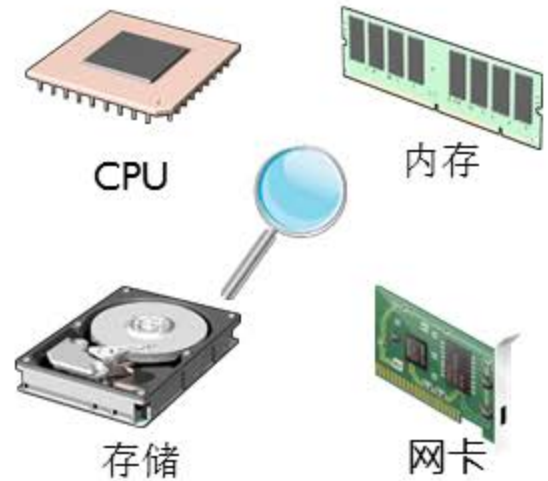
性能监视与优化

概述

- 性能监视与优化方法论
- 监视工具
- RHEL/CentOS 7 KVM性能特性和改进
- virt-manager中的性能管理
- 网络性能优化
- 存储性能优化
- 内存性能优化
- CPU性能优化

性能监视与优化方法论

- 为什么要监视性能?
- 建立性能基线
- 服务器性能监视的最佳策略



为什么要监视性能?

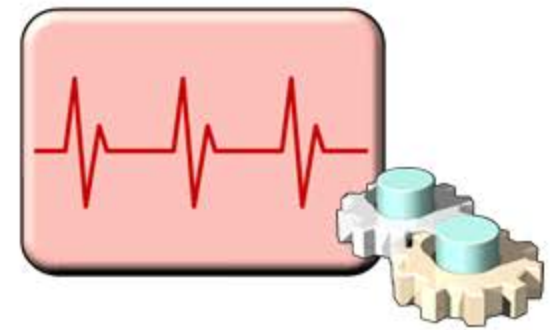
- 通过监视性能，你可以:
  - 了解服务器的工作负荷以及对服务器资源的影响
  - 了解性能的改变和性能趋势以便及采取措施
  - 测试调整结果
  - 诊断系统问题并确定优化措施
- 分析性能数据确定系统瓶颈

建立性能基线

- 性能基线是在一段时间中在典型的工作负荷和用户连接数量的情况下收集的服务器性能数据
- 在确定性能基线时，应当了解服务器所执行的任务，以及执行任务的时间和日期
- 在部署阶段建立性能基线，然后和实际性能进行比较
- 及早建立性能基线有助于快速发现和解决性能瓶颈问题

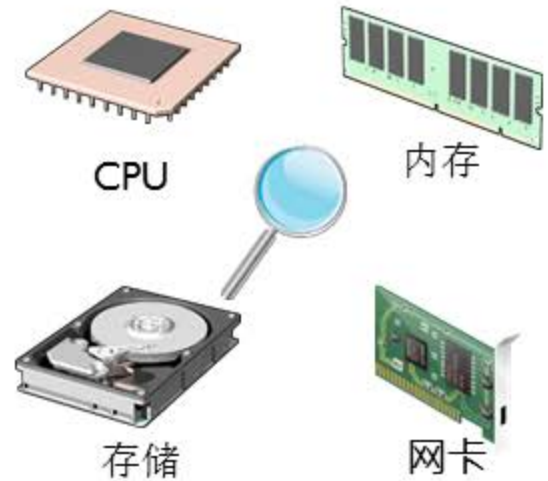
服务器性能监视的最佳策略

- 建立性能日志
- 尽量减少性能监视本身对服务器所造成的影响
- 分析监视结果，建立性能基线
- 创建警报
- 调整系统
  - Scale Up
  - Scale Out
- 分析性能趋势，提前采取措施



