

使用 KVM 部署 ASA Virtual

您可以在能够运行基于内核的虚拟机 (KVM) 的任何服务器类 x86 CPU 设备上部署 ASA virtual。

重要事项 ASA virtual的最低内存要求为 2GB。如果当前 ASA virtual的内存少于 2GB，您将无法在不增加 ASA virtual机内存的情况下，从早期版本升级到 9.13(1) 及更高版本。您也可以使用最新版本重新部署新的 ASA virtual机。

- 准则和限制，第 1 页
- 概述，第 4 页
- 前提条件，第 4 页
- 准备 Day 0 配置文件，第 5 页
- 准备虚拟网桥 XML 文件，第 7 页
- 部署 ASA Virtual，第 9 页
- 性能调优，第 10 页
- CPU 使用情况和报告，第 20 页

准则和限制

根据所需部署的实例数量和使用要求，ASA virtual部署所使用的具体硬件可能会有所不同。创建的每台虚拟设备都需要主机满足最低资源配置要求，包括内存、CPU 数量和磁盘空间。

重要事项 ASA virtual部署时的磁盘存储大小为 8GB。无法更改磁盘空间的资源配置。

注释 从 ASA Virtual 版本 9.16.x 开始，当您从设备配置为 16 vCPU 和 32GB RAM 的 ASAv100 降级到 ASAv10 时，您必须为设备配置 1 vCPU 和 4GB RAM。

在部署 ASA virtual之前，请查看以下准则和限制。

KVM 上的 ASA Virtual系统要求

请确保遵循以下规范，以确保最佳性能。ASA virtual具有以下要求：

- 主机 CPU 必须是包含虚拟化扩展的基于 x86 的服务器类 Intel 或 AMD CPU。

  例如，ASA virtual性能测试实验室最少使用以下设备：使用以 2.6GHz 运行的 Intel® Xeon® CPU E5-2690v4 处理器的 Cisco Unified Computing System™ (Cisco UCS®) C 系列 M4 服务器。

建议的 vNIC

推荐使用以下 vNIC 以获得最佳性能。

- PCI 直通中的 i40e - 将服务器的物理 NIC 指定给 VM，并通过 DMA（直接内存访问）在 NIC 与 VM 之间传输数据包数据。移动数据包不需要任何 CPU 周期。
- i40evf/ixgbe-vf - 基本同上（在 NIC 与 VM 之间传输 DMA 数据包），但允许在多个 VM 之间共享 NIC。SR-IOV 通常是首选的，因为它具有更多部署灵活性。请参阅
- virtio - 这是并行虚拟化的网络驱动程序，支持 10Gbps 操作，但也需要 CPU 周期。

注释 在 KVM 系统上运行的 ASA virtual 实例可能会在使用 vNIC 驱动程序 i40e 版本 2.17.4 的 SR-IOV 接口时遇到数据连接问题。我们建议您将此 vNIC 版本升级为其他版本，以便解决此问题。

性能优化

为实现 ASA virtual的最佳性能，您可以对 VM 和主机进行调整。有关详细信息，请参阅 性能调优，第 10 页。

- **NUMA** - 您可以通过将来宾 VM 的 CPU 资源隔离到单一非一致内存访问 (NUMA) 节点来提高 ASA virtual的性能。有关详细信息，请参阅 NUMA 准则，第 11 页。
- **接收端扩展** - ASA virtual 支持接收端扩展 (RSS)，网络适配器利用这项技术将网络接收流量分发给多个处理器内核。有关详细信息，请参阅用于接收端扩展 (RSS) 的多个 RX 队列，第 13 页。
- **VPN 优化 (VPN Optimization)** - 有关使用 ASA virtual优化 VPN 性能的其他注意事项，请参阅 VPN 优化，第 15 页。

集群

从版本 9.17 开始，KVM 上部署的 ASA 虚拟实例支持集群。有关详细信息，请参阅 ASAv 的 ASA 集群。

CPU 固定

要让 ASA virtual在 KVM 环境中正常工作，需要 CPU 固定；请参阅启用 CPU 固定功能，第 10 页。

通过故障转移实现高可用性准则

对于故障转移部署，请确保备用设备具有相同的许可证权限；例如，两台设备均应具备 2Gbps 权限。

重要事项 使用 ASA virtual创建高可用性对时，需要按相同顺序将数据接口添加到每个 ASA virtual。如果完全相同的接口添加到每个 ASA virtual，但采用不同的顺序，在 ASA virtual控制台上会显示错误。故障转移功能可能也会受到影响。

Proxmox VE 上的 ASA Virtual

Proxmox 虚拟环境 (VE) 是可以管理 KVM 虚拟机的开源服务器虚拟化平台。Proxmox VE 还提供基于 Web 的管理界面。

在 Proxmox VE 上部署 ASA virtual时，需要配置 VM 以拥有模拟串行端口。如果没有串行端口，ASA virtual会在启动过程中进入环路。所有管理任务均可使用 Proxmox VE 基于 Web 的管理界面来完成。

注释 对于习惯使用 Unix shell 或 Windows Powershell 的高级用户，Proxmox VE 提供了一个命令行界面来管理虚拟环境的所有组件。此命令行界面具有智能制表符补全和 UNIX 手册页形式的完整文档。

要让 ASA virtual正常启动，虚拟机需要配置串行设备：

1. 在主管理中心中，在左侧导航树中选择 ASA virtual机。
2. 断开虚拟机电源。
3. 依次选择**硬件 (Hardware)** > **添加 (Add)** > **网络设备 (Network Device)**并添加串行端口。
4. 接通虚拟机电源。
5. 使用 Xterm.js 访问 ASA virtual机。

有关如何在访客/服务器上设置和激活终端的信息，请参阅 Proxmox 串行终端 (Serial Terminal) 页面。

IPv6 支持

要在 KVM 上创建具有 IPv6 支持配置的 vNIC，您必须为每个包含 IPv6 配置参数的接口创建一个 XML 文件。您可以使用命令 **virsh net-create** <<*interface configuration XML file name*>> 来安装具有 IPV6 网络协议配置的 vNIC。

对于每个接口，您可以创建以下 XML 文件：

- 管理接口 - *mgmt-vnic.xml*
- 诊断接口 - *diag-vnic.xml*
- 内部接口 - *inside-vnic.xml*
- 外部接口 - *outside-vnic.xml*

示例：

使用 IPv6 配置为管理接口创建 XML 文件。

```
<network>
            <name>mgmt-vnic</name>
            <bridge name='mgmt-vnic' stp='on' delay='0' />
            <ip family='ipv6' address='2001:db8::a111:b220:0:abcd' prefix='96'/>
        </network>
```

同样，您也必须为其他接口创建 XML 文件。

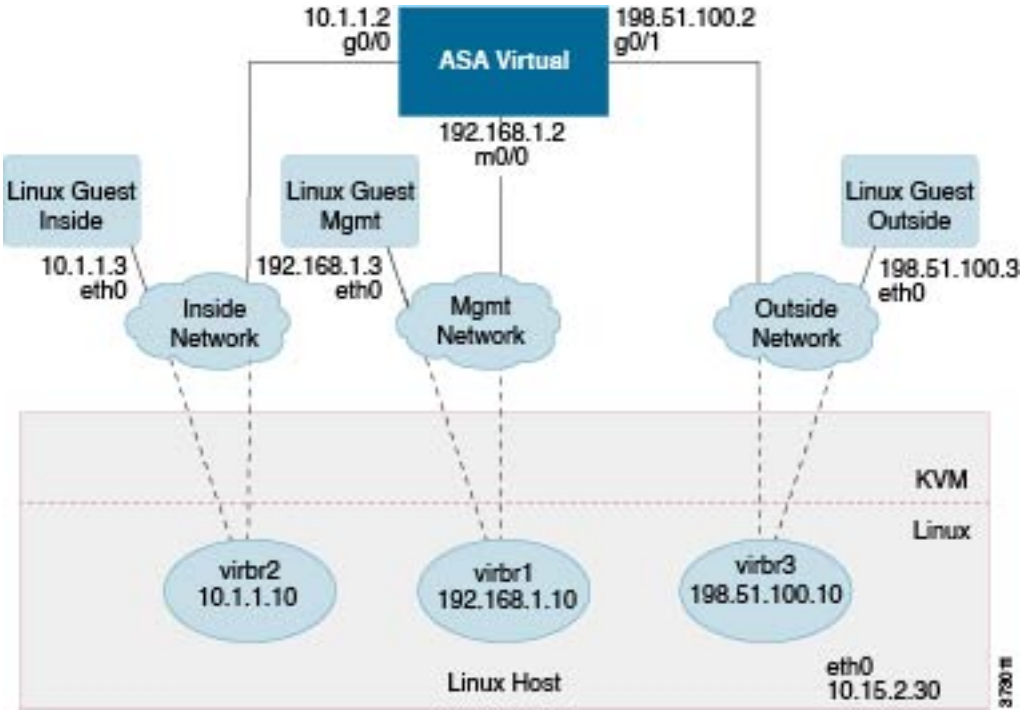
您可以通过运行以下命令来验证 KVM 上安装的虚拟网络适配器。

```
virsh net-list
         brctl show
```

概述

下图显示了使用 ASA virtual和 KVM 的网络拓扑示例。本章所述的程序均基于此拓扑示例。ASA virtual 用作内部和外部网络之间的防火墙。另外，此示例中还配置了一个单独的管理网络。

图 *1:* 使用 *KVM* 的 *ASA Virtual*部署示例



前提条件

- 从 Cisco.com 下载 ASA virtual qcow2 文件并将其放在 Linux 主机上：

  http://www.cisco.com/go/asa-software

注释 需要 Cisco.com 登录信息和思科服务合同。

- 本文档出于示例部署目的，假设您使用 Ubuntu 18.04 LTS。在 Ubuntu 18.04 LTS 主机之上安装以下软件包：
  - qemu-kvm
  - libvirt-bin
  - bridge-utils
  - virt-manager
  - virtinst
  - virsh tools
  - genisoimage
- 性能受主机及其配置的影响。通过调整主机，您可以最大化 KVM 上的 ASA virtual吞吐量。有关一般的主机调整概念，请参阅 NFV 与 Intel 携手实现高数据包处理性能。
- Ubuntu 18.04 的有用优化包括以下各项：
  - macvtap - 高性能 Linux 网桥；您可以使用 macvtap，而不是 Linux 网桥。注意，您必须配置特定设置才能使用 macvtap，而不是 Linux 网桥。
  - 透明大页 - 增加内存页面大小，在 Ubuntu 18.04 中默认开启。

    禁用超线程 - 用于将两个 vCPU 减少到一个单核。
  - txqueuelength - 用于将默认 txqueuelength 增加到 4000 个数据包并减少丢包率。
  - 固定 - 用于将 qemu 和 vhost 进程固定到特定 CPU 内核；在某些情况下，固定可显著提高性能。
- 有关优化基于 RHEL 的分布的信息，请参阅《Red Hat Enterprise Linux 7 虚拟化调整和优化指南》。
- 对于 ASA 软件和 ASA virtual 虚拟机监控程序兼容性，请参阅 CISCO Cