管理虚拟存储

概述

- 虚拟磁盘概述
- 使用qemu-img管理虚拟磁盘
- 快照管理理
- 存储池
- 存储卷
- 虚拟磁盘离线访问工具

虚拟磁盘概述

- 虚拟化项目中存储的注意事项
- KVM的存储模式
- KVM的虚拟磁盘类型

虚拟化项目中存储的注意事项

- 存储的性能几乎总是虚拟化的瓶颈
- 通过多个硬盘驱动以分布磁盘I/O来实现存储解决方案
- 驱动器的速度越快越好，考虑SSD与机械硬盘的混合使用
- 考虑部署集中化的SAN/NFS来实现高可用性和实时迁移

物理

DAS (SCSI, SATA, SAS, N-SAS)
SAN (Fibre Channel, FCoE, iSCSI, SAS)
NAS (NFS、SMB)

虚拟适配器

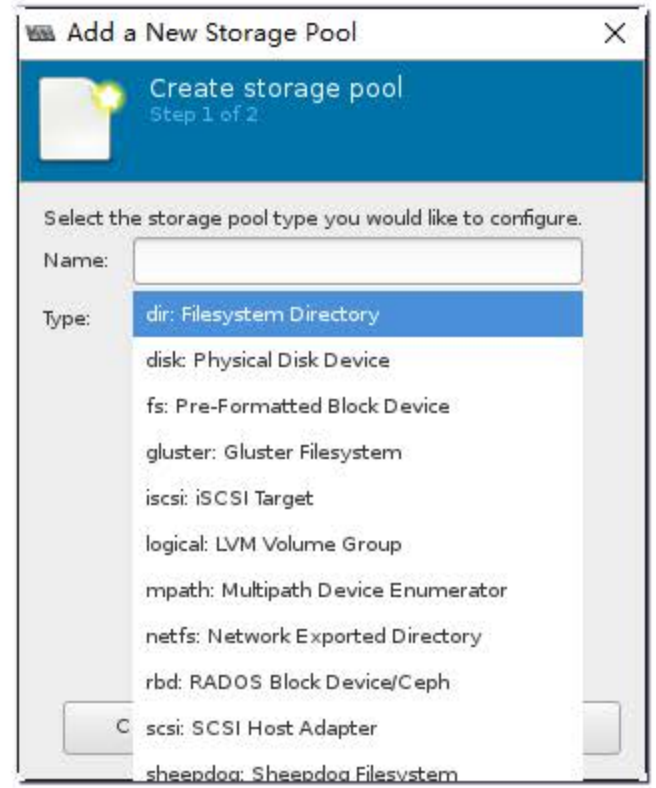
IDE, VirtIO, SCSI

虚拟磁盘

固定、动态、差异
缓存

KVM存储模式

- 基于文件系统的存储
  - dir: Filesystem Directory
  - fs: Pre-Formatted Block Device
  - netfs: Network Exported Directory
- 基于设备的存储
  - Disk: Physical Disk Device
  - Iscsi: iSCSI Target
  - logical: LVM Volume Group
  - ....
- 通过存储池来简介存储的管理

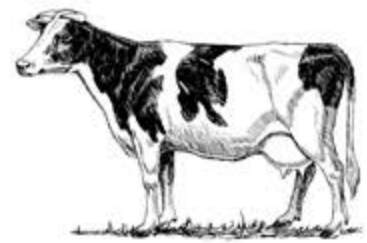


虚拟磁盘类型

- 固定　　Fixed
  - 在配置时，指定磁盘大小
  - 不管在虚拟磁盘上实际存储多少数据，都将占用相同大小主机磁盘空间
- 动态　　Dynamic
  - 增长到最大容量，但是只根据需求使用更多的空间
- 差异　　Differencing
  - 因为创建是差异磁盘，所以只保存变更的数据
  - 例如，将操作系统安装在父盘，然后创建差异化磁盘来执行进一步配置

KVM支持的虚拟磁盘类型

- raw
  - 这并非是一种真正的磁盘格式，而是代表虚拟机所使用的原始镜像
  - 它并不存储元数据，因此可以作为保证虚拟机兼容性的候选方案。然而，也正因为它不存储元数据，因此不能支持某些高级特性，比如快照和压缩等
  - 格式简单，容易转换为其他的格式。需要文件系统的支持才能支持sparse file
- cow：copy-on-write格式，昙花一现
- qcow： QEMU 早期的copy-on-write格式，过渡性方案
- qcow2
  - 按需进行分配磁盘空间，不管文件系统是否支持
  - 支持快照
  - 支持zlib的磁盘压缩
  - 支持AES的加密
- vmdk （Virtual Machine Disk）
  - VMware环境当中默认使用的磁盘格式
- vhd \ vhdx (Virtual Hard Disk)
  - 微软默认采用的文件格式
- vdi (VirtualBox)
- ..... 通过`qemu-img --help`查看支持的格式



```
# qemu-img --help  | grep Supported
Supported formats: vvfat vpc vmdk vhdx vdi ssh sheepdog
rbd raw host_cdrom host_floppy host_device file qed
qcow2 qcow parallels nbd iscsi gluster dmg tftp ftps
ftp https http cloop bochs blkverify blkdebug
```

使用qemu-img管理虚拟磁盘

- qemu-img概述
- 创建虚拟磁盘
- 检查虚拟磁盘
- 预分配磁盘策略
- 后备差异虚拟磁盘
- 虚拟磁盘格式转换
- 调整虚拟磁盘大小

qemu-img概述

- qemu-img是一个功能强制磁盘镜像管理工具
- qemu-img --help 包括以下功能
  - check 检查完整性
  - create 创建镜像
  - commit 提交更改
  - compare 比较
  - convert 转换
  - info 获得信息
  - map 映射
  - snapshot 快照管理
  - rebase 在已有的镜像的基础上创建新的镜像
  - resize 调整大小
  - amend 修订镜像格式选项

qcow2格式选项

- backing_file
  - 于指定后端镜像文件。
- backing_fmt
  - 设置后端镜像的镜像格式。
- cluster_size
  - 设置镜像中的簇大小，取值在512到2M之间，默认值为64K。
- preallocation
  - 设置镜像文件空间的预分配模式
- encryption
  - 用于设置加密