监视工具

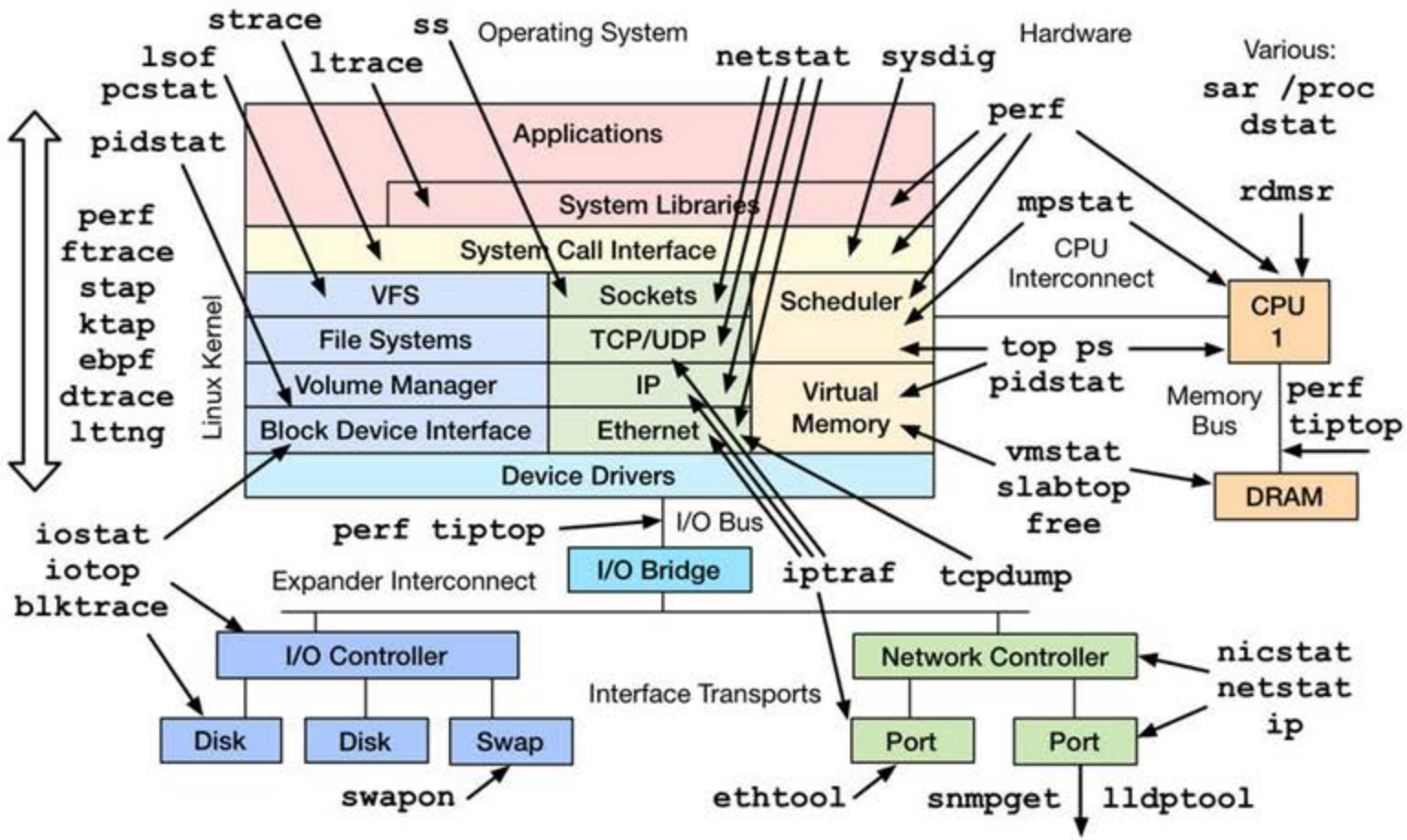
- 工具选择
- 常见工具



工具选择

- 您熟悉的
- 容易得到的
- 针对“四大”资源子系统

常见工具



图片来源：百度文库中的《Linux操作系统性能监控工具和指标分析V1.0》

RHEL/CentOS 7 KVM性能特性和改进

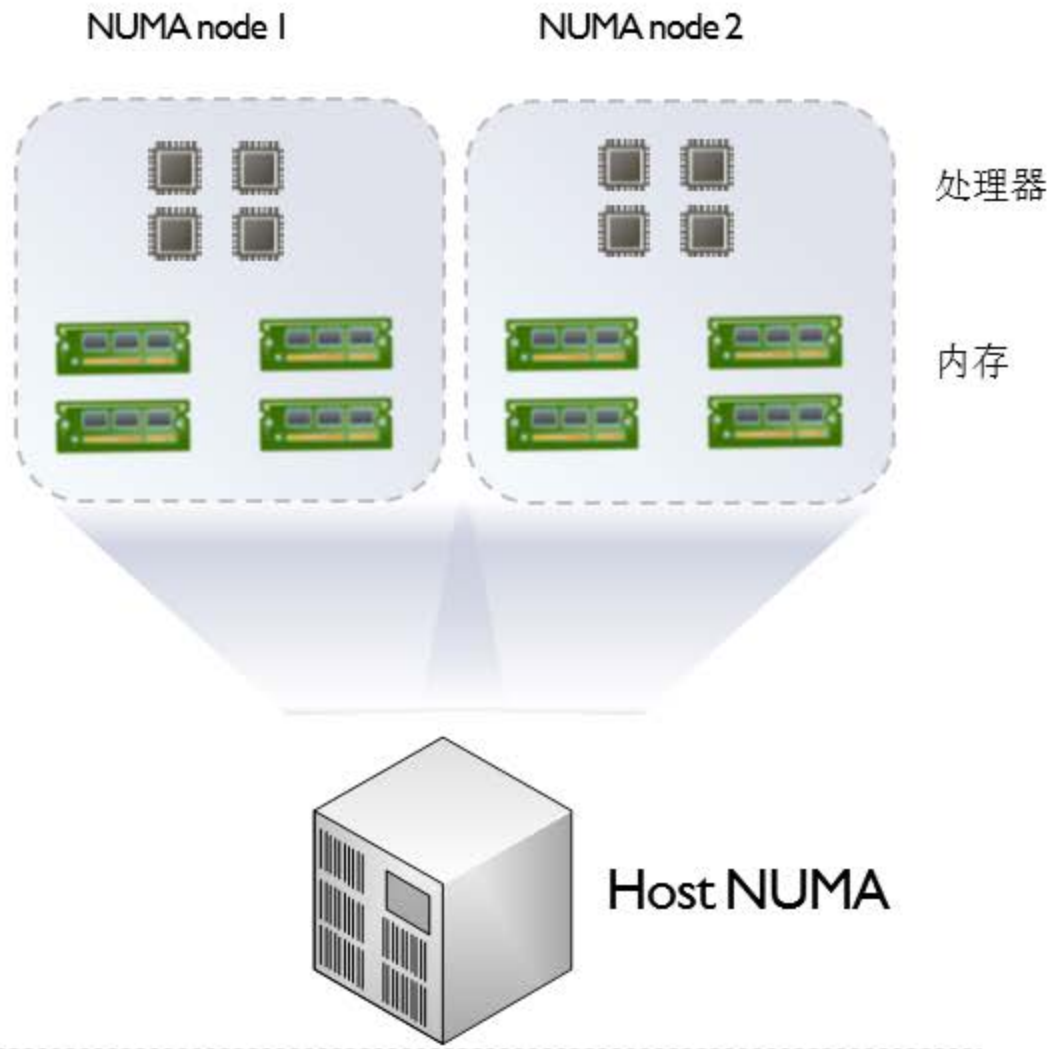
- RHEL/CentOS 7 中虚拟化性能特性
- RHEL/CentOS 7 中虚拟化性能改进

Red Hat Enterprise Linux 中的虚拟化性能特性

- CPU/Kernel
  - NUMA：非一致性内存访问
  - CFS：完全公平调度程序。一种新的、使用类（class）的调度程序。
  - RCU：读取复制更新。更好地处理共享线程数据
  - 多达 160 种虚拟 CPU（vCPU） 参考：https://access.redhat.com/articles/rhel-kvm-limits
- 内存
  - 大页面和其他内存密集型环境下的优化
- 网络
  - vhost-net：一种基于内核的 VirtIO 快速解决方案。
  - SR-IOV：使联网性能级别接近本机。
- 块 I/O
  - AIO：支持线程和其他 I/O 操作重叠。
  - MSI ：PCI 总线设备中断生成。
  - 磁盘 I/O 节流：虚拟机磁盘 I/O 的控制要求避免过度使用主机资源

物理NUMA

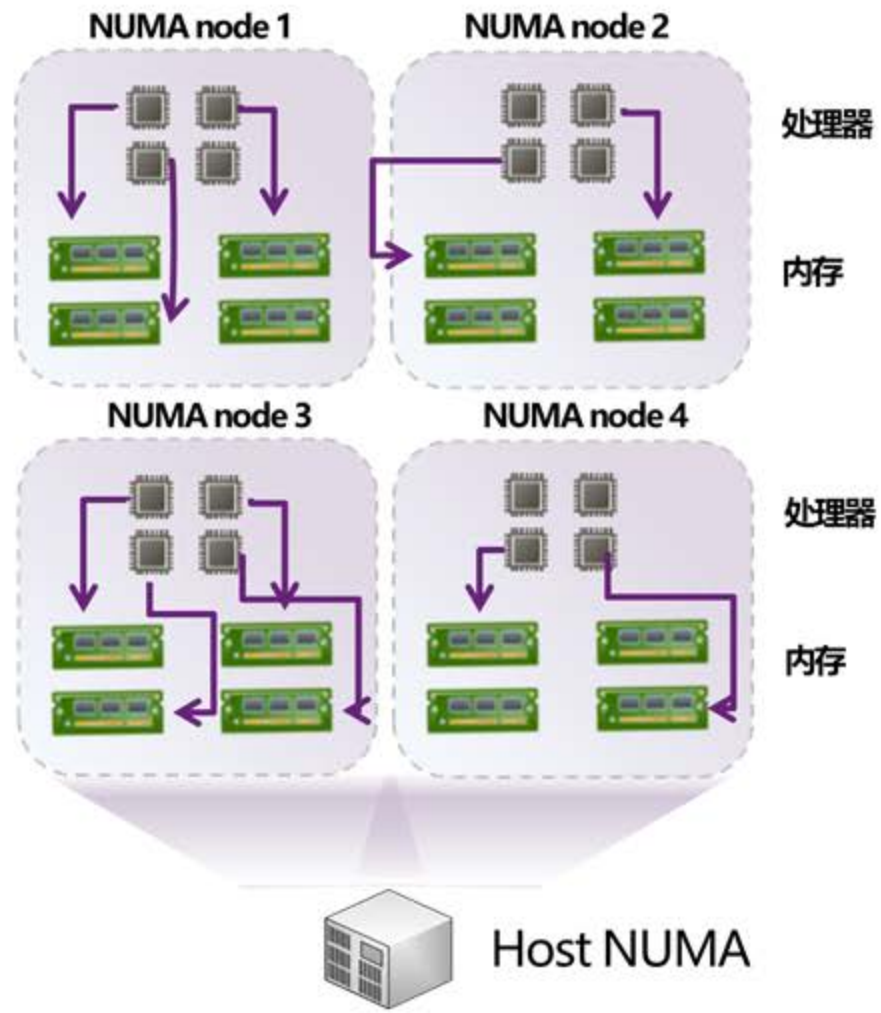
- Non-uniform memory access
  非一致性内存访问
- 帮助主机可以访问更多的内核和内存
- 将内核和内存划分为多个节点nodes
- 分配和延迟取决于内存与处理器的相对位置
- 高性能应用程序探测NUMA 并最小化跨节点(cross-node ）内存访问



物理NUMA

这是最理想的……

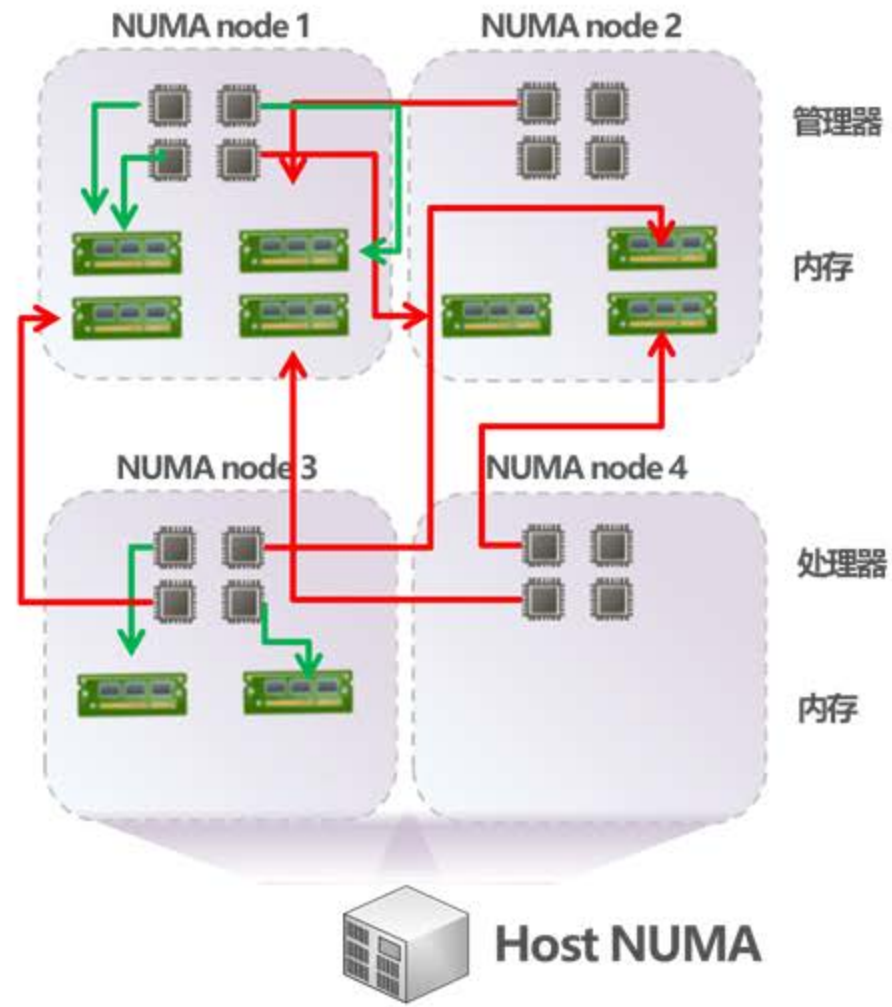
- 内存分配和线程分配在相同NUMA节点内
- 在每个NUMA 节点中配置内存



物理NUMA

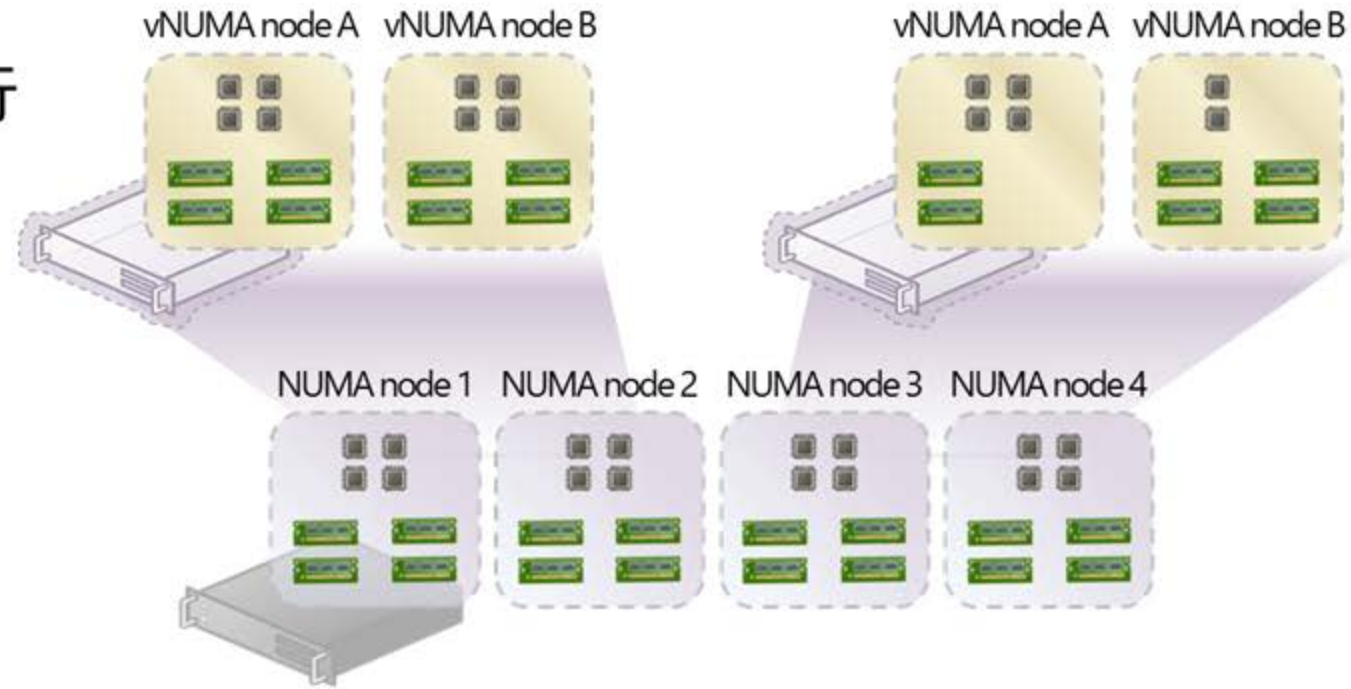
这是最糟糕的...

