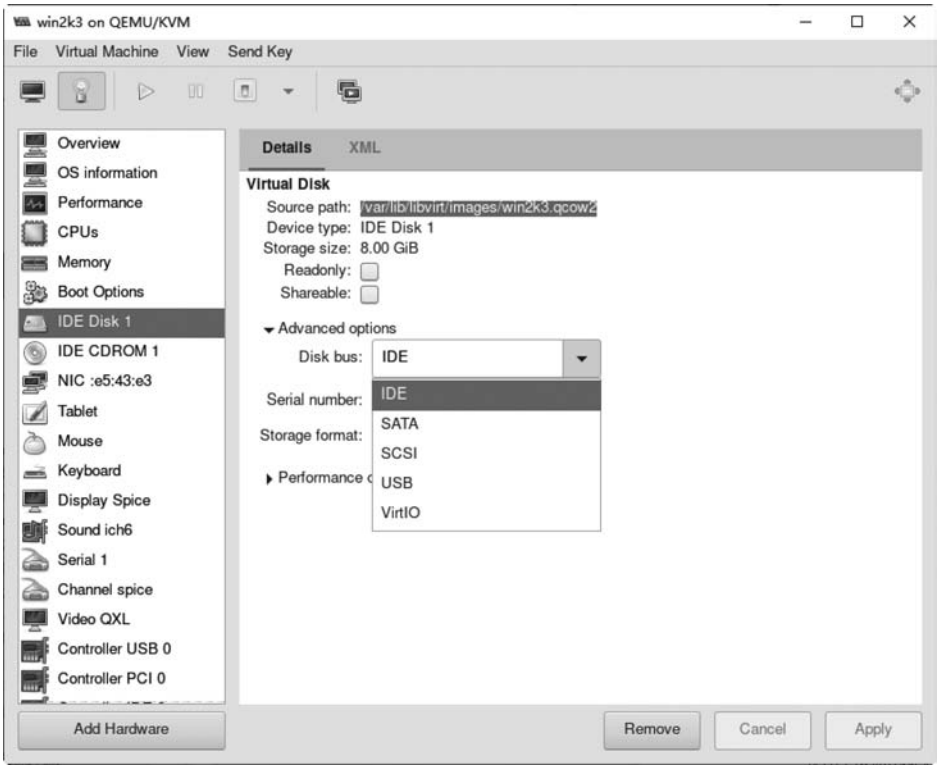


图 3-29　虚拟磁盘总线接口类型

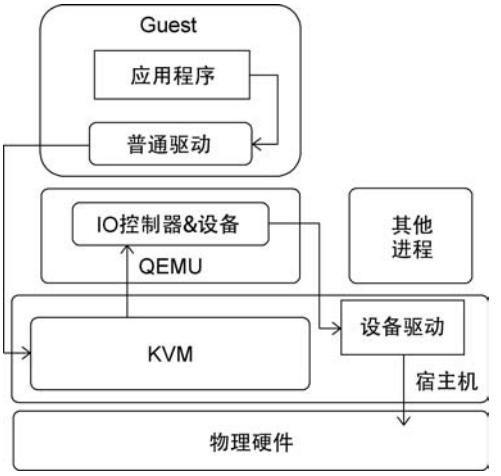


图 3-30　采用普通驱动的全虚拟化

如果为虚拟机分配的是 VirtIO 类型的虚拟设备，而且在虚拟机操作系统中安装了半虚拟化驱动 VirtIO，则这时虚拟机操作系统知道自己是虚拟机，所以数据直接发送给由 QEMU 提供的半虚拟化设备，而不经过 Hypervisor，通过宿主机上的驱动程序发送给真实的物理硬件设备，如图 3-31 所示。

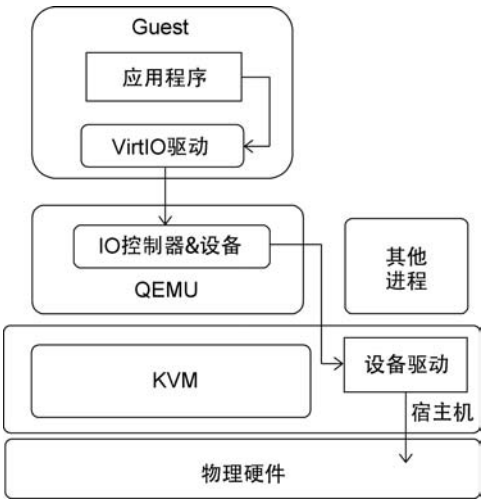


图 3-31 采用 VirtIO 驱动的半虚拟化

由于不与 Hypervisor 交互，所以半虚拟化驱动 VirtIO 缩短了访问路径，这种模式带来的好处就是性能要比采用普通的驱动好很多，从而让虚拟设备接近物理设备的性能。这个工作原理与 VMware Tools 驱动类似。

3.4.2 半虚拟化驱动 VirtIO 的安装

RHEL/CentOS 4.8、5.3 之后的发行版本都包含 VirtIO 设备的驱动程序，所以在虚拟机中安装这些发行版本时，如果安装程序检测到虚拟机的硬件是 VirtIO 类型的，就会自动安装相应的驱动程序。

由于 Windows 操作系统中不包含 VirtIO 设备的驱动程序，所以需要手工安装。在安装 Windows Server 2008 或之后的操作系统时，可以指定 VirtIO 驱动程序所在的位置（光盘、软件均可）。在安装 Windows Server 2003 或更早的操作系统时，安装程序仅会扫描并读取软盘中的驱动程序。

提示：微软公司于 2015 年 7 月就停止了对 Windows Server 2003 的支持，我们现在仅仅出于实验的目的使用它。

如何获得 VirtIO for Windows 驱动程序呢？一种方法是 RHEL/CentOS 8 的安装介质和软件仓库中包括 virtio-win 软件包，另外一种方法从 KVM 项目的网站下载。下载链

接为 http://www.Linux-kvm.org/page/WindowsGuestDrivers/Download_Drivers。

下面,我们采用第 1 种方法来为 Windows Server 2019 的安装程序提供驱动程序。

提示:在嵌套环境中安装 Windows Server 2019 通常会很慢。如果有条件,则建议在物理环境做这个实验。

首先,检查一下宿主机上是否有 VirtIO for Windows 的驱动程序,命令如下:

```
#rpm -qi virtio-win
Name        : virtio-win
Version     : 1.9.12
Release     : 2.el8
Architecture: noarch
Install Date: Wed 05 Aug 2020 10:19:48 PM CST
Group       : Applications/System
Size        : 663922676
License     : Red Hat Proprietary and GPLv2
Signature   : RSA/SHA256, Wed 22 Jul 2020 03:23:43 AM CST, Key ID 05b555b38483c65d
Source RPM  : virtio-win-1.9.12-2.el8.src.rpm
Build Date  : Wed 22 Jul 2020 02:46:14 AM CST
Build Host  : aarch64-05.mbox.centos.org
Relocations : (not relocatable)
Packager    : CentOS Buildsys <bugs@centos.org>
Vendor      : CentOS
URL         : http://www.redhat.com/
Summary     : VirtIO para-virtualized drivers for Windows(R)
Description :
VirtIO para-virtualized Windows(R) drivers for 32-bit and 64-bit Windows(R) guests.

#rpm -ql virtio-win | grep .iso
/usr/share/virtio-win/virtio-win-1.9.12.iso
/usr/share/virtio-win/virtio-win.iso

#ls -l /usr/share/virtio-win/*.iso
-rw-r--r--. 1 root root 332519424 Jul 22 02:46 /usr/share/virtio-win/virtio-win-1.9.
12.iso
lrwxrwxrwx. 1 root root 21 Jul 22 02:46 /usr/share/virtio-win/virtio-win.iso -> virtio-
win-1.9.12.iso
```

virtio-win.iso 是 virtio-win-1.9.12.iso 的符号链接文件。可以通过它为虚拟机提供驱动程序。

通过 virt-manager 创建 Windows Server 2019 的虚拟机。默认情况下,虚拟磁盘的接口为 SATA,我们将其修改为 VirtIO,如图 3-32 所示。