预分配策略

- off

  缺省策略，即不使用预分配策略

- metadata

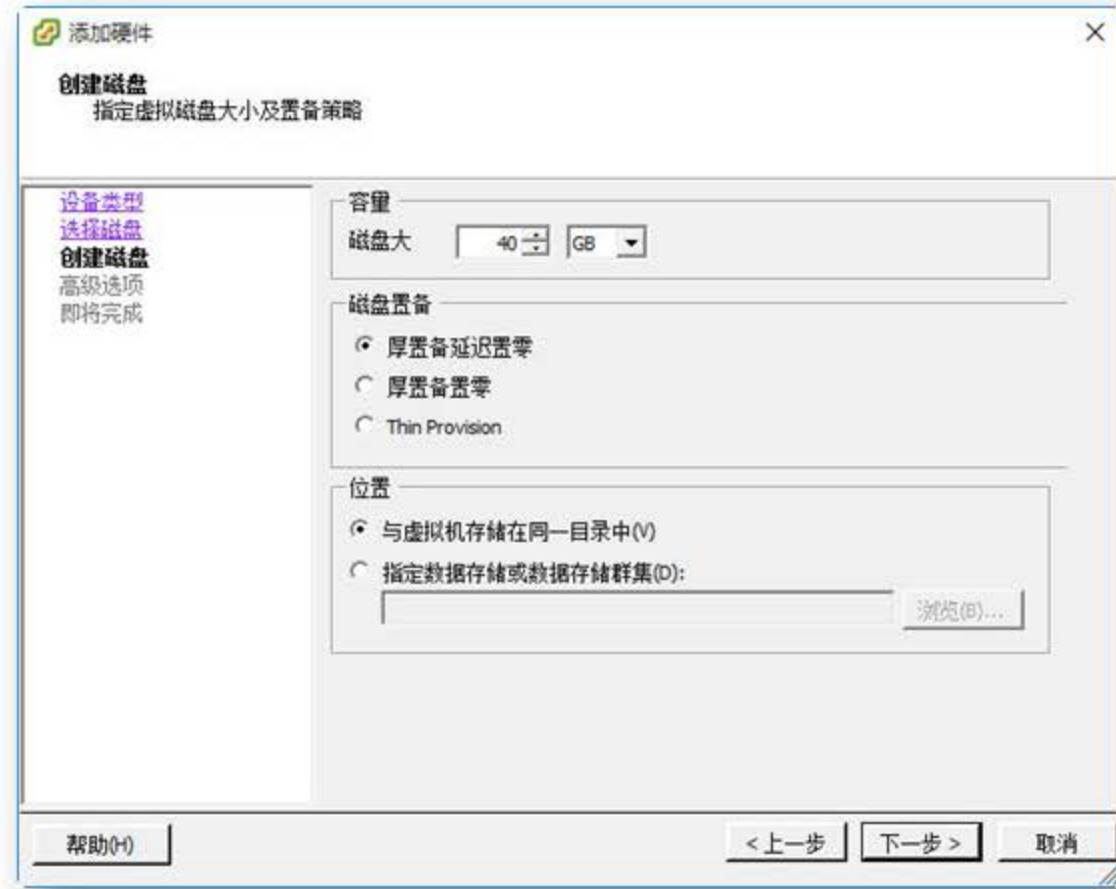
  分配元数据(metadata)，预分配后的虚拟磁盘仍然属于稀疏映像类型

- full

  分配所有磁盘空间并置零，预分配后的虚拟磁盘属于非稀疏映像类型

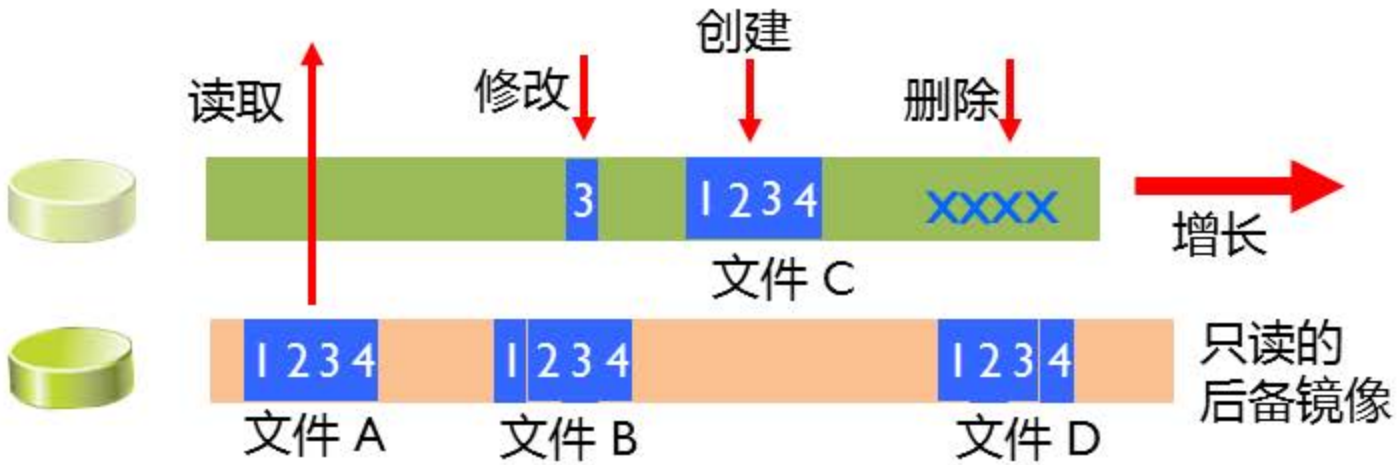
- falloc

  分配文件的块并标示它们的状态为未初始化，相对full模式来说，创建虚拟磁盘的速度要快很多



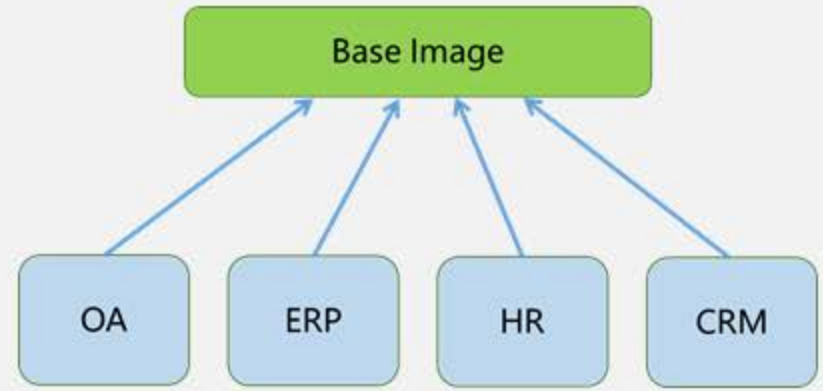
后备差异虚拟硬盘

- 存储与基础镜像（父）磁盘的变化
  - 基础镜像（父）磁盘不会改变
  - 差异磁盘隔离变化
  - 多个差异磁盘可以使用相同的基础镜像（父）磁盘
- 优点：标准化基础镜像，节省空间
- 缺点：增加了开销，较差的性能



演示：backing_file

```
# qemu-img create -f qcow2 \
-o backing_file=Base_CentOS7-1151-disk0.qcow2 \
oa-disk0-with-b.qcow2

# virt-install \
  --import \
  --name=oa \
  --vcpus=1 --ram=1024 \
  --disk path=/vm/oa-disk0-with-b.qcow2 \
  --network network=default \
  --graphics vnc,listen=0.0.0.0 \
  --os-type=linux \
  --os-variant=rhel6
```