- 系统不均衡
- 内存分配和线程分配跨不同的NUMA节点
- 多个节点来回跨越
- NUMA 节点2 有奇数个DIMM
- NUMA 节点3 没有足够内存
- NUMA 节点4 无本地内存（最糟糕的的情况）

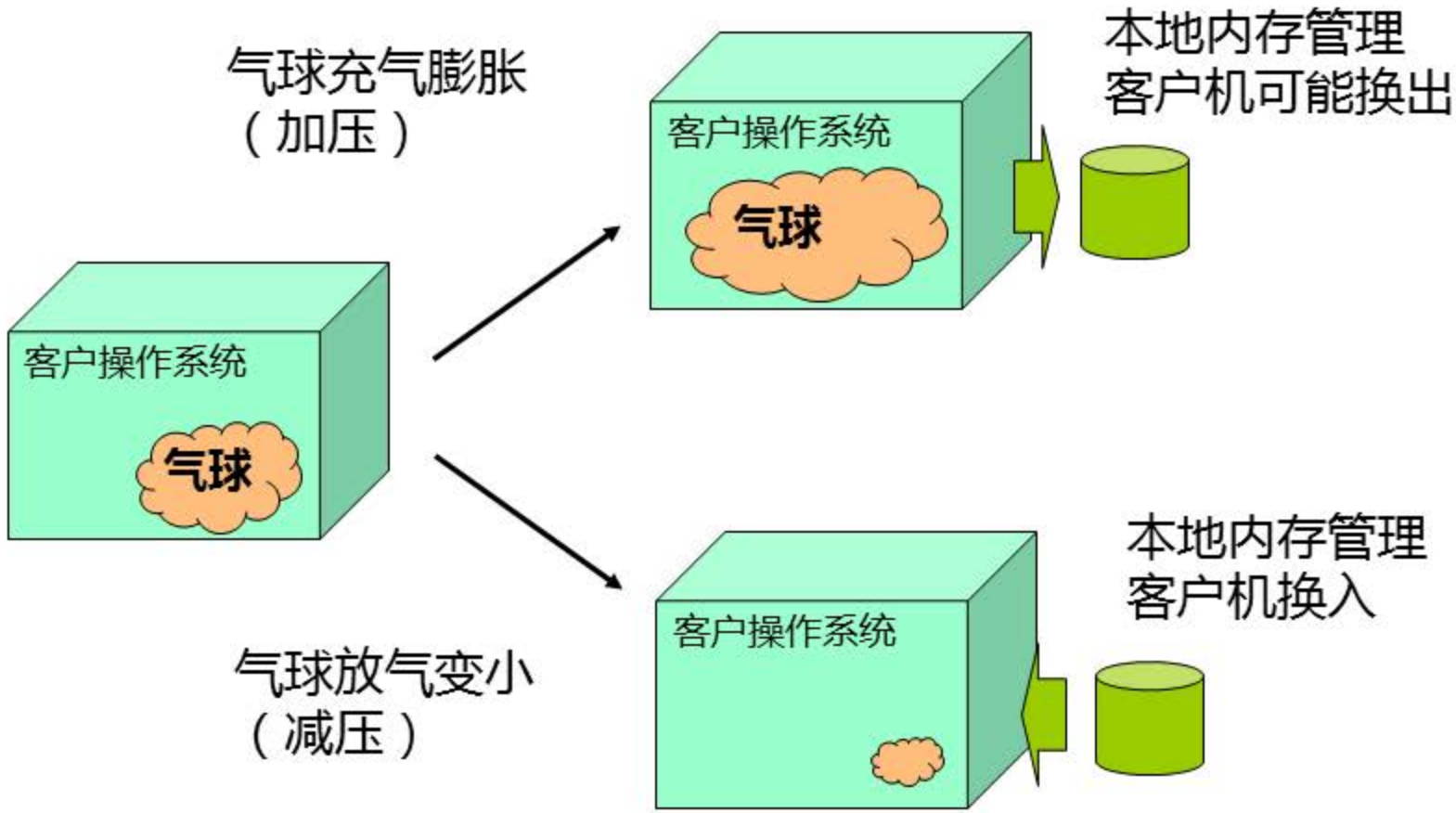


Guest NUMA

- Guest NUMA
  - 在VM 内表示NUMA 拓扑
  - Guest操作系统与应用可以进行关于线程和内存分配的智能NUMA决策
- Guest NUMA 节点与主机资源匹配
- RHEL/CentOS 7 KVM 的自动化 NUMA 平衡功能

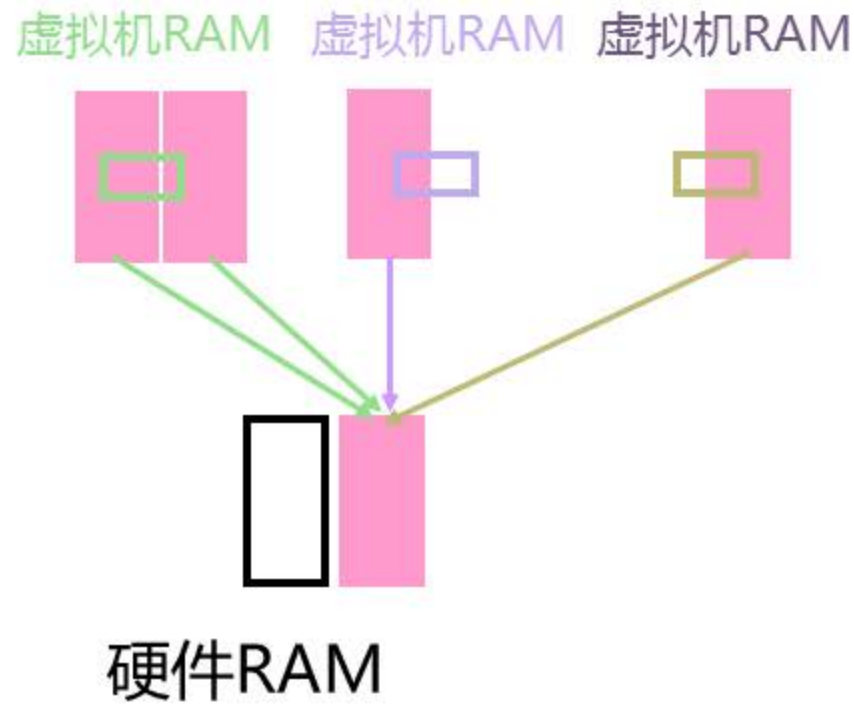


内存的气球控制技术（ Memory Ballooning ）



透明页面共享 Transparent Page Sharing (TPS)