启动虚拟机,开始安装 Windows Server 2019。由于安装程序没有 VirtIO 设备的驱动

程序，所以无法识别 VirtIO 接口的磁盘，它会提示“我们找不到任何驱动器。要获取存储设备驱动程序，请单击‘加载驱动程序’”，如图 3-33 所示。

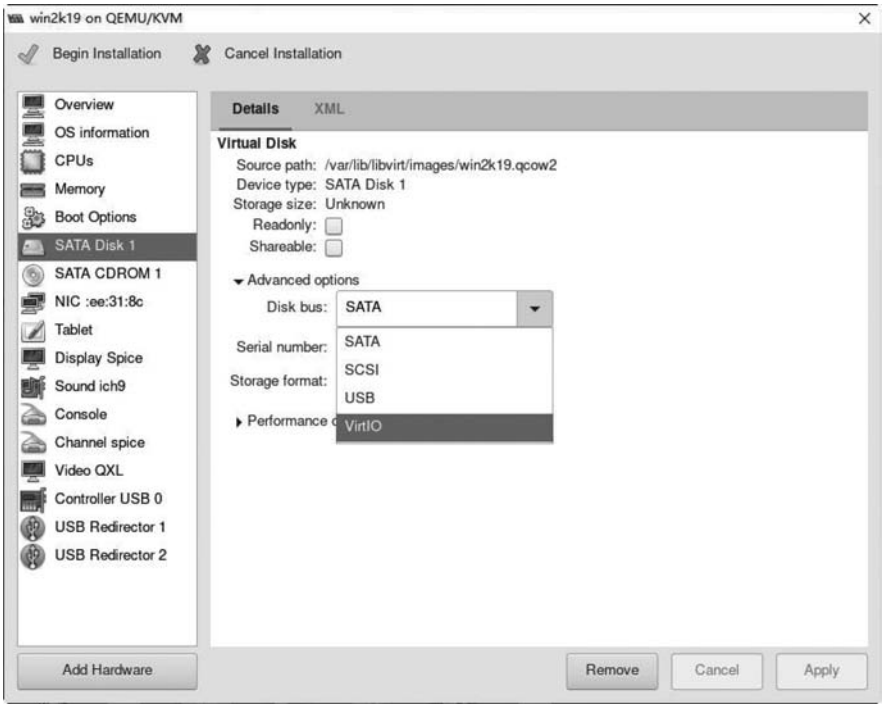


图 3-32 将虚拟机磁盘接口修改为 VirtIO

图 3-33 需要为安装程序加载驱动程序

在单击“加载驱动程序”之前，我们需要修改虚拟机的配置，为将虚拟光驱“换一张”光盘。在虚拟机配置中找到 CDROM，单击 Browse 按钮，如图 3-34 所示。

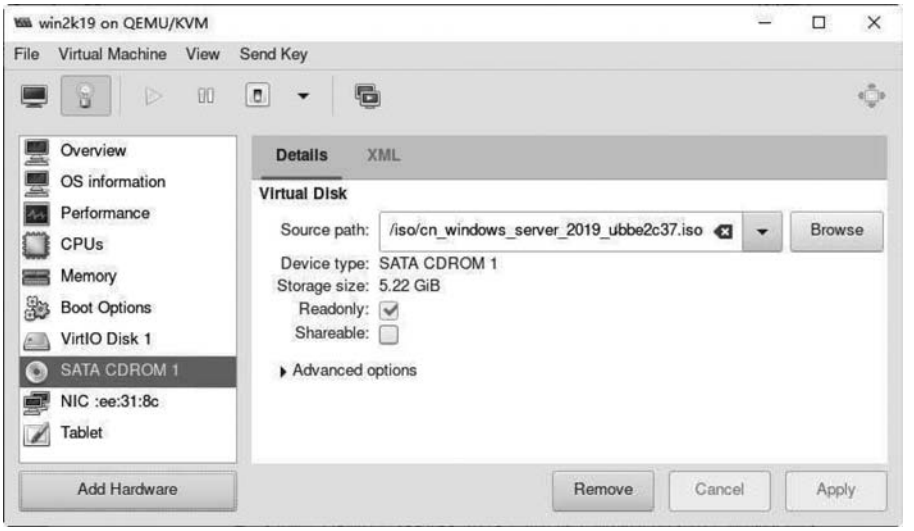


图 3-34　修改虚拟机 CDROM 配置

选中/usr/share/virtio-win/目录中 virtio-win. iso 文件，单击 Open 按钮，如图 3-35 所示。

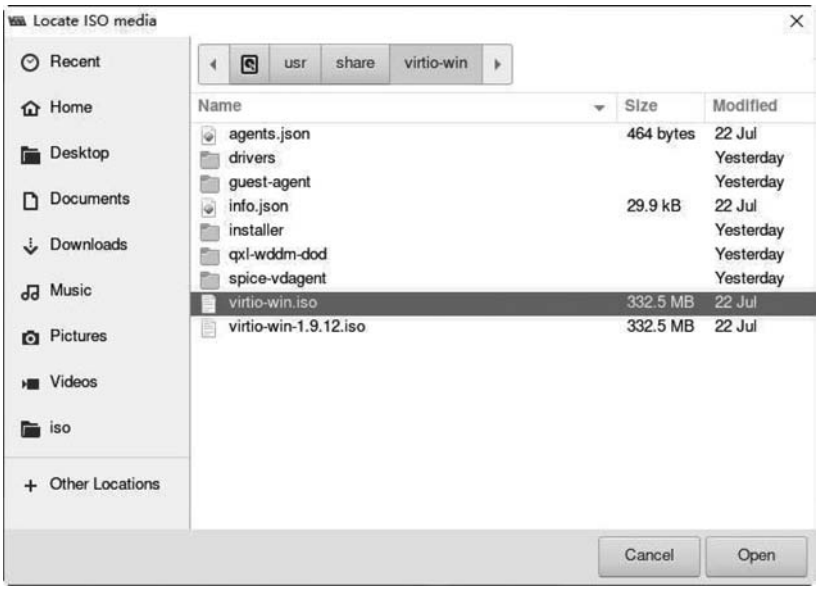


图 3-35　选中 VirtIO for Windows 驱动程序的 ISO 文件

这个操作相当于给虚拟机更换了一张光盘，单击 Apply 按钮，如图 3-36 所示。

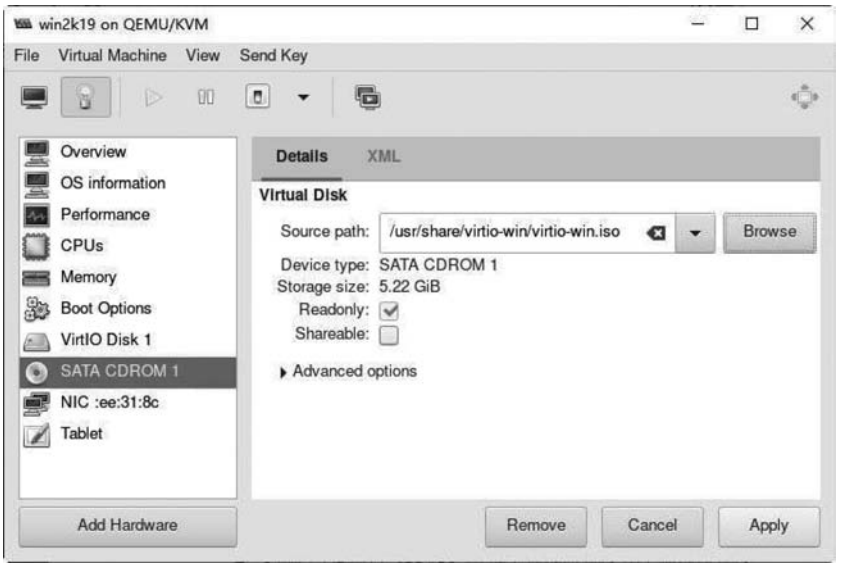


图 3-36　为虚拟机 CDROM 指定 VirtIO for Windows 驱动程序的 ISO 文件

返回 Windows 安装程序，单击“加载驱动程序”，会出现加载程序提示窗口。单击“确定”按钮，如图 3-37 所示。

图 3-37　为安装程序加载驱动程序

安装程序会搜索 CD、DVD 或 U 盘，并将找到的驱动程序显示出来。在本实验中，选择 Windows Server 2019 的驱动程序（光盘中\amd64\2k19\目录下的 viostor.inf），然后单击“下一步”按钮，如图 3-38 所示。

图 3-38　在找到的驱动程序中选择适合的版本

有了 VirtIO 驱动程序，安装程序就可以识别虚拟机的磁盘了，如图 3-39 所示。

图 3-39　安装程序识别出 VirtIO 接口的磁盘

如果是 Windows Server 2003 或更早的操作系统，安装程序仅会扫描并读取保存在软盘中的驱动程序。最简单的方法是从 Fedora 社区下载包含 VirtIO 驱动程序的虚拟磁盘文件，其链接为 https://fedorapeople.org/groups/virt/virtio-win/direct-downloads/stable-virtio/virtio-win_x86.vfd。

通过 virt-manager 创建 Windows Server 2003 虚拟机，将虚拟磁盘的接口设置为 VirtIO，再添加一个软盘驱动器，在 Add New Virtual Hardware 中选中 Storage，然后在右框选中 Select or create custom storage，单击 Manage 按钮，如图 3-40 所示。

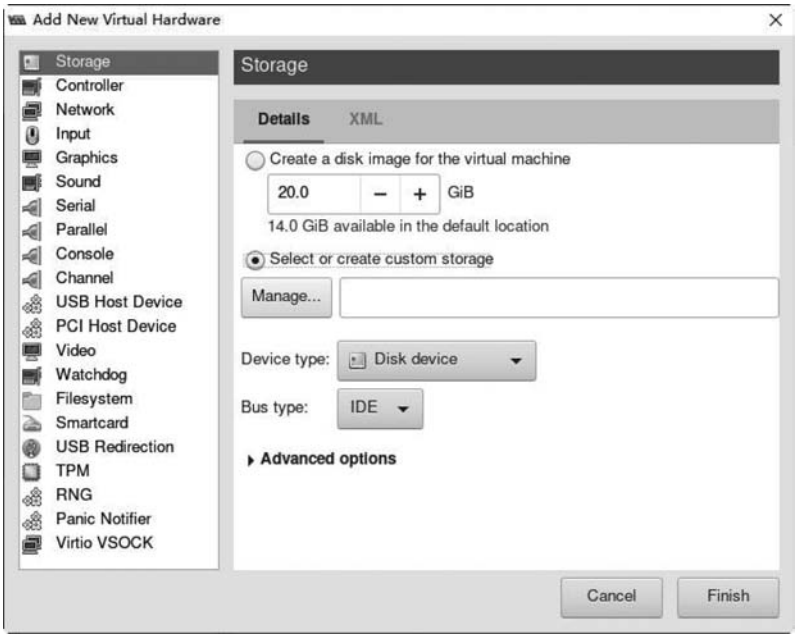


图 3-40　添加新的存储

指定包含 VirtIO 驱动程序的虚拟软盘文件 virtio-win_x86.vfd，如图 3-41 所示。

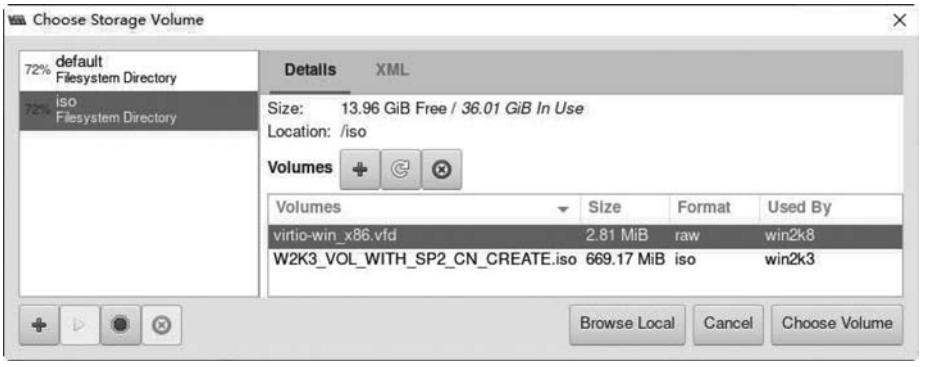


图 3-41　指定包含 VirtIO 驱动程序的虚拟软盘文件

然后将这个新存储设备的设备类型指定为 Floppy device，然后单击 Finish 按钮继续，如图 3-42 所示。

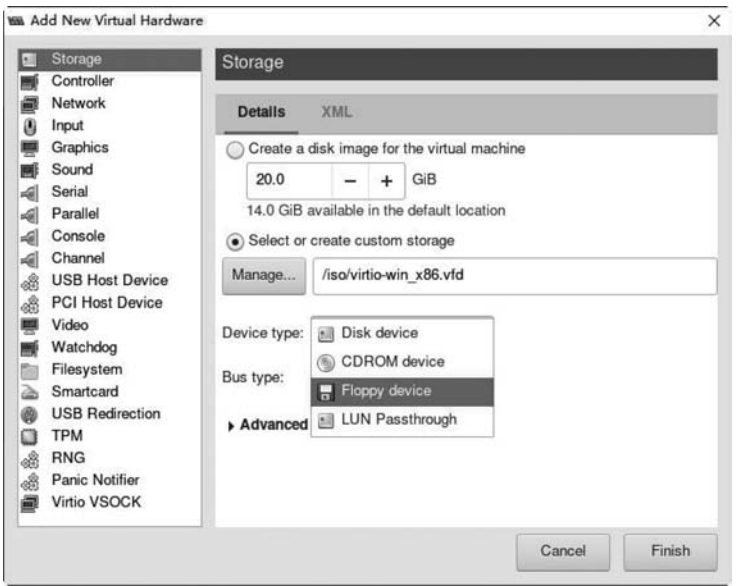


图 3-42　将新存储的设备类型指定为 Floppy device

Windows Server 2003 的安装程序在安装过程中会读取软盘。由于找到了驱动程序，所以会看到 VirtIO 接口的虚拟磁盘，如图 3-43 所示。

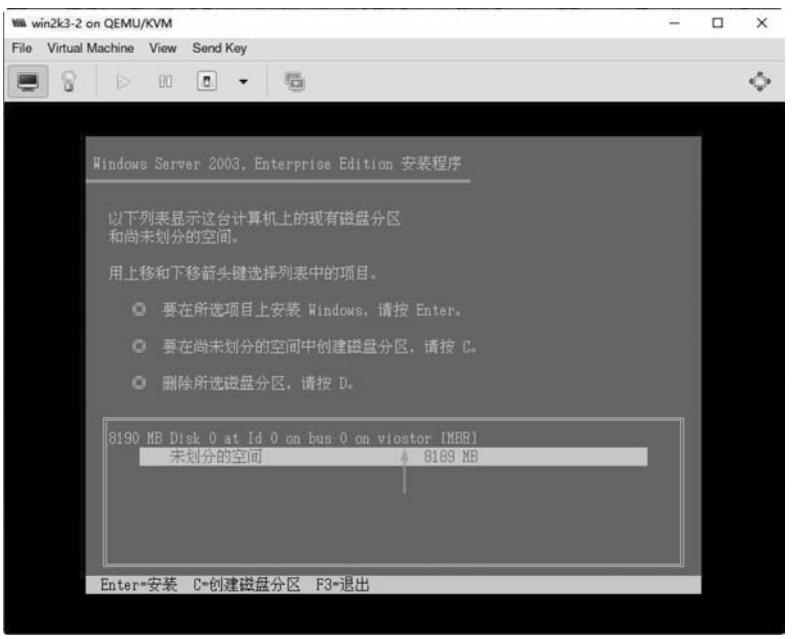


图 3-43　Windows Server 2003 安装程序读取并加载保存在软盘中的 VirtIO 驱动程序

不管通过哪种方式为 Windows 操作系统提供了 VirtIO 驱动程序，都会在设备管理器看到磁盘驱动器和存储控制器类型为 VirtIO SCSI 接口，如图 3-44 所示。

图 3-44　正确安装 VirtIO 存储控制器驱动程序的虚拟机

3.5　QEMU Guest Agent

3.5.1　QEMU Guest Agent 原理

为了更好地管理虚拟机，包括 KVM 在内的虚拟化平台都需要通过某种机制与虚拟机进行通信，这样既可以获得虚拟机操作系统的详细信息，也可以向虚拟机操作系统发出指令。例如：文件系统的冻结和解冻、系统挂起或安全地关闭等。VMware 虚拟机平台是通过在虚拟机中安装 VMware Tools 实现的，而 KVM 虚拟机则需要安装 QEMU Guest Agent 并正确运行，除此之外还需要为 KVM 虚拟机配置 VirtIO 串行控制器。

QEMU Guest Agent 是虚拟机上的守护程序或服务。想让它正常地运行，虚拟机上还必须有一个名为 Channel qemu-ga 的设备(ga 是 Guest Agent 的缩写)，如图 3-45 所示。

宿主机上的 virsh、virt-manager 等基于 libvirt 的应用程序与虚拟机上的 QEMU Guest Agent 进行通信，此通信并不是通过网络进行的，而是通过虚拟机上 VirtIO 串行控制器设备进行的，如图 3-46 所示。这个 VirtIO 串口控制器设备是通过一个字符设备驱动程序(通

常是 UNIX 套接字)连接到宿主机上的,而在虚拟机中 QEMU Guest Agent 负责侦听此串行通道上的信号。

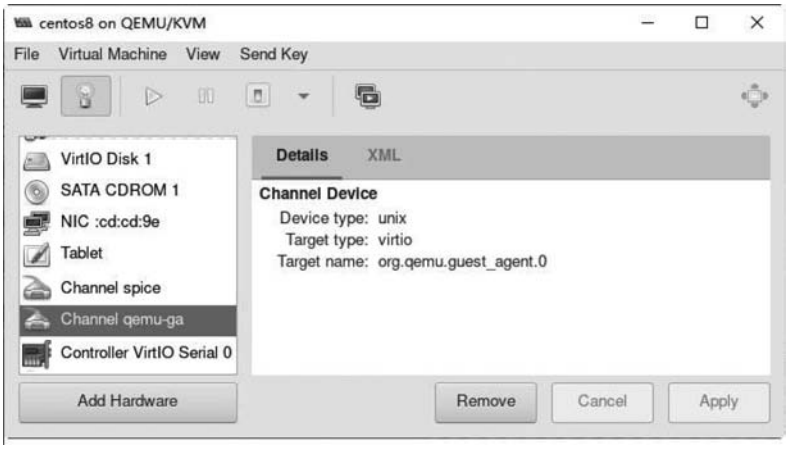


图 3-45 虚拟机上的 qemu-ga 通道设备

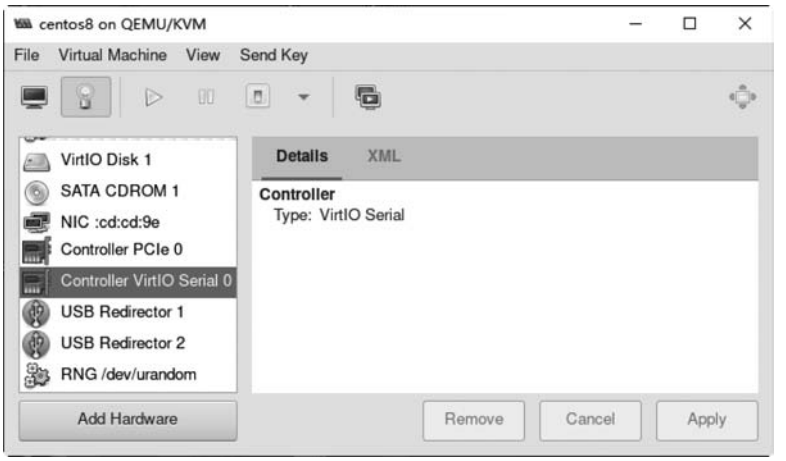


图 3-46 虚拟机上的 VirtIO 串口控制器设备

一旦有了 QEMU Guest Agent 的支持,很多基于 libvirt 的命令或应用程序功能变得更强大,以 virsh 命令为例:

(1) 使用 virsh shutdown --mode = agent 命令来关闭虚拟机,这种关闭方法比命令 virsh shutdown --mode = acpi 更可靠,因为相当于在虚拟机操作系统中发出类似 shutdown 指令,从而确保以干净状态关闭虚拟机。

(2) 使用 virsh domfsinfo 命令可以获得正在运行的虚拟机中已挂载文件系统的列表。

(3) 使用 virsh domtime 命令可以查询或设置虚拟机的时钟。

(4) 使用 virsh domifaddr --source agent 命令可以查询虚拟机操作系统的 IP 地址。

提示：宿主机通过 QEMU Guest Agent 与虚拟机交互的范围大小与功能的多寡还依赖于虚拟机的操作系统的版本。例如 Windows Server 2003 仅支持很少的命令。可以通过 virsh qemu-agent-command 虚拟机名称 '{"execute":"guest-info"}' 命令来查询支持的指令。

3.5.2 Linux 下的 QEMU Guest Agent

目前，不管是通过 Cockpit、virt-manager 还是 virt-install 创建的 Linux 虚拟机，默认都会配置 VirtIO 串口控制器和 qemu-ga 通道设备，而且 RHEL/CentOS 7 之后发行版本的安装程序，一旦检测出拥有 VirtIO 串口控制器的设备就会自动安装 QEMU Guest Agent，这使对 Linux 虚拟机的管理更加方便。命令如下：

```
#virsh domfsinfo centos8
Mountpoint      Name      Type      Target
-----------------------------------------------
/               dm-0      ext4      vda
/boot           vda1      ext4      vda

#virsh reboot centos8  --mode agent
Domain centos8 is being rebooted
```

这两个命令之所以成功是由于虚拟机 centos8 有 VirtIO 串行控制器设备，而且 QEMU Guest Agent 守护程序正常运行。下面在虚拟机上进行查看，命令如下：

```
[root@guest ~]#cat /etc/system-release
CentOS Linux release 8.1.1911 (Core)

[root@guest ~]#systemctl status qemu-guest-agent.service
● qemu-guest-agent.service - QEMU Guest Agent
   Loaded: loaded (/usr/lib/systemd/system/qemu-guest-agent.service; enabled; v>
   Active: active (running) since Wed 2020-08-19 21:48:57 CST; 41min ago
Main PID: 866 (qemu-ga)
    Tasks: 1 (limit: 11493)
   Memory: 2.1M
   CGroup: /system.slice/qemu-guest-agent.service
           └─866 /usr/bin/qemu-ga --method=virtio-serial --path=/dev/virtio-
por>

Aug 19 21:48:57 localhost.localdomain systemd[1]: Started QEMU Guest Agent.
```

如果将虚拟机中的 QEMU Guest Agent 守护程序停止，命令如下：

```
[root@guest ~]# systemctl stop qemu-guest-agent.service
```

这时再在宿主机上使用 virsh 命令来与虚拟机进行交互，命令如下：

```
1 # virsh domfsinfo centos8
    error: Unable to get filesystem information
    error: Guest agent is not responding: QEMU guest agent is not connected

2 # virsh reboot centos8 --mode agent
    error: Failed to reboot domain centos8
    error: Guest agent is not responding: QEMU guest agent is not connected

3 # virsh reboot centos8 --mode acpi
    Domain centos8 is being rebooted
```

由于 QEMU Guest Agent 守护程序停止，所以第 1 和第 2 行命令执行失败。第 3 行命令成功的原因是指定的模式为 APCI(高级配置和电源管理接口 Advanced Configuration and Power Management Interface)。

如果是 RHEL/CentOS 6 的虚拟机，就需要手工来安装 QEMU Guest Agent，命令如下：

```
# cat /etc/system-release
  CentOS release 6.10 (Final)

# yum -y install qemu-guest-agent

# chkconfig qemu-ga on

# chkconfig --list qemu-ga
  qemu-ga        0:off    1:off    2:on    3:on    4:on    5:on    6:off
```

提示：由于 CentOS 6 于 2020 年 11 月 30 日到期，原软件仓库已经被弃用，所以执行 YUM 命令会出错。解决方法是修改/etc/yum.repos.d/CentOS-Base.repo，使用 vault 软件仓库源：

[base]中的地址：baseURL=https://vault.centos.org/6.10/os/$basearch/

[updates]中的地址：baseURL=https://vault.centos.org/6.10/updates/$basearch/

[extras]中的地址：baseURL=https://vault.centos.org/6.10/extras/$basearch/

3.5.3 Windows 下的 QEMU Guest Agent

与创建 Linux 虚拟机不同，不管是通过 Cockpit、virt-manager 还是 virt-install 创建的

Windows 虚拟机，默认仅仅会配置 VirtIO 串口控制器设备，而没有 qemu-ga 通道设备，所以我们需要手工添加该设备。

在 Add New Virtual Hardware 窗口中选择 Channel，然后在 Name 的下拉列表框中选中 org. qemu. guest_agent. 0，在 Device Type 中选择 Unix Socket（unix），选中 Auto socket，单击 Finish 按钮，如图 3-47 所示。