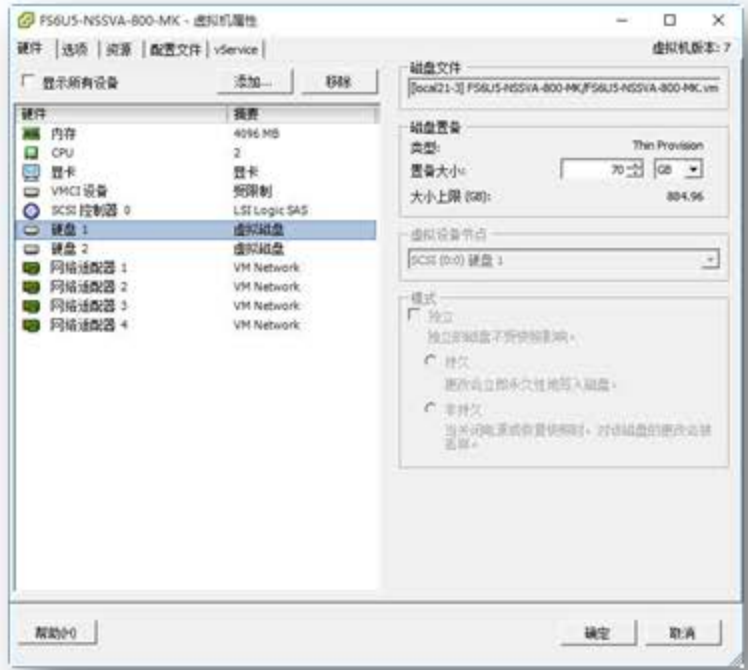
虚拟磁盘格式转换

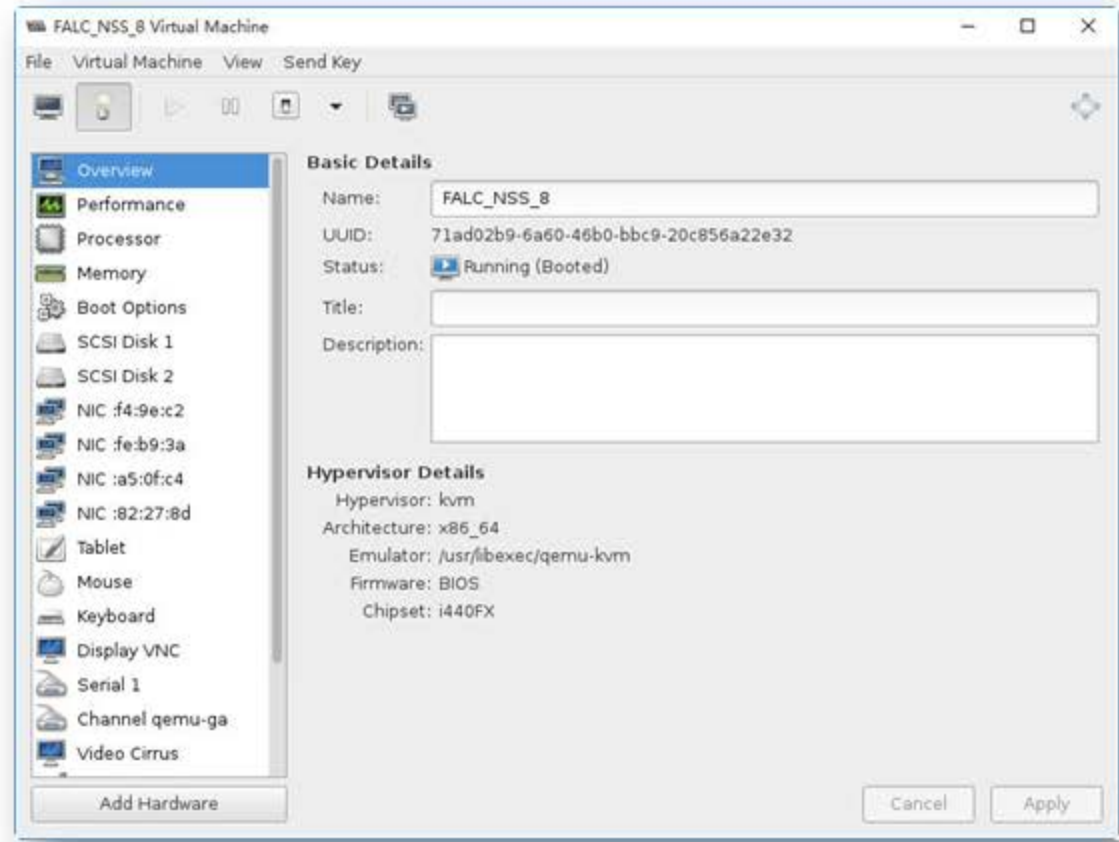
- 语法格式

```
convert [-c] [-p] [-q] [-n] [-f fmt] [-t cache] [-T
src_cache] [-O output_fmt] [-o options] [-s snapshot_name]
[-S sparse_size] filename [filename2 [...]] output_filename
```

```
# qemu-img convert my-vmware.vmdk my-kvm.img

# qemu-img convert -O qcow2 rhel6u3.img rhel6u3-a.img
```

演示：导入OVF格式的虚拟机到KVM





调整虚拟磁盘大小

- 语法格式

  `resize filename [+ | -]size`

- 操作之前，一定要做好数据备份
- 增加文件大小后，需要在客户机中使用fdisk、parted等分区工具进行相应的操作才能真正让客户机使用到增加后的镜像空间。
- 缩小镜像之前，要在客户机中保证里面的文件系统有空余空间，否则会数据丢失。
- qcow2不支持缩小镜像的操作。

```
# qemu-img resize crm-disk0-with-b.qcow2 +2G
Image resized.
```

快照管理

- 快照/检查点概述
- 磁盘快照分类
- 管理磁盘快照

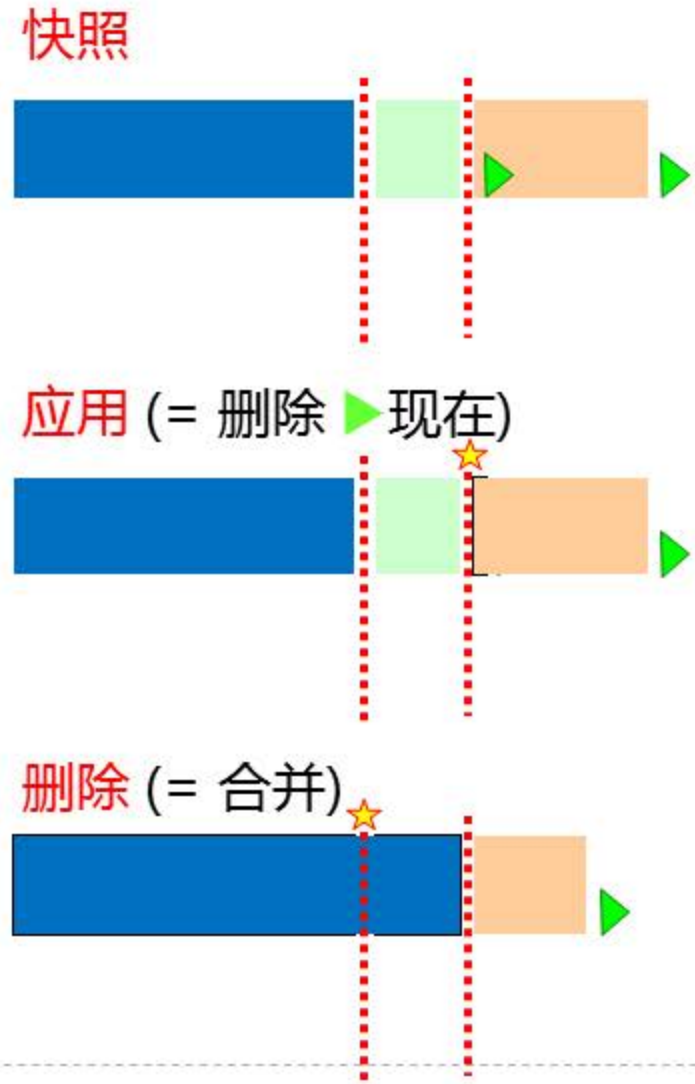
快照/检查点　Snapshot/Checkpoint

- 磁盘快照
  - 对磁盘数据进行快照
  - 主要用于虚拟机备份等场合
- 内存快照
  - 对虚拟机的内存/设备信息进行保存
  - 该机制同时用于休眠恢复，迁移等场景
  - 主要使用virsh save（qemu migrate to file）实现，只能对运行的虚拟机进行
- 检查点快照
  - 同时保存虚拟机的磁盘快照和内存快照
  - 用于将虚拟机恢复到某个时间点
  - 可以保证数据的一致性

磁盘快照分类

- 按快照信息保存分为：
  - 内置快照：快照数据和base磁盘数据放在一个qcow2文件中
  - 外置快照：快照数据单独的qcow2文件存放
- 按虚拟机状态可以分为:
  - 关机态快照：数据可以保证一致性
  - 运行态快照：数据无法保证一致性，类似与系统crash后的磁盘数据。使用是可能需要fsck等操作。
- 按磁盘数量可以分为:
  - 单盘：单盘快照不涉及原子性
  - 多盘：涉及原子性。主要分两个方面：1.是所有盘快照点相同 2.所有盘要么都快照成功，要么都快照失败。主要依赖于qemu的transaction实现

磁盘快照原理



调整虚拟磁盘大小

- 语法格式

```
snapshot [-l | -a snapshot | -c snapshot | -d snapshot ] filename
snapshot is the name of the snapshot to create, apply or delete
        -a  applies a snapshot (revert disk to saved state)
        -c  creates a snapshot
        -d  deletes a snapshot
        -l  lists all snapshots in the given image
```

```
# qemu-img snapshot -l vm1-disk1.qcow2
Snapshot list:
ID        TAG                 VM SIZE                DATE       VM CLOCK
1         s1                        0 2016-05-06 14:13:40   00:00:00.000
2         s2                        0 2016-05-06 14:28:52   00:00:00.000
3         s3                        0 2016-05-06 14:28:57   00:00:00.000
```