- 多台虚拟计算机
  - 冗余的代码和数据
  - 举例说明： 相同操作系统
- 共享相同的页面
  - 复制检测算法
  - 举例说明： 超过30%共享/空闲Win虚拟计算机
- 写时才拷贝（Copy-on-write, COW）
  - 共享时只读
  - 专用的写时才拷贝模式



磁盘I/O控制

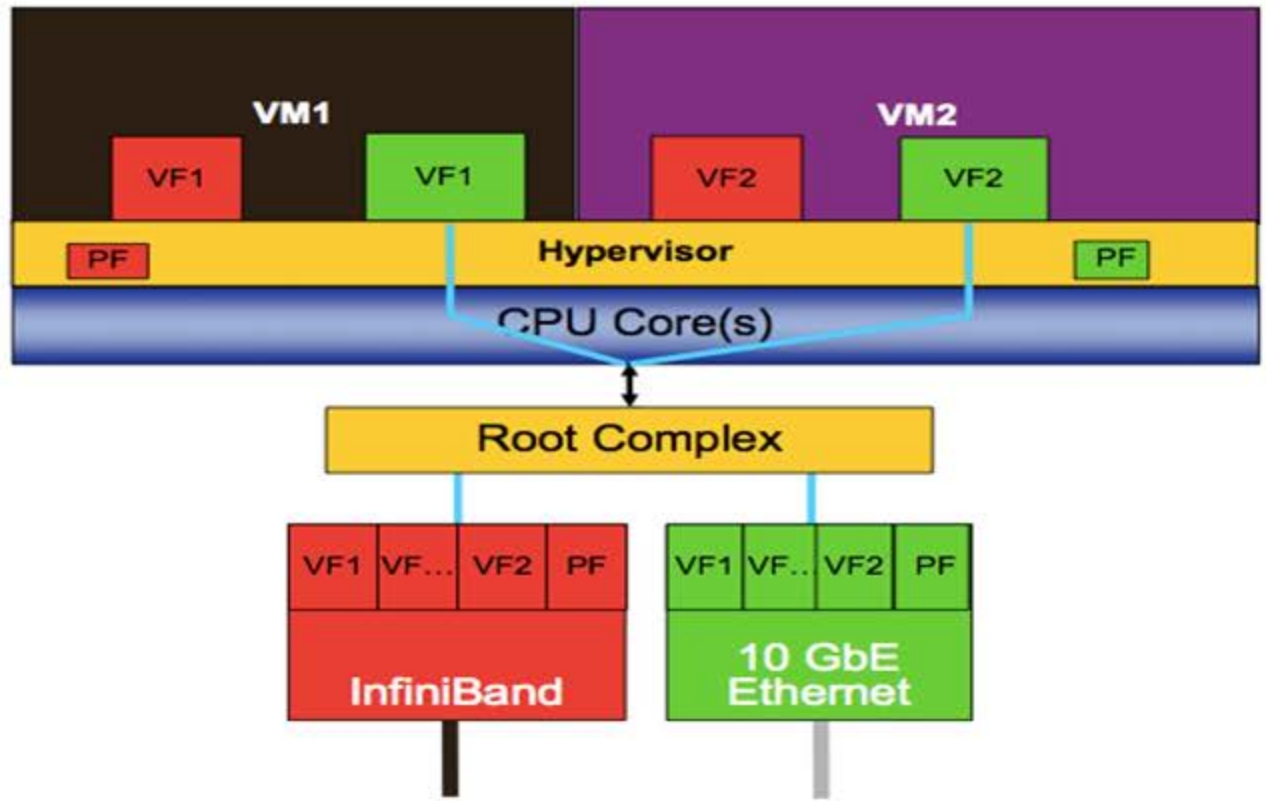
- 磁盘的I/O控制可避免某个虚拟机过度使用主机资源，有两种方式：
  - 设置磁盘的权重：在整体中的权重，数值范围在100-1000。
  - 限制具体的IO
- 通来`virsh blkiotune`实现，或修改虚拟机的XML文件

```
# virsh blkiotune 25 --weight 700 -live

# virsh blkdeviotune 8 hda \
 --total_bytes_sec 10240 \
 --total-bytes-sec 10240 \
 --read-bytes-sec 4096 \
 --write-bytes-sec 2028 \
 --total-iops-sec 30 \
 --read-iops-sec 15 \
 --write-iops-sec 15
```

Single-Root I/O虚拟化

- PCI-SIG 为虚拟化指定多个功能元素以解决性能和安全的问题
- PCIe 设备拥有多虚拟化功能(VF)



RHEL/CentOS 7 中虚拟化性能的改进

- 自动化 NUMA 平衡
- 多队列 virtio-net
- 桥接零复制传输
- APIC 虚拟化
- EOI 加速
- 多队列 virtio-scsi
- 半虚拟化 ticketlocks
- 半虚拟化页面错误
- 半虚拟化时间 vsyscall 优化
- ……