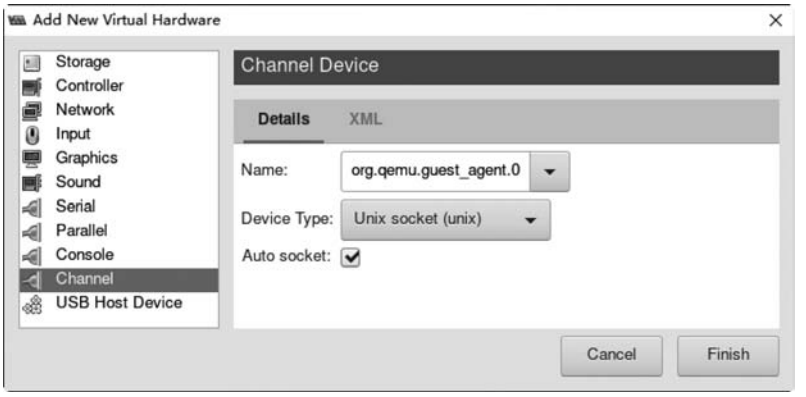


图 3-47　为虚拟机添加 qemu-ga 通道设备

有了 VirtIO 串口控制器和 qemu-ga 通道设备之后，还需要在 Windows 操作系统中安装 VirtIO 串口控制器的驱动程序及 qemu-ga 软件。如果没有安装驱动程序，则 Windows 虚拟机会将 VirtIO 串口控制器设备显示为“PCI 简单通信控制器”，如图 3-48 所示。

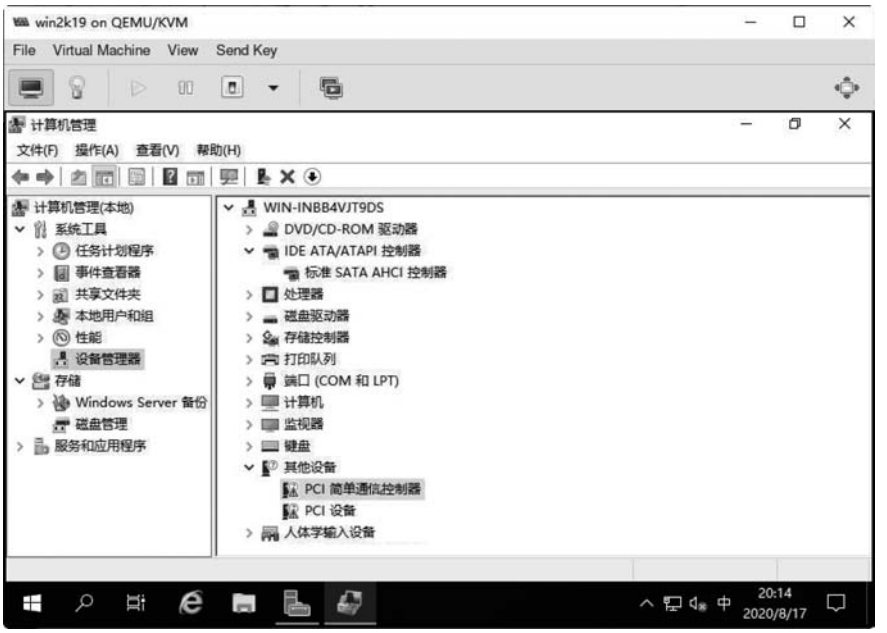


图 3-48　没有安装驱动程序的 VirtIO 串口控制器设备

提示：另外一个没有驱动程序的“PCI 设备”是 VirtIO Balloon 设备，这会在后续章节中介绍。

下面以 Windows Server 2019 的虚拟机为例，演示 VirtIO 串口控制器设备驱动及 QEMU Guest Agent 的安装。

首先，将宿主机的/usr/share/virtio-win/virtio-win.iso 通过 CDROM 附加给虚拟机，然后在虚拟机中双击光盘中的 virtio-win-guest-tools.exe 来启动安装程序，如图 3-49 所示。

图 3-49　VirtIO-win guest tools 安装程序

VirtIO-win guest tools 除了包括多种 VirtIO 驱动程序之外，还包括 QEMU Guest Agent、QEMU Guest Agent VSS Provider 和 Spice Agent 共 3 个代理服务，如图 3-50 所示，保持默认值，单击 Next 按钮开始安装。

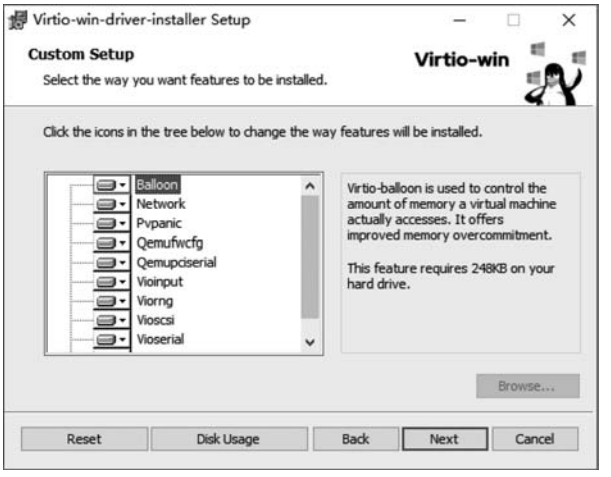


图 3-50　VirtIO-win guest tools 安装选项

安装结束之后，会在设备管理器中看到安装了正确驱动程序的 VirtIO 设备，如图 3-51 所示。

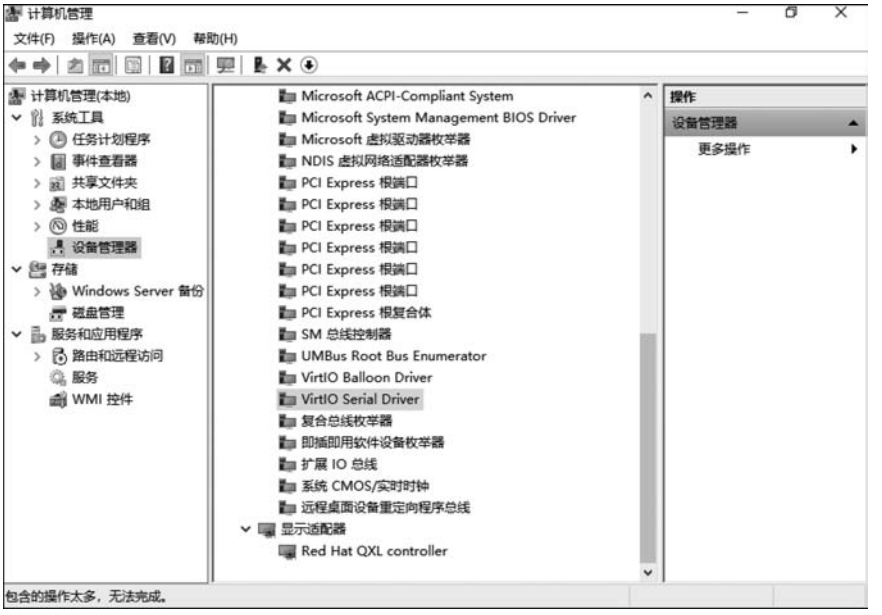


图 3-51　安装了正确驱动程序的 VirtIO 设备

在系统服务中会看到新增的 QEMU Guest Agent、QEMU Guest Agent VSS Provider 和 Spice Agent 等服务，如图 3-52 所示。