快照小结

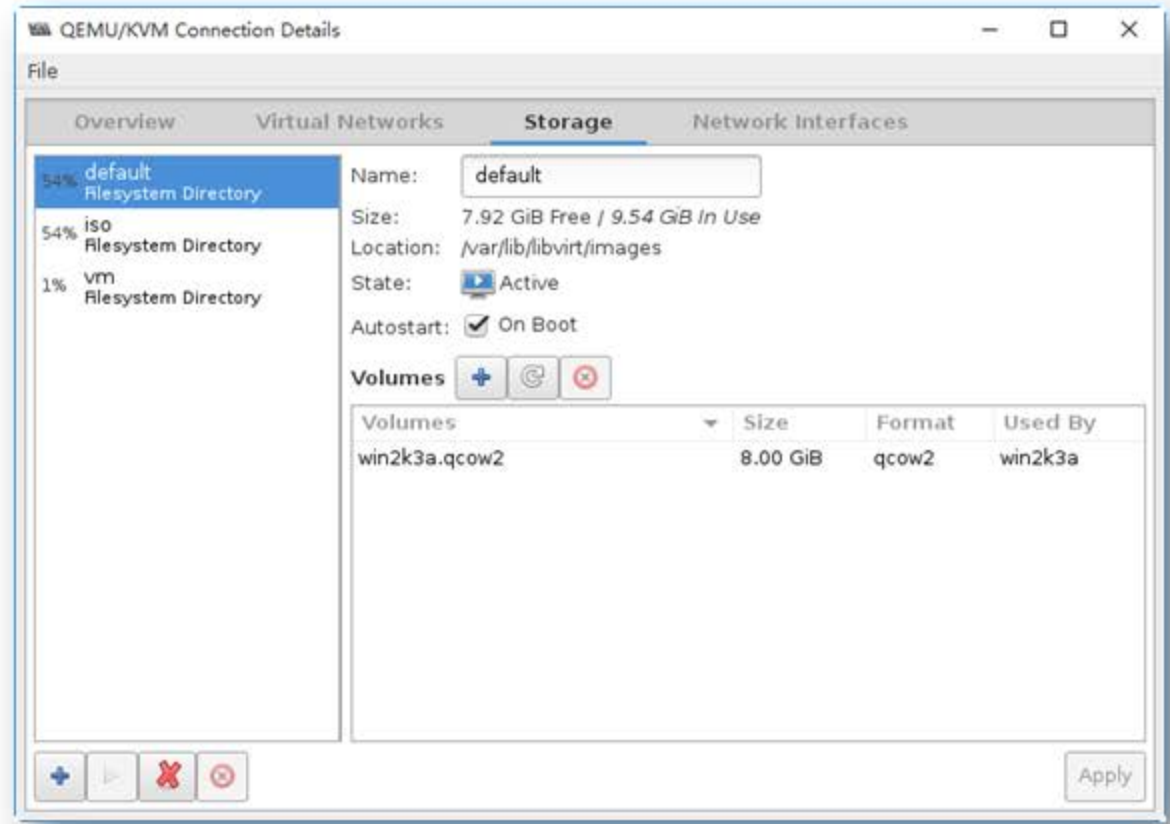
- 向虚拟机磁盘里写入文件，虚拟磁盘会变大。当删除该文件时候，虚拟磁盘大小依然不变。
- 当从一个原来的虚拟镜像base过来一个虚拟镜像，在首次启动新虚拟镜像之前，删除原来虚拟镜像某个文件，在新的虚拟镜像中一样看不到。在运行新的虚拟镜像后，在原始镜像中删除某个文件，在新的镜像中能看到，且依然能够访问。
- qemu-img snapshot 实现的是内部快照
- qemu-img backing_file实现的是外部快照
- libvirt的快照实现是在qemu的基础上实现的
- libvirt的外部快照实现可能使用了qemu的base，rebase，commit功能
- 从原代码来看一下libvirt创建非活动的内部快照其实调用了qemu-img snapshot功能。

存储池

- 基本概念
- 显示池与卷的信息
- 基于目录的存储池
- 基于磁盘的存储池
- 基于分区的存储池
- 基于LVM的存储池
- 构建实验用的集中存储环境
- 基于iSCSI的存储池
- 基于NFS的存储池

存储池的基本概念

- Libvirt可以以存储池的形式对存储进行统一管理、简化操作
- 对于虚拟机操作来说，存储池和卷并不是必需的
- 支持以下存储池
  - dir: Filesystem Directory
  - disk: Physical Disk Device
  - fs: Pre-Formatted Block Device
  - gluster: Gluster FileSystem
  - iscsi: iSCSI Target
  - logical: LVM Volume Group
  - mpath: Multipath Device Enumerator
  - netfs: Network Export Directory
  - rbd: RADOS Block Device/Ceph
  - scsi: SCSI Host Adapter
  - sheepdog: Sheepdog Filesystem



virsh中的存储池相关命令

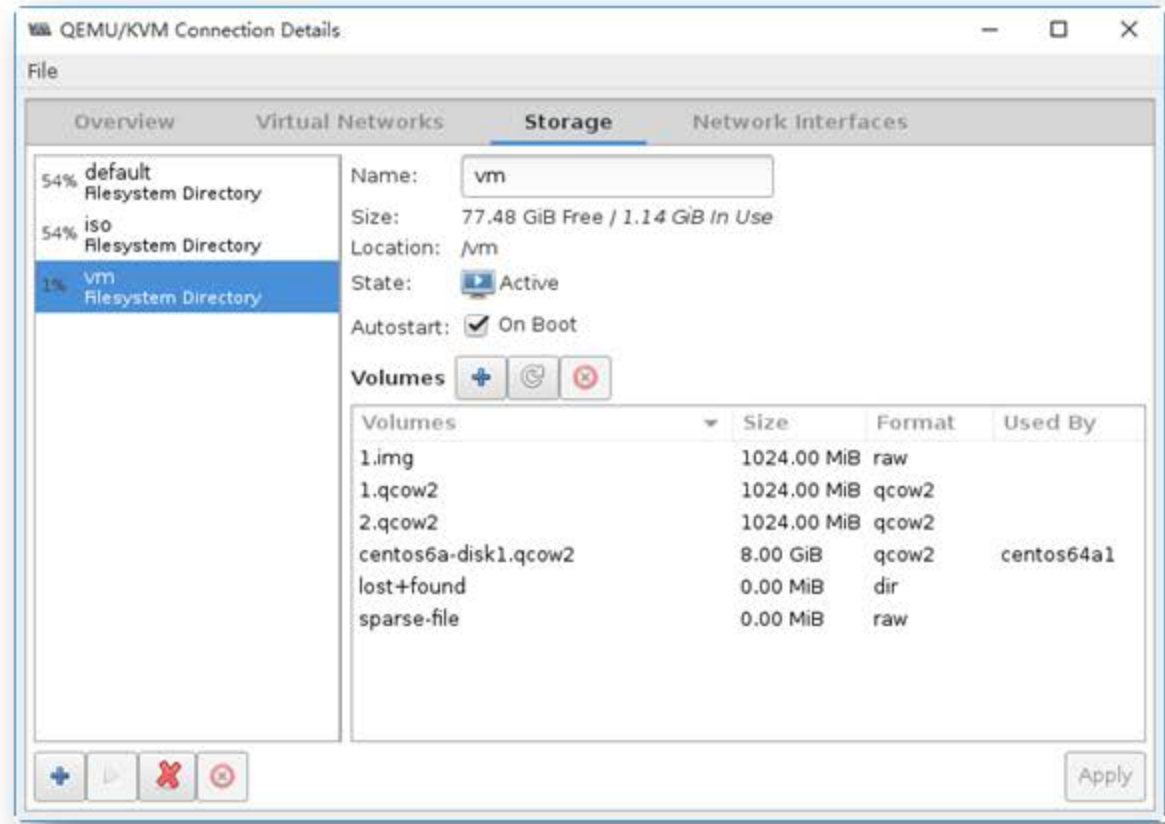
find-storage-pool-sources-as	通过参数查找存储池源find potential storage pool sources
find-storage-pool-sources	通过XML文档查找存储池源找到潜在存储池源
pool-autostart	自动启动某个池
pool-build	建立池
pool-create-as	从一组变量中创建一个池
pool-create	从一个 XML 文件中创建一个池
pool-define-as	在一组变量中定义池
pool-define	在一个 XML 文件中定义（但不启动）一个池或修改已经有池
pool-delete	删除池
pool-destroy	销毁（停止）池
pool-dumpxml	将池信息保存到XML文件中的
pool-edit	为存储池编辑 XML 配置
pool-info	存储池信息
pool-list	列出池
pool-name	将池 UUID 转换为池名称
pool-refresh	刷新池
pool-start	启动一个（以前定义的）非活跃的池
pool-undefine	取消定义一个不活跃的池
pool-uuid	把一个池名称转换为池 UUID

virsh中的存储卷相关命令

- **vol-clone** 克隆一个卷
- **vol-create** 从一个 XML 文件创建一个卷
- **vol-create-from** 使用另外一个卷做为输出，创建一个新卷
- **vol-create-as** 从一组变量中创建卷
- **vol-delete** 删除卷
- **vol-wipe** wipe一个卷
- **vol-dumpxml** 保存卷信息的信息到XML文件中
- **vol-info** 存储卷信息
- **vol-list** 列出卷
- **vol-pool** 根据卷的key或路径返存储池
- **vol-path** 根据卷名或key返回卷的路径
- **vol-name** 根据卷key或路径返回卷的名称
- **vol-key** 根据卷名或路径返回卷的key