virt-manager中的性能管理

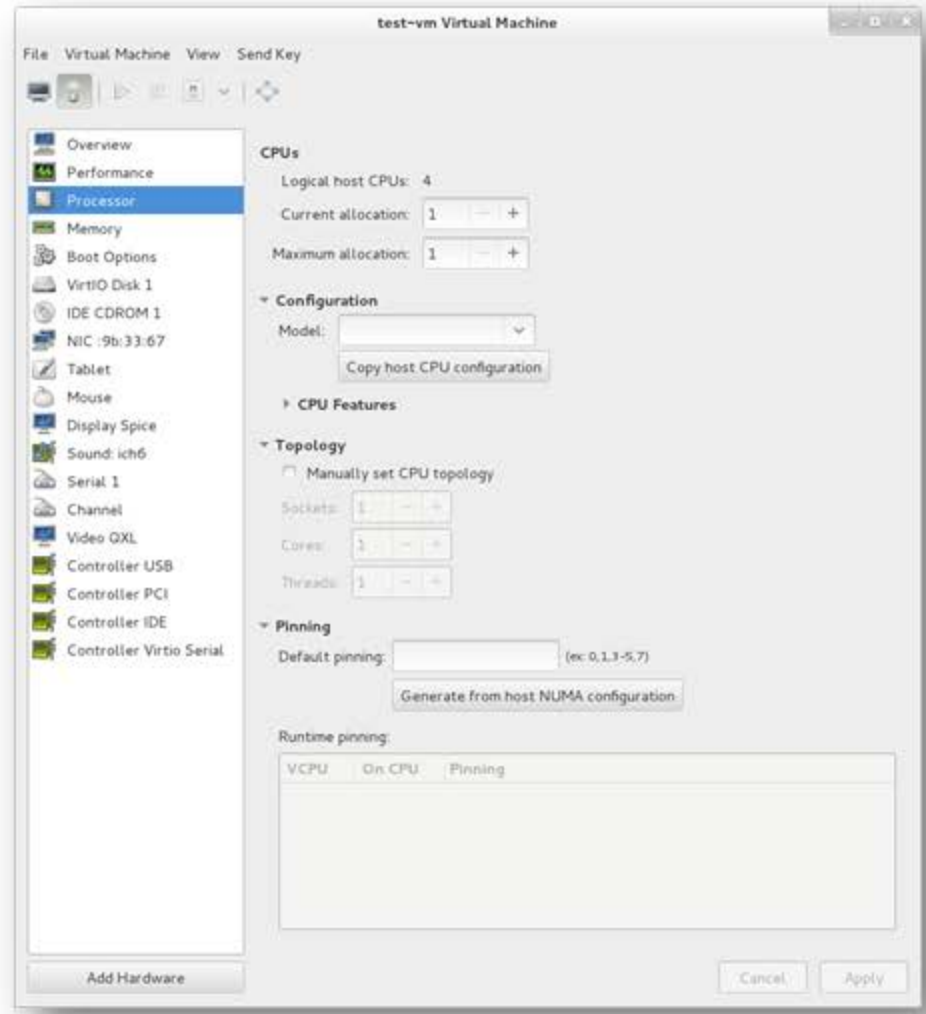
- 虚拟机操作系统细节和设备
- CPU 性能选项
- 虚拟磁盘性能选项

虚拟机操作系统细节和设备

- 尽可能地提供客户虚拟机详细信息，以便通过启用适用于特定客户虚拟机类型的功能来提高性能
- 删除不使用的设备
  - 例如：声卡

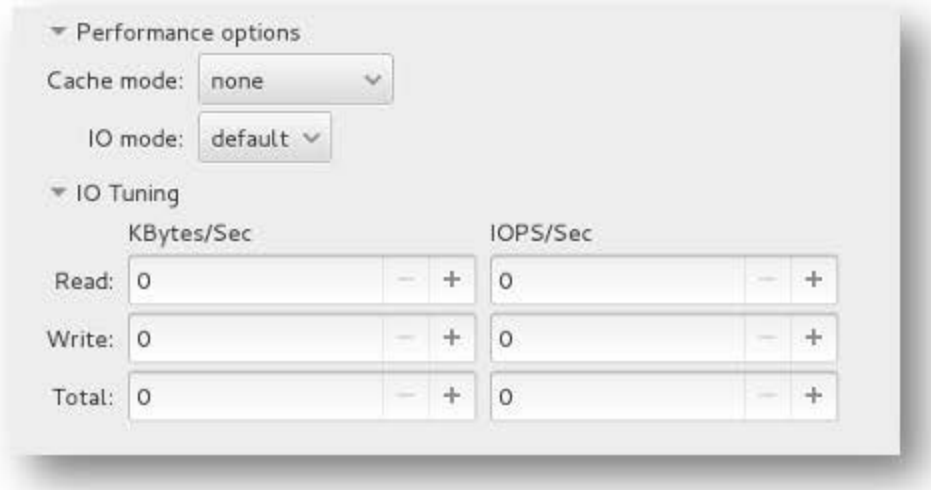
CPU 性能选项

- 可用的 CPU
- CPU 配置
- CPU 拓扑
- CPU 钉选（pinning）



虚拟磁盘性能选项

- 缓存模式
- IO 模式
- IO 调整



网络性能优化

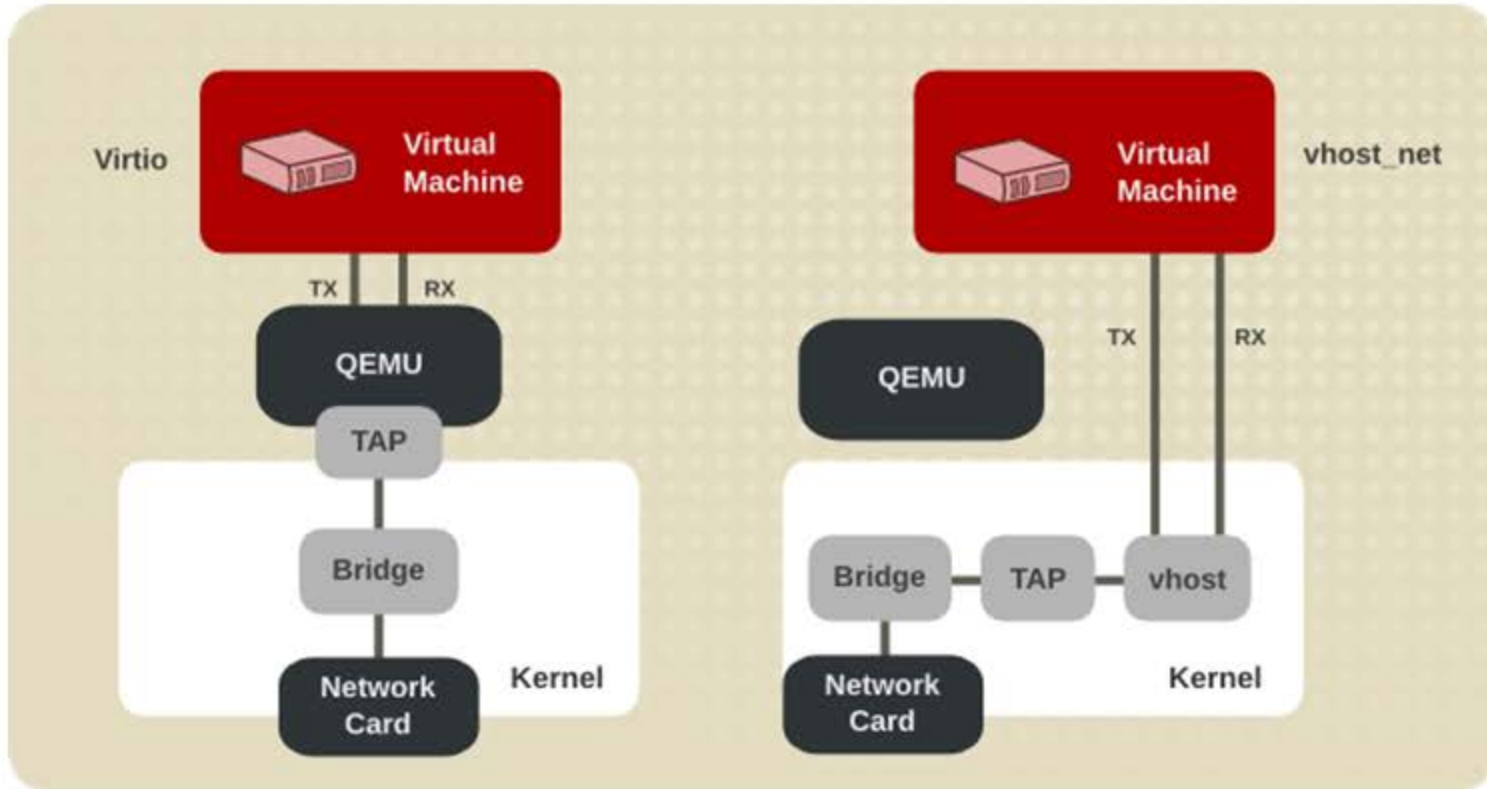
- 常规建议
- VIRTIO 和 VHOST_NET
- 设备分配和 SR-IOV
- 桥接零复制传输
- 多队列 virtio-net

常规建议

- 使用多个网络以避免单一网络过载
- 调整MTU，通过提高最大传输单元值可以减少碎片。
- 使用 arp_filter 阻止 ARP Flux，
  - 运行 echo 1 > /proc/sys/net/ipv4/conf/all/arp_filter
  - 或编辑 /etc/sysctl.conf 让重启后这设置得以持续

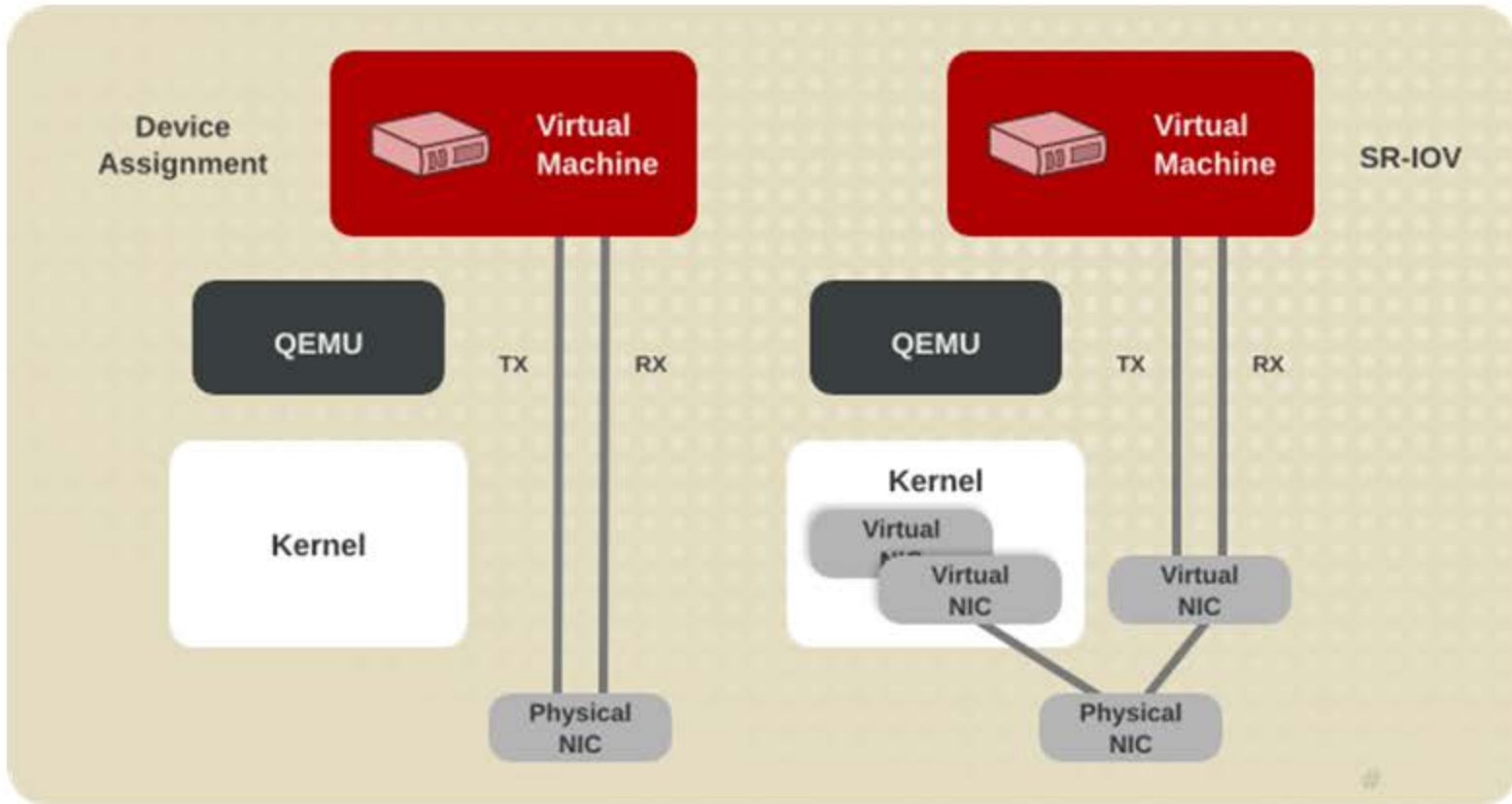
VIRTIO 和 VHOST_NET

- vhost_net 将部分 Virtio 驱动从用户空间移至 kernel
- 这将减少复制操作、降低延迟和 CPU 占用量



设备分配和 SR-IOV

- 设备分配使整个设备在客户机中可用
- SR-IOV 需要驱动和硬件中的支持，实现最低延迟。



桥接零复制传输　　Bridge Zero Copy Transmit

- 对于大尺寸的数据包较为有效
  - 客户机网络和外部网络间的大数据包传输中，它对主机 CPU 负荷的减少可达到 15%，对吞吐量没有影响
  - 它不对虚拟机到虚拟机、虚拟机到主机或小数据包负载造成影响
- RHEL/CentOS 7 虚拟机完全支持桥接零复制传输，但是被默认禁用
- 启动零复制传输模式

```
创建新文件vhost-net.conf
# vi /etc/modprobe.d/vhost-net.conf 添加如下内容:
options vhost_net  experimental_zcopytx=1

如果想再次禁用
# modprobe -r vhost_net
# modprobe vhost_net experimental_zcopytx=0
```

多队列 virtio-net

- 允许每一次通过一组以上的 virtqueue 传输数据包
- 在以下场景中，可以提供最佳性能：
  - 流量数据包相对较大
  - 客户机同时在各种连接中活跃，流量从客户机、客户机到主机或客户端运行到外部系统。
  - 队列数量与虚拟 CPU 相同
- 配置方法：修改虚拟机配置文件

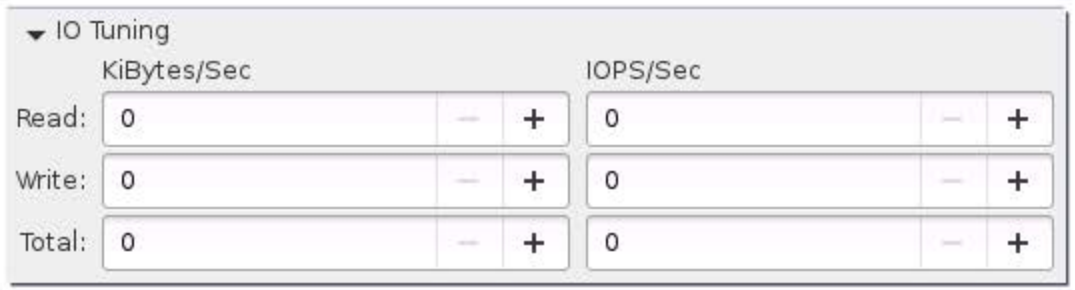
```
<interface type='network'>
      <source network='default'/>
      <model type='virtio'/>
      <driver name='vhost' queues='N'/>
</interface>
```

存储性能优化

- 磁盘 I/O 节流　　Disk I/O Throttling
- 缓存
- I/O 模式
- 块 I/O 调整　　BLOCK I/O TUNING
- 多队列 virtio-scsi