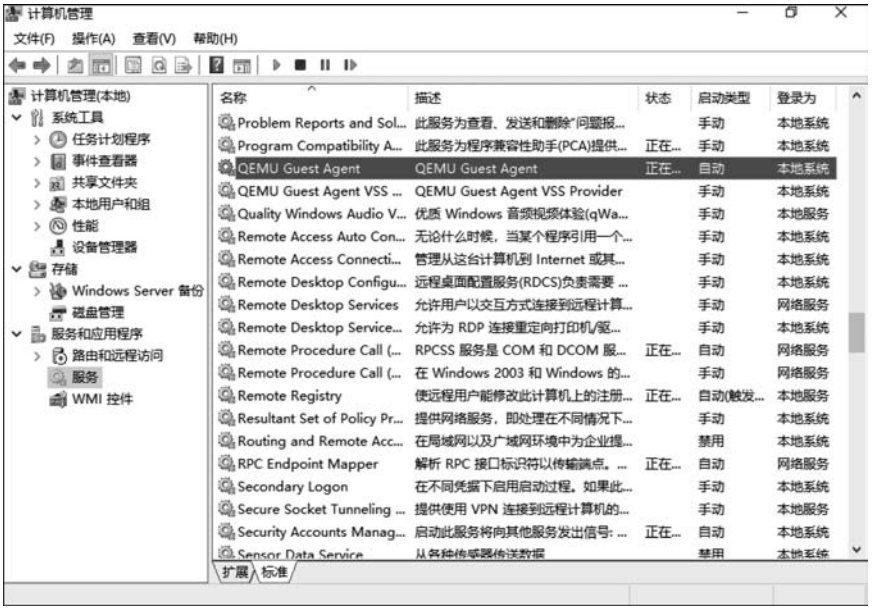


图 3-52　QEMU Guest 的 3 个代理服务

下面，我们在宿主机上通过 virsh 命令与一个名为 win2k19 的虚拟机进行交互操作，包括重置 administrator 的密码，命令如下：

```
#virsh domfsinfo win2k19
Mountpoint          Name      Type      Target
------------------------------------------------------------------------
System Reserved     \\?\Volume{6ecd2dc1-0000-0000-0000-100000000000}\ NTFS
C:\                 \\?\Volume{6ecd2dc1-0000-0000-0000-602200000000}\ NTFS
D:\                 \\?\Volume{f6e0b9f5-df06-11ea-a74a-806e6f6e6963}\ UDF
E:\                 \\?\Volume{f6e0b9f6-df06-11ea-a74a-806e6f6e6963}\ CDFS

#virsh domifaddr win2k19 --source agent
Name              MAC address          Protocol       Address
-------------------------------------------------------------------------
以太网            52:54:00:9c:85:e5    ipv6           fe80::c50:cad9:2ddf:c8f8%6/64
-                 -                    ipv4           192.168.122.94/24
Loopback  Pseudo-Interface 1          ipv6           ::1/128
-                 -                    ipv4           127.0.0.1/8

#virsh set-user-password win2k19 --user administrator --password P@ssw0rd1
Password set successfully for administrator in win2k19
```

宿主机的/usr/share/virtio-win/virtio-win.iso 其实是同目录下 virtio-win-1.9.12.iso 的符号链接。这个版本的 virtio-win-guest-tools.exe 仅支持 Windows 8、Windows Server 2012 及以后的 Windows 的版本，即使手工安装驱动及软件也仅仅可支持到 Windows 7。

如果需要使用更早的 Windows 操作系统，例如在实验环境中需要使用 Windows Server 2003，则可以在 Fedora 社区网站上下载早期的 VirtIO 驱动和 QEMU Guest Agent 软件，下载网址为 https://fedorapeople.org/groups/virt/virtio-win/direct-downloads。

提示：笔者在 Windows Server 2003 下测试过 virtio-win-0.1.100.iso、virtio-win-0.1.118.iso，均可正常运行。

3.6 显示设备与协议

为了让虚拟机显示图形，需要为其提供两个组件：显示设备（显卡）及从客户端访问图形的方法或协议。

3.6.1 显示设备

显示设备（显卡）是计算机的输出设备，但并不是必需的设备，例如：对于 Linux 来讲，如果不需要显示图形或进行控制台操作，串口的输出就工作得很好，但是多数情况下还是会

为虚拟机配置显示设备。

虽然可以将宿主机的显卡通过透传的方式分配给虚拟机，但是最常见的还是为其分配虚拟显卡。

libvirt 官方目前支持 vga、cirrus、vmvga、xen、vbox、qxl、virtio、gop、bochs、ramfb、none 等 11 种不同类型的虚拟/仿真图形适配器，它们的区别主要是功能和分辨率。目前在 Cockpit 中无法配置虚拟显卡的类型，只能使用默认的 QXL。virt-manager 中支持 4 种虚拟显卡，默认也是 QXL，如图 3-53 所示。

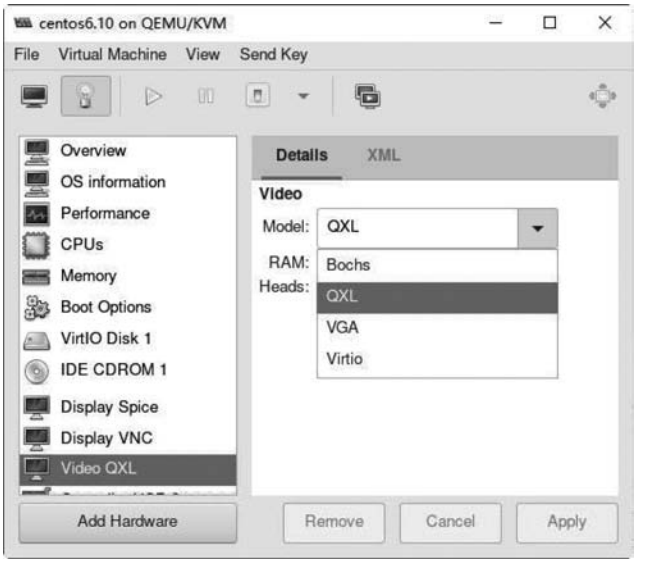


图 3-53　虚拟机的虚拟显卡

QXL 是具有 2D 支持的半虚拟图形驱动程序，其默认的 VGA 内存大小为 16MB，可以满足 2560×1440 分辨率需求，而且是与 SPICE 远程显示协议配合得最好的虚拟图形卡，所以推荐使用 QXL 虚拟图形适配器。

在使用 QXL 虚拟图形适配器时，通常保持默认的配置参数就可以很好地工作了。

```
<video>
  <model type='qxl'/>
</video>
```

对于 Windows 操作系统来讲，还需要安装 QXL 显卡驱动程序(可以在 virtio-win 软件包中找到)。安装了 QXL 虚拟显卡驱动程序的 Windows 操作系统，如图 3-54 所示。

3.6.2　显示协议

客户端访问远程主机图形界面的方法或协议有很多种，包括著名的 RDP、VNC、

SPICE、PCoIP 和 HDX，还有“小众”的 CodeCraft、NoMachine、FreeNX 等。它们之间的区别主要是色彩深度、加密、声频、带宽管理、文件系统重定向、打印机重定向、USB 和其他端口重定向等。本书仅涉及开源世界中最常见的 2 个协议 VNC 和 SPICE。

图 3-54　安装了 QXL 虚拟显卡驱动程序的 Windows Server 2003

通过 Cockpit、virt-manager 或 virt-install 创建虚拟机时，默认情况下会自动为其创建图形设备。图形设备对应的协议是显示协议。目前有两种类型的图形设备。

(1) SPICE：SPICE 是 Simple Protocol for Independent Computing Environment 协议的缩写。从本质上来讲，它仅仅是一种通信通道，它对在通道传递的数据没有要求。在 KVM 环境中，可以在这个通道中传递图像、声音及 USB 设备流等多种数据。

(2) VNC：VNC 是 Virtual Network Console 协议的缩写。VNC 是一种远程控制协议，可以实现对虚拟机的远程控制。

可以同时把这两种图形设备添加给虚拟机，如图 3-55 所示。当然，如果不需要通过控制台访问虚拟机，则可以不给虚拟机配置任何图形设备。

这两种图形设备都是 C/S 架构的。如果为虚拟机配置了 SPICE 图形设备，它就是 SPICE 的服务器端，常用的客户端有 virt-manager、virt-viewer。如果为虚拟机配置了 VNC 图形设备，它就是 VNC 的服务器端，除了 virt-manager、virt-viewer 之外，传统的 VNC 客户端如 VNC Viewer、noVNC 都可以访问虚拟机的控制台。

提示：noVNC 提供了一种使用支持 HTML5 Canvas 的浏览器访问 VNC Server 的方法。包括 Cockpit 在内的很多云计算、虚拟机管理软件采用了 noVNC 访问控制台。