显示池与卷的信息

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

virsh # pool-info vm
Name:           vm
UUID:           a9fddf2c-48d6-43b8-b970-d79308a501fc
State:          running
Persistent:     yes
Autostart:      yes
Capacity:       78.62 GiB
Allocation:     1.14 GiB
Available:      77.48 GiB

virsh #  vol-list vm
```



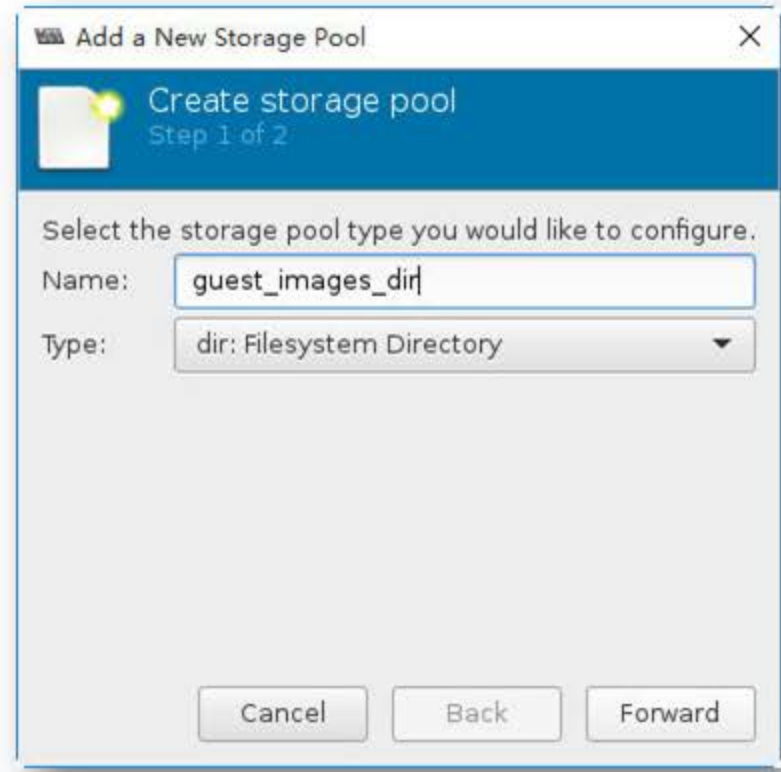
基于目录的存储池　　dir: Filesystem Directory

- 准备目录
  - 设置目录权限

```
# chown root:root /guest_images/
# chmod 700 /guest_images
```

  - 配置目录的SELinux上下文

```
# semanage fcontext -a -t virt_image_t \
'/guest_images(/.*)?'
```



- 通过virt-manager创建
- 通过virsh 创建

```
# virsh pool-define-as guest_images dir --target "/guest_images2"
```

基于分区的存储池　　fs: Pre-Formatted Block Device

- libvirtd会自动mount分区
- 准备分区并创建文件系统

```
# fdisk /dev/sdc
……
# mkfs.ext4 /dev/sdc1
```

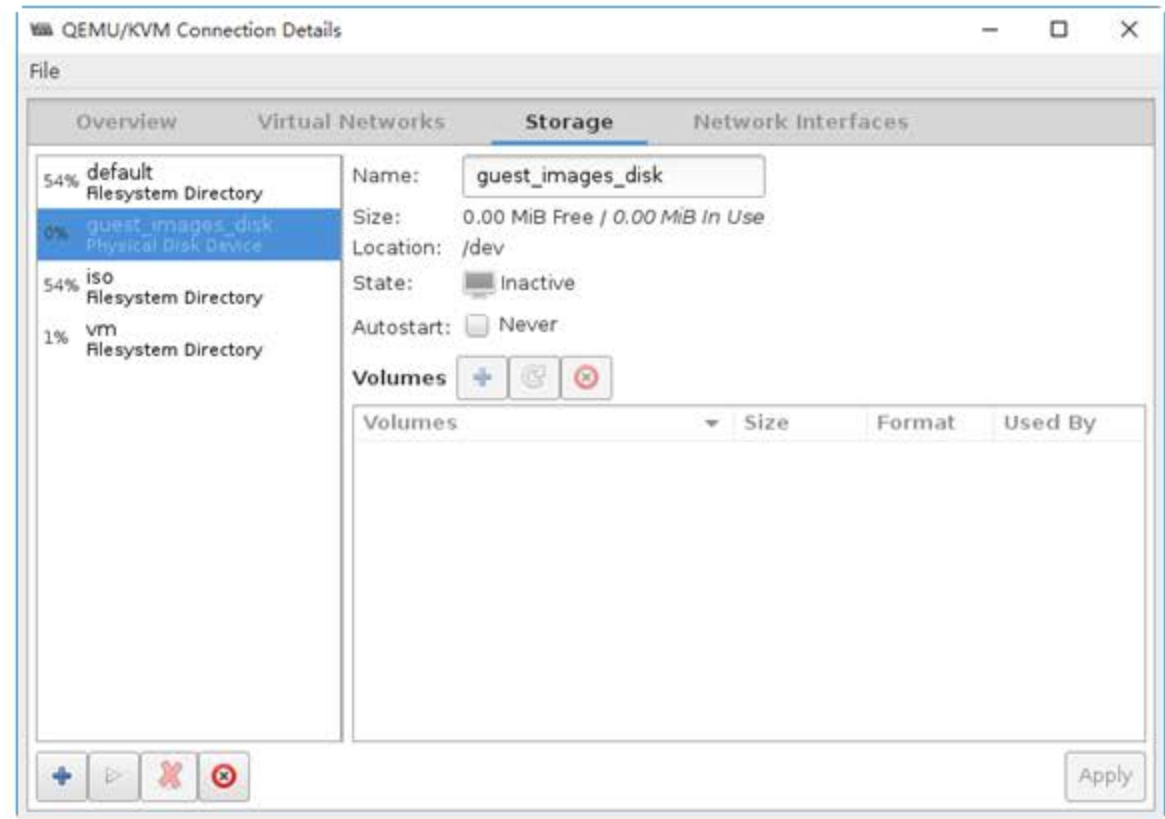
- 创建：
  - Source Path: 块设备名
  - Target Path：mount到的目录名

```
# virsh pool-define-as guest_images_fs fs \
--source-dev "/dev/sdc1" --target "/guest_images2"
```

基于磁盘的存储池　　disk: Physical Disk Device

- 准备XML文件