磁盘 I/O 节流　　Disk I/O Throttling

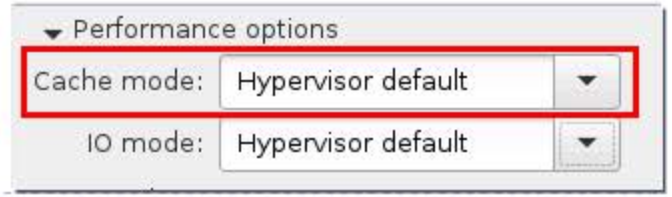
- 避免虚拟机过度使用共享资源，并防止影响其他虚拟机的性能
- 可通过virt-manager或virsh来进行配置



例如，将虚拟机中的 vda 节流至 I/O 每秒 1000、吞吐量为每秒 50 MB：

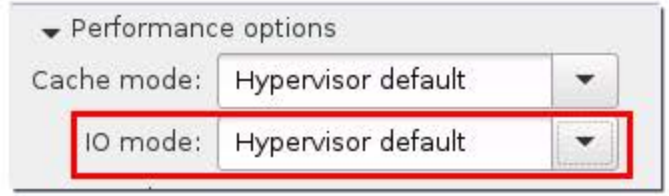
```
# virsh blkdeviotune testvm vda --total-iops-sec 1000 \
  --total-bytes-sec 52428800
```

缓存



缓存选项	描述
none	虚拟机中的 I/O 不能在主机上缓存，但可以保留在回写磁盘缓存中。在虚拟机中使用此选项来应对大多数的需求。此选项通常是支持迁移的最佳和唯一选项。
writethrough	虚拟机中的 I/O 在主机上缓存，但在物理媒介中直写。该模式较慢且更易造成缩放问题。最好是在虚拟机数量较小且 I/O 需求较低的情况下使用。推荐的应用对象是无需迁移、不支持回写缓存的虚拟机（如 Red Hat Enterprise Linux 5.5 或更早的版本）。
writeback	虚拟机中的 I/O 在主机上缓存。
directsync	与 writethrough 相似，但虚拟机中的 I/O 将忽略主机页面缓存。
unsafe	主机可能会缓存所有的 I/O 磁盘，虚拟机的同步要求将被忽略。
default	如果没有指定缓存模式，将会选择系统默认设置。

I/O 模式



IO 模式选项	描述
native	RHEL环境的默认值。该模式适用于直接 I/O 选项的 kernel 非同步 I/O
threads	默认为基于主机用户模式的线程。
default	RHEL/CentOS 7 默认为线程模式。

虚拟机的XML配置

```
<disk type='file' device='disk'>
          <driver name='qemu' type='raw' io='threads'/>
```

块 I/O 调整　BLOCK I/O TUNING

```
# virsh blkiotune virtual_machine [--weight number] \
  [--device-weights string] [--config] [--live] [--current]
```

- weight
  - I/O 的权重范围在 100 到 1,000 之间
- device-weights
  - 如何有多个设备，可以实现更细粒度的控制
- config
  - 使更改在下次启动时生效。
- Live
  - 在运行的虚拟机中应用这些更改。
- Current
  - 在当前的虚拟机中应用这些更改。

多队列 virtio-scsi

- 提供了改进的存储性能和 virtio-scsi 驱动中的可扩展性
- 使每个虚拟 CPU 可以使用独立队列和中断，从而不会影响到其他虚拟 CPU
- RHEL/CentOS 7 中默认禁用多队列 virtio-scsi
- 启用方法：
  - 在客机 XML 配置中添加以下命令，其中 N 为虚拟 CPU 队列的总数

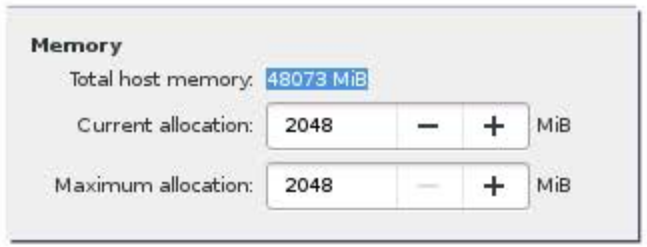
```
<controller type='scsi' index='0' model='virtio-scsi'>
   <driver queues='N' />
</controller>
```

内存性能优化

- 常规建议
- 内存监控工具
- 使用 virsh 调整内存配置
- 大页和透明大页

常规建议

- 请勿为虚拟机分配超过所需的其它资源




```
<memory unit='KiB'>2097152</memory>
```

- 在可能的情况下，如果 NUMA 节点中有足够的资源，将一台客机分配到一个单一 NUMA 节点

内存监控工具

- 在物理机环境中使用的监控内存的工具也可以在虚拟机中使用。
- 以下工具可以被用来监控内存的使用、诊断与内存相关的问题
  - `top`
  - `vmstat`
  - `numastat`
  - `/proc/`目录下的文件

使用 virsh 调整内存配置

- XML 中可选的 <memtune> 元素用于配置客户虚拟机内存设置
- 如果省略 <memtune>，内存设置将被默认应用。
- virsh memtune 命令显示或设置虚拟机

```
# virsh memtune virtual_machine --parameter size
```

- hard_limit　　虚拟机可以使用的最大内存，
- soft_limit　　发生内存争用时，内存限制使用
- swap_hard_limit　　虚拟机加上交换(swap)可使用的最大内存
- min_guarantee　　保证分配给虚拟机可以使用的最小内存

以上参数的单位均是千字节

大页和透明大页

- Huge Pages and Transparent Huge Pages
- x86 CPU 通常会在 4kB 页中处理内存，但可以使用更大的 2MB 或 1GB 页，即 huge page（大页）。
- 大页内存可以支持 KVM 客户机部署，以改善性能。

示例：为虚拟机启用 1GB 大页

- 在启动时分配 4 个 1GB 的大页面和 1,024 个 2MB 的大页面

```
'default_hugepagesz=1G hugepagesz=1G hugepages=4 hugepagesz=2M hugepages=1024'

# echo 4 > /sys/devices/system/node/node1/hugepages/hugepages-1048576kB/nr_hugepages
# echo 1024 > /sys/devices/system/node/node3/hugepages/hugepages-2048kB/nr_hugepages
```

- 将 2MB 和 1GB 的大页面挂载到主机

```
# mkdir /dev/hugepages1G
# mount -t hugetlbfs -o pagesize=1G none /dev/hugepages1G
# mkdir /dev/hugepages2M
# mount -t hugetlbfs -o pagesize=2M none /dev/hugepages2M
```

- 重启 libvirtd，使 1GB 大页面可以在客户机上启用：

```
# systemctl restart libvirtd
```

CPU性能优化

- 在非统一内存访问（NUMA，Non-Uniform Memory Access）中，系统内存被划分到 NUMA 节点（node），并且与 socket 相对应，或与特定某一组与本地系统内存子集具有相同访问延迟的 CPU 相对应。
- RHEL/CentOS 7中
  - 默认启用了NUMA
  - 自动化 NUMA 平衡改进了 NUMA 硬件系统中运行应用的性能。

总结

- 性能监视与优化方法论
- 监视工具
- RHEL/CentOS 7 KVM性能特性和改进
- virt-manager中的性能管理
- 网络性能优化
- 存储性能优化
- 内存性能优化
- CPU性能优化