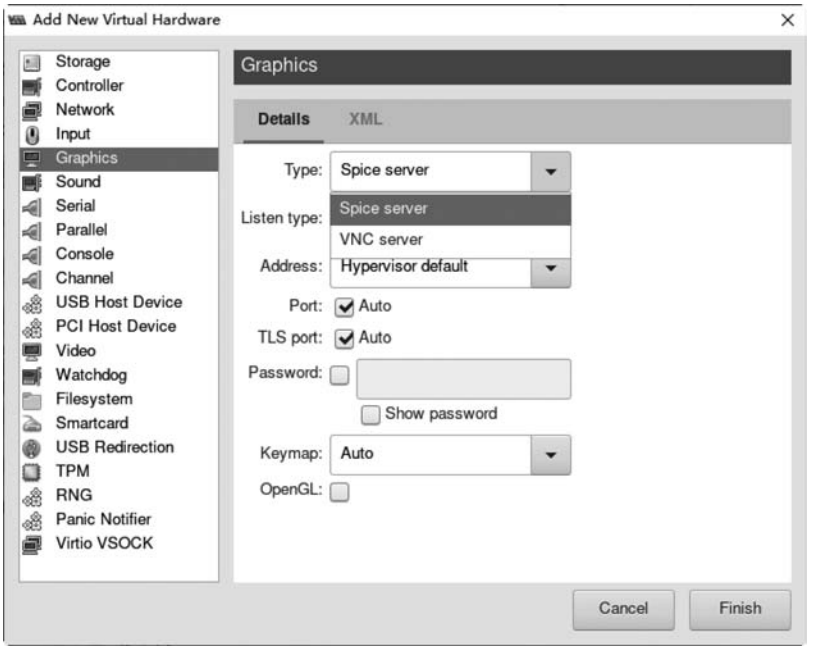
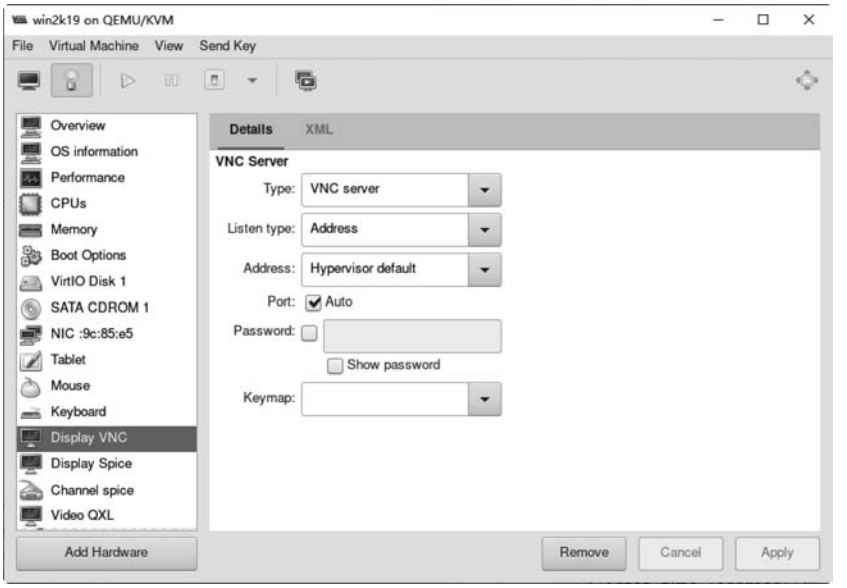


图 3-55　为虚拟机添加图形显示设备

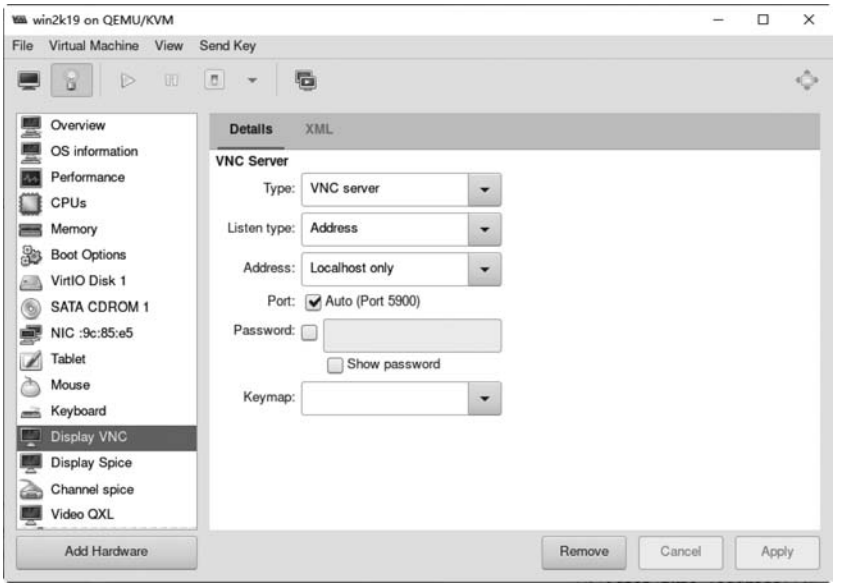
出于安全考虑，QEMU 配置文件中的 SPICE 和 VNC 的默认监听地址为 127.0.0.1，也就是仅允许来自宿主机自身的 SPICE 和 VNC 客户端进行连接。查看 QEMU 配置的命令如下：

```
# cat /etc/libvirt/qemu.conf
...
# VNC is configured to listen on 127.0.0.1 by default.
# To make it listen on all public interfaces, uncomment
# this next option.
#
# NB, strong recommendation to enable TLS + x509 certificate
# verification when allowing public access
#
# vnc_listen = "0.0.0.0"
...
# SPICE is configured to listen on 127.0.0.1 by default.
# To make it listen on all public interfaces, uncomment
# this next option.
#
# NB, strong recommendation to enable TLS + x509 certificate
# verification when allowing public access
#
# spice_listen = "0.0.0.0"
...
```

默认情况下，新创建的虚拟机都会“继承”这些默认的配置。如果虚拟机未启动，则可在 virt-manager 中查看 VNC 和 SPICE 监听地址，显示的配置为 Hypervisor Default，如图 3-56(a)所示。



(a)



(b)

图 3-56 虚拟机图形显示设备配置

启动虚拟机之后，再在 virt-manager 中查看 VNC 和 SPICE 监听地址，会发现配置变为 Localhost only，同时还会显示自动生成的监听端口，例如 5900，如图 3-56(b)所示。

可以在宿主机的命令行中查看监听的地址与端口，命令如下：

```
#netstat -anp | grep 590
tcp    0    0 127.0.0.1:5900 0.0.0.0:* LISTEN 18408/qemu-kvm
tcp    0    0 127.0.0.1:5901 0.0.0.0:* LISTEN 18408/qemu-kvm
```

在本示例中，VNC 型的图形设备监听地址是 127.0.0.1、监听端口是 5900，SPICE 型的图形设备监听地址也是 127.0.0.1，不过监听端口则是 5901。

也可以在 Cockpit 虚拟机控制台中查看监听的地址与端口，如图 3-57 所示。

图 3-57　Cockpit 虚拟机控制台中显示的图形设备监听地址与端口

3.6.3　Remote Viewer 连接虚拟机排错

掌握了 KVM 图形设备的原理之后，下面我们来分析一个故障案例。

我们在一台 Windows 主机上通过 Google Chrome 或 Firefox 等浏览器远程访问宿主机上的 Cockpit。

如果给虚拟机配置了 VNC 型的图形设备，则可以在 Cockpit 虚拟机台中使用“图形控制台(VNC)”来正常地访问虚拟机，如图 3-58 所示。

这是由于 Cockpit 的 noVNC 运行在宿主机上，所以它使用 127.0.0.1 访问虚拟机当然没有问题。

1. 故障现象

如果给虚拟机配置了 SPICE 型的图形设备，则可以在 Cockpit 虚拟机控制台中选择“Desktop Viewer 中的图形控制台”，单击“加载 Remote Viewer”按钮，浏览器会下载一个以 vv 为后缀的文件(vv 是 virt-viewer 的缩写)，如图 3-59 所示。

如果这台 Windows 主机已经安装了 Remote Viewer 软件(Windows 下的 virt-viewer)，由于文档关联的原因，会自动启动 Remote Viewer 软件，但是几秒之后，会提示“无法连接到图形服务器”，如图 3-60 所示。

图 3-58　在 Cockpit 虚拟机控制台中使用 VNC 访问虚拟机

图 3-59　在 Cockpit 虚拟机控制台中使用 Remote Viewer 访问虚拟机

图 3-60　Remote Viewer 软件提示无法连接到图形服务器

单击“确定”按钮，Remote Viewer 软件会自动退出，而且下载的那个以 vv 为后缀的文件也不见了。

2. 分析排错

以 vv 为后缀的文件其实是一个文本文件。我们在出现错误提示时，如果不单击“确定”按钮，保持提示窗口不关闭，则可找到下载的以 vv 为后缀的文件，通过文本编辑器打开它，会看到其内容如下：

```
[virt-viewer]
type=spice
host=127.0.0.1
port=5901
delete-this-file=1
fullscreen=0

[.............................GraphicsConsole]
```

这个文件中 host 和 port 所指定的 IP 地址和端口就是 Remote Viewer 软件要访问的目标。也就是说 Remote Viewer 访问的是 127.0.0.1，而不是宿主机的 IP 地址。

故障原因找到了，下面我们就进行排错操作。在实验环境中，最简单的方法就是将虚拟机的两种图形设备的监听地址都修改为宿主机的 IP 地址，如图 3-61 所示。

重新启用虚拟机之后，Cockpit 控制台中显示的监听地址就变成了宿主机的 IP 地址，如图 3-62 所示。

在宿主机的命令行中查看监听的地址与端口，命令如下：

```
#netstat -an | grep 590
tcp    0    0 192.168.114.231:5900    0.0.0.0:*  LISTEN
tcp    0    0 192.168.114.231:5901    0.0.0.0:*  LISTEN
```