```
# vi /tmp/guest_images_disk.xml
添加如下的内容：
<pool type='disk'>
  <name>guest_images_disk</name>
  <source>
    <device path='/dev/sdc'/>
    <format type='gpt'/>
  </source>
  <target>
    <path>/dev</path>
  </target>
</pool>
```

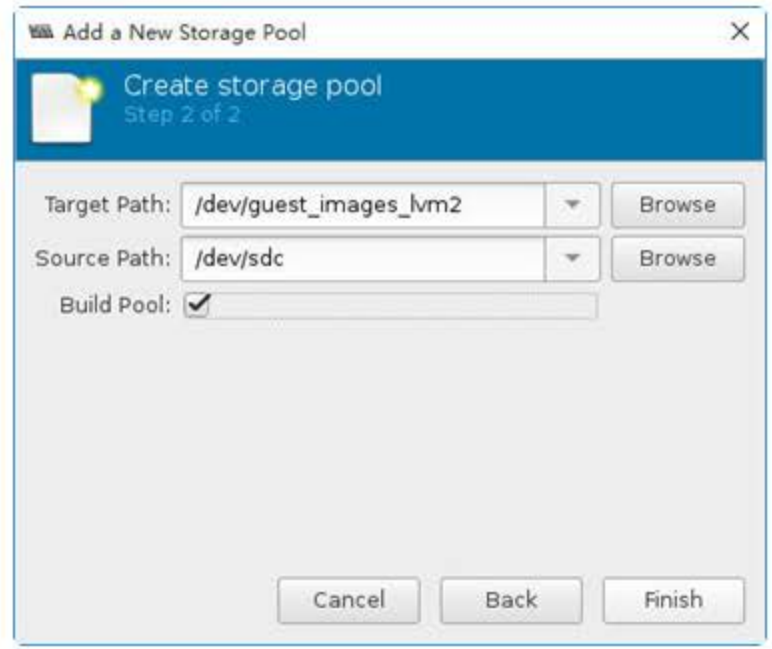


- 通过virsh 创建

```
# virsh pool-define /tmp/guest_images_disk.xml
```

基于LVM的存储池　　logical: LVM Volume Group

- 基于LVM的存储池要求使用全部磁盘分区
- 创建时存储池，有两种方法
  - 使用现有的VG
  - 创建新的VG
    - Target Path：新的卷组名
    - Source Path：存储设备的位置
    - Build Pool：会创建新的VG.
- 创建




```
# virsh pool-define-as guest_images_lvm3 logical \
--source-dev=/dev/sdc --source-name=libvirt_lvm \
--target=/dev/libvirt_vg
Pool guest_images_lvm3 defined
```

实验环境准备：安装Linux的存储服务器

- 最小化安装的Linux
- 安装targetcli软件包

  `# yum -y install targetcli`
- 使用targetcli配置存储

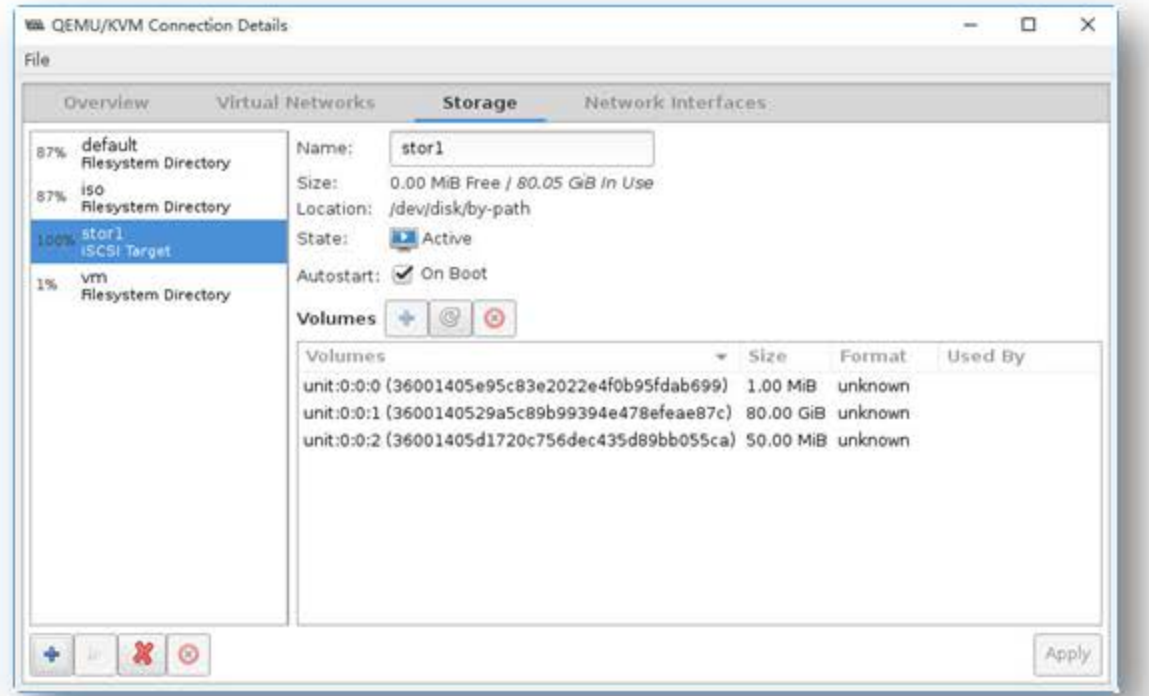
```
[root@stor1 ~]# targetcli
targetcli shell version 2.1.fb41
Copyright 2011-2013 by Datera, Inc and others.
For help on commands, type 'help'.

/> ls
o- / ......................................................... [...]
  o- backstores .............................................. [...]
  | o- block .................................. [Storage Objects: 1]
  | | o- block1 ........... [/dev/vdb1 (80.0GiB) write-thru activated]
  | o- fileio ................................. [Storage Objects: 1]
  | | o- fileio1 ........... [/foo.img (50.0MiB) write-back activated]
  | o- pscsi .................................. [Storage Objects: 0]
  | o- ramdisk ................................ [Storage Objects: 1]
  |   o- ramdisk1 ............................. [(1.0MiB) activated]
  o- iscsi ................................................ [Targets: 1]
  | o- iqn.2010-05.com.linuxplus.srv1:stor1 ................ [TPGs: 1]
  |   o- tpg1 ................................ [no-gen-acls, no-auth]
  |     o- acls ..................................................... [ACLs: 1]
  |     | o- iqn.1994-05.com.redhat:ctkvm2 .......... [Mapped LUNs: 3]
  |     |   o- mapped_lun0 ............... [lun0 ramdisk/ramdisk1 (rw)]
  |     |   o- mapped_lun1 ................... [lun1 block/block1 (rw)]
  |     |   o- mapped_lun2 ................ [lun2 fileio/fileio1 (rw)]
  |     o- luns ..................................................... [LUNs: 3]
  |     | o- lun0 ................................ [ramdisk/ramdisk1]
  |     | o- lun1 ........................ [block/block1 (/dev/vdb1)]
  |     | o- lun2 ....................... [fileio/fileio1 (/foo.img)]
  |     o- portals ............................................... [Portals: 1]
  |       o- 0.0.0.0:3260 ..................................................... [OK]
  o- loopback ............................................. [Targets: 0]
/>
```