在 Cockpit 控制台中单击“加载 Remote Viewer”按钮，浏览器便会下载配置文件，这次使用 Remote Viewer 就可以正常地访问虚拟机了，如图 3-63 所示。

这次下载的以 vv 为后缀的文件的内容如下：

```
[virt-viewer]
type=spice
```

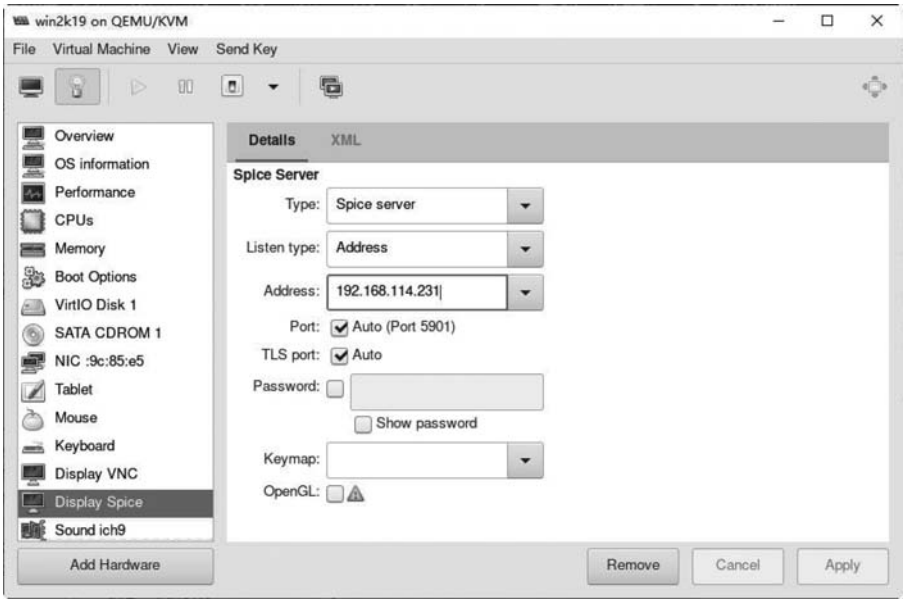


图 3-61 修改虚拟机图形显示设备监听的地址

图 3-62 Cockpit 虚拟机控制台中显示的图形设备监听的地址与端口

```
host = 192.168.114.231
port = 5901
delete - this - file = 1
fullscreen = 0

[.............................GraphicsConsole]
```

图 3-63　Remote Viewer 软件连接到虚拟机控制台

当然还要考虑宿主机上的防火墙配置。由于虚拟机图形设备监听的端口是 5900＋序号，所以根据宿主机上同时运行虚拟机的数量可以指定一个端口范围。例如端口范围是 5900～5950，命令如下：

```
#firewall-cmd --add-port=5900-5950/tcp --permanent

#firewall-cmd --add-port=5900-5950/tcp

#firewall-cmd --list-all
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: ens32
  sources:
  services: cockpit dhcpv6-client rdp ssh vnc-server
  ports: 5900-5950/tcp
  protocols:
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

3.6.4 Linux 下的 SPICE Agent

与 QEMU Guest Agent 类似，SPICE Agent 也是一种运行在 KVM 虚拟机中的守护程序或服务软件，它可以与 SPICE 客户端软件（例如：virt-manager、virt-viewer）协助工作，为用户提供更流畅的图形显示。例如，在 virt-manager 中调整窗口大小时，SPICE Agent 会自动调整虚拟机的分辨率来适应窗口的大小。除此之外，现在 SPICE Agent 还支持宿主机和虚拟机之间剪贴板数据的交换、USB 设备的重定向等功能。

在虚拟机上安装 SPICE Agent 之前，需要保证虚拟机上有 spice 通道设备，如图 3-64 所示。

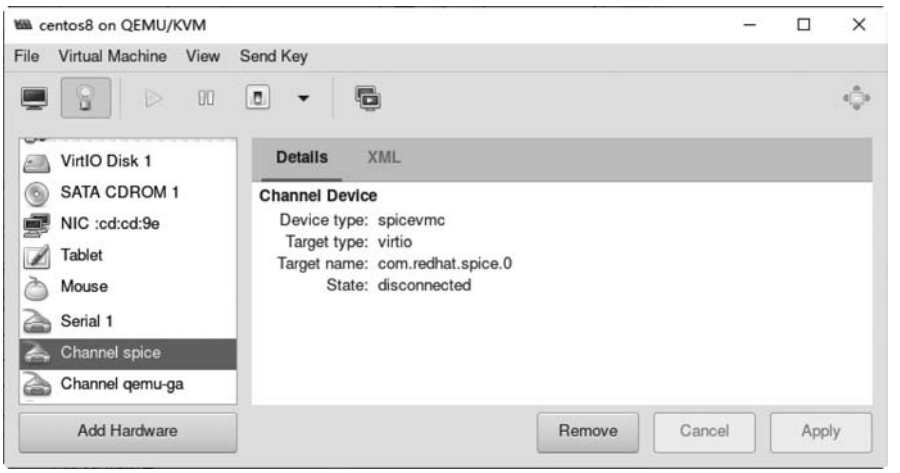


图 3-64 虚拟机的 spice 通道设备

然后，在 Linux 虚拟机中通过 yum 或 dnf 命令安装 SPICE Agent，命令如下：

```
[root@guest ~]# cat /etc/system - release
CentOS Linux release 8.3.2011

[root@guest ~]# dnf  - y install spice - vdagent

[root@guest ~]# rpm  - qi spice - vdagent
Name      : spice - vdagent
Version   : 0.20.0
Release   : 1.el8
…
URL       : https://spice - space.org/
Summary   : Agent for Spice guests
Description :
Spice agent for Linux guests offering the following features:
```

```
Features:
 * Client mouse mode (no need to grab mouse by client, no mouse lag)
   this is handled by the daemon by feeding mouse events into the Kernel
   via uinput. This will only work if the active X-session is running a
   spice-vdagent process so that its resolution can be determined.
 * Automatic adjustment of the X-session resolution to the client resolution
 * Support of copy and paste (text and images) between the active X-session and the client

[root@guest ~]# systemctl start spice-vdagentd
```

提示：在虚拟机中安装 SPICE Agent 的目的是为了提高图形显示效果，所以如果 Linux 虚拟机中不使用 X Windows，则可以不安装 SPICE Agent。

3.6.5 Windows 下的 SPICE Agent

如果希望提高 Windows 虚拟机的图形界面的显示效果，则强烈建议安装 SPICE Agent。

将 VirtIO-win 的 ISO 文件分配给虚拟机，双击 virtio-win-guest-tools.exe 安装程序。安装程序会自动安装 spice-guest-agent，如图 3-65 所示。

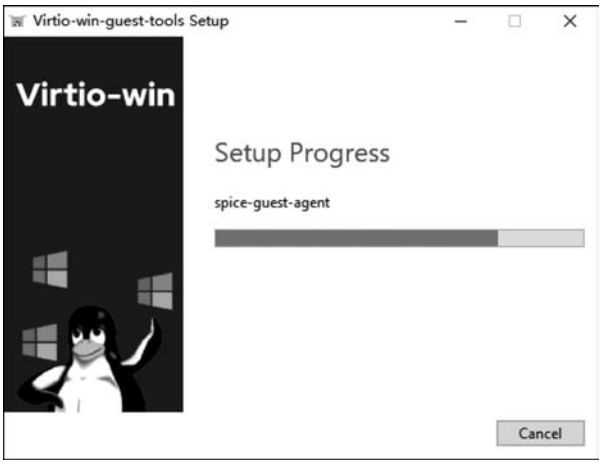


图 3-65 virtio-win-guest-tools 安装程序会安装 spice-guest-agent

安装之后，会看到系统中有一个名为 Spice Agent 的系统服务，如图 3-66 所示。

在这个服务的辅助下，当调整 virt-viewer 窗口的大小时，虚拟机图像也会随之进行缩放，同时还可以在宿主机与虚拟机之间通过剪贴板进行文本内容的交换。

图 3-66 Spice Agent 服务

3.7 本章小结

本章讲解了如何通过 Cockpit、virt-manager 和 virt-install 来安装虚拟机，VirtIO 驱动程序、QEMU Guest Agent、显示设备与协议原理、SPICE Agent 的工作原理及安装。

第 4 章将讲解虚拟机的管理。