```
auth --enableshadow --passalgo=sha512
cdrom
text
firstboot --enable
ignoredisk --only-use=vda
keyboard --vckeymap=us --xlayouts='us'
lang en_US.UTF-8
network --onboot yes --bootproto dhcp --noipv6
network  --hostname=localhost.localdomain
rootpw 123456
firewall --disable
selinux --disable
timezone --utc Asia/Shanghai
bootloader --location=mbr --boot-drive=vda
autopart --type=lvm
clearpart --none --initlabel
reboot
%packages
@core
@base
net-tools
%end
```

实验：配置Target

1. 创建存储对象
   - 创建块存储对象　　create block1 dev=/dev/vdb1
   - 创建fileio对象　　create fileio1 /foo.img 50M
   - 创建ramdisk对象　　create ramdisk1 1M
2. 创建iSCSI Target　　create iqn.2010-05.org.linuxplus.srv1:stor1

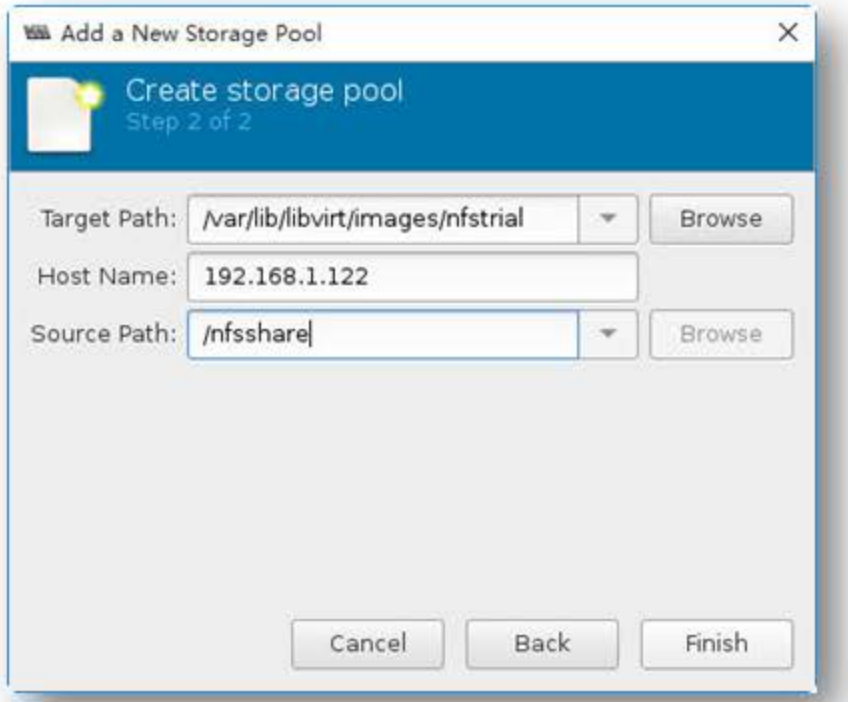
基于iSCSI的存储池　　iscsi: iSCSI Target

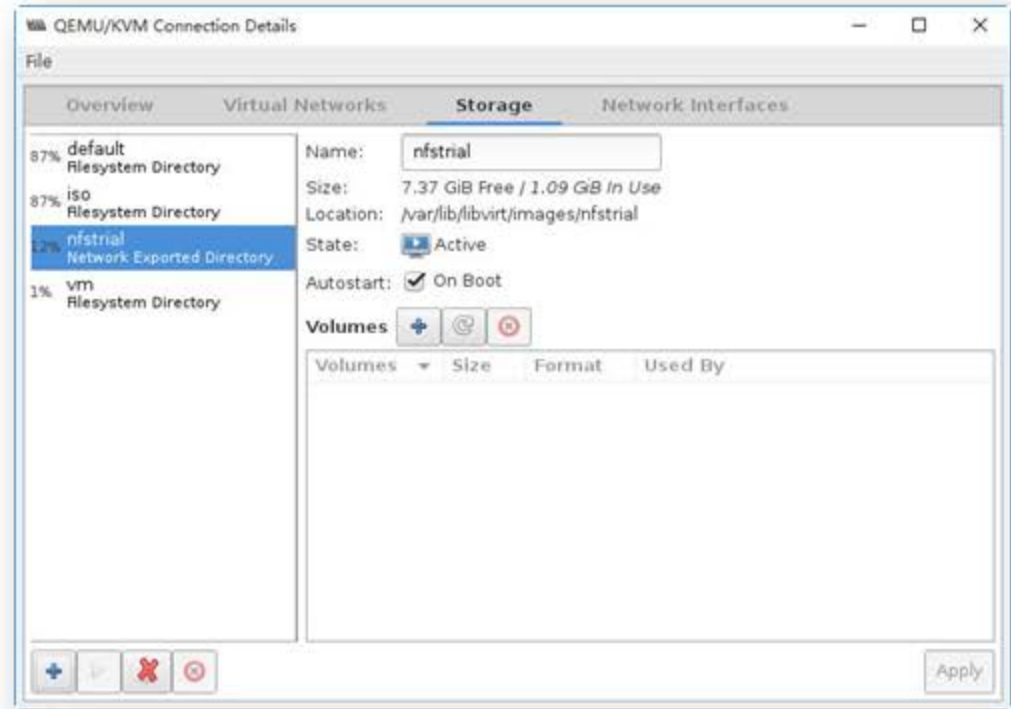



```
# virsh pool-define-as --name stor2 --type iscsi \
--source-host 192.168.1.122 \
--source-dev iqn.2010-05.com.linuxplus.srv1:stor1 \
--target /dev/disk/by-path
```

LinuxPlus

基于NFS的存储池　netfs: Network Export Directory

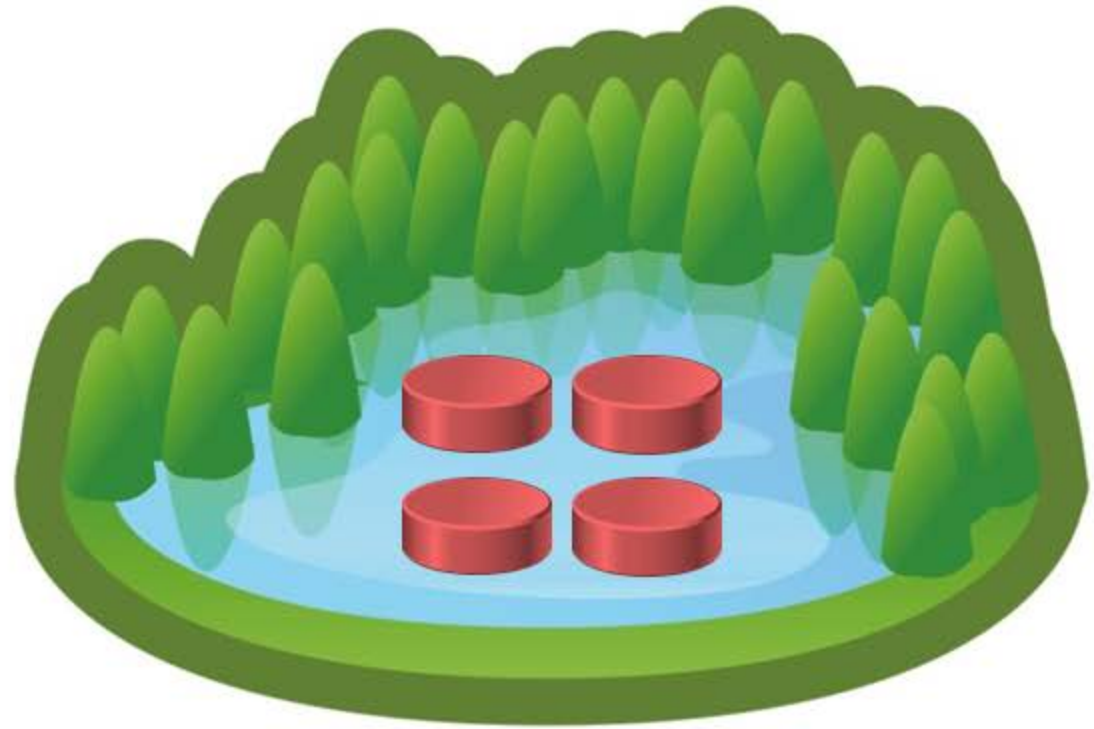





```
# virsh pool-define-as --name nfstrial2 --type netfs \
--source-host 192.168.1.122 \
--source-path /nfsshare \
--target /nfstrial2
```

存储卷

- 存储卷概述
- 存储卷管理
- 向虚拟机添加卷



存储卷概述

- 存储池被分割为存储卷（Storage Volume）
- 存储卷
  - 文件
  - 块设备（如物理分区、LVM逻辑卷等）
  - libvirt管理的其他类型存储的抽象

vol-clone	克隆卷
vol-create-as	通过一组参数创建卷
vol-create	通过XML文件创建卷
vol-create-from	通输入的其他卷创建一个新的卷
vol-delete	删除一个卷
vol-download	下载卷的内容到一个文件
vol-dumpxml	XML格式的卷信息
vol-info	存储卷的信息
vol-key	根据卷名或路径返回卷的key
vol-list	列出卷
vol-name	根据卷的key或路径返回卷名
vol-path	根据卷名或key返回卷的路径
vol-pool	根据卷的key或路径返回存储池
vol-resize	调整卷大小
vol-upload	上传文件内容到一个卷
vol-wipe	wipe a vol

存储卷管理

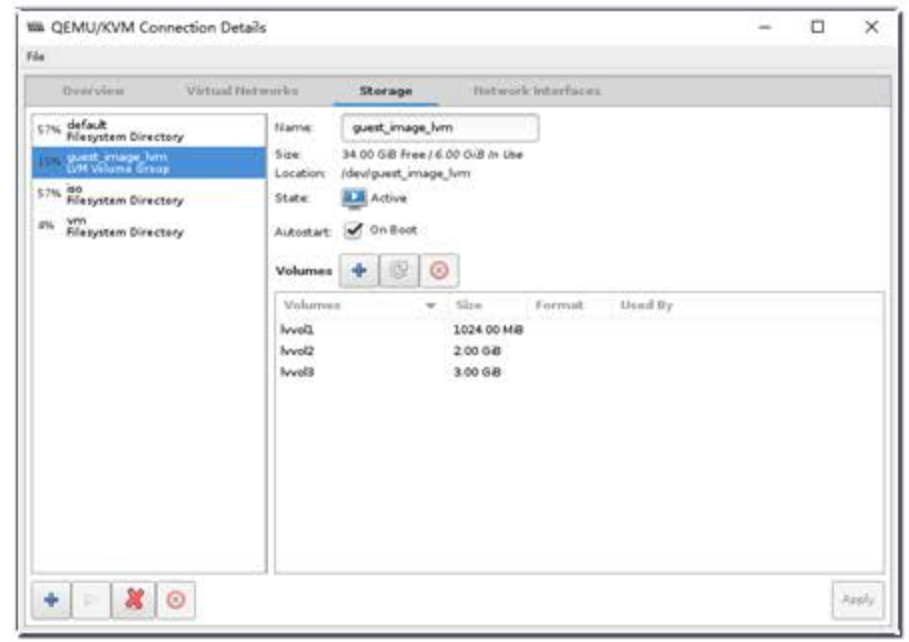
- 创建
- 克隆
- 删除
- 移动
- 修改大小

```
# virsh vol-create-as -help

vol-create-as <pool> <name> <capacity> [--allocation <string>] [--
format <string>] [--backing-vol <string>] [--backing-vol-format
<string>] [--prealloc-metadata]

  OPTIONS
    [--pool] <string>  pool name
    [--name] <string>  name of the volume
    [--capacity] <string>  size of the vol, as scaled integer (default
bytes)
    --allocation <string>  initial allocation size, as scaled integer
(default bytes)
    --format <string>  file format type raw,bochs,qcow,qcow2,qed,vmdk
    --backing-vol <string>  the backing volume if taking a snapshot
    --backing-vol-format <string>  format of backing volume if taking
a snapshot
    --prealloc-metadata  preallocate metadata (for qcow2 instead of
full allocation)
```

演示：存储卷管理

- 基本目录的存储池中的存储卷管理
- 基本LVM的存储池中的存储卷管理

```
virsh # vol-list vm
Name                 Path
-----------------------------------------
centos6a-disk0.qcow2 /vm/centos6a-disk0.qcow2
erp-disk0.qcow2      /vm/erp-disk0.qcow2
lost+found           /vm/lost+found
oa-disk0.qcow2       /vm/oa-disk0.qcow2
t1.img               /vm/t1.img
test1.qcow2          /vm/test1.qcow2
test2.qcow2          /
```

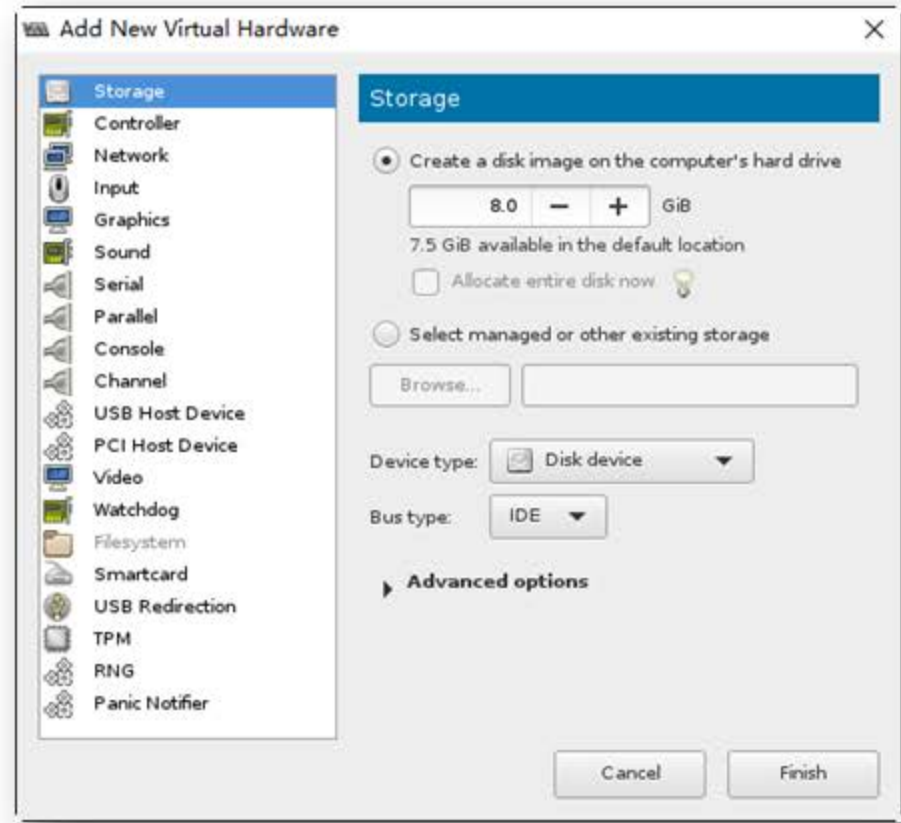


向虚拟机添加卷

- attach-device
  - 通过XML添加新的设备

```
<disk type='file' device='disk'>
   <driver name='qemu' type='raw' cache='none'/>
   <source file='/vm/testdisk1.img'/>
   <target dev='vdb'/>
</disk>
```

```
<disk type='file' device='cdrom'>
   <driver name='qemu' type='raw' cache='none'/>
   <source file='/iso/CentOS.iso'/>
   <readonly/>
   <target dev='hdc'/>
</disk >
```



- attach-disk
  - 通过参数添加新的磁盘设备

```
# virsh attach-disk --domain VM1  --source=/vm/t2.img --target=vdb --cache=none
```

虚拟磁盘离线访问工具

- 离线访问工具应用场景
- guestfish Shell常用操作
- 其他离线工具简介

离线访问工具应用场景

- 观看或下载位于虚拟机磁盘中的文件
- 编辑或上传文件到虚拟机磁盘
- 读取或写入的虚拟机配置。
- 准备新的磁盘映像，其中包含文件、目录、文件系统、分区、逻辑卷和其他选项
- 拯救和修复客户无法启动或需要更改启动配置的虚拟机
- 监控虚拟机的磁盘使用情况
- 根据组织安全标准审计虚拟机的合规性
- 通过克隆和修改模板来部署虚拟机
- 读取CD和DVD ISO和软盘映像

guestfish Shell常用操作

- Libguestfs提供了一个简单地访问虚机磁盘镜像文件的方法，即使是在虚机无法启动的情况下
- Libguestfs是由一组丰富的工具集组成，可以让管理员访问虚机文件，甚至调整和挽救文件。
- guestfish是一个基于libguestfs API的交互shell

```
# guestfish --ro -a /vm/centos6a-disk1.qcow2

Welcome to guestfish, the guest filesystem shell for
editing virtual machine filesystems and disk images.

Type: 'help' for help on commands
      'man' to read the manual
      'quit' to quit the shell

><fs> 有命令自动补全功能
```

其他离线工具简介

- virt-df
  - 监视磁盘使用
- virt-resize
  - 离线调整虚拟磁盘大小
- virt-inspector
  - 虚拟机检视
- virt-win-reg
  - Windows注册表读取和修改
- virt-sysprep
  - 虚拟机设置重置

总结

- 虚拟磁盘概述
- 使用qemu-img管理虚拟磁盘
- 快照管理理
- 存储池
- 存储卷
- 虚拟磁盘离线访问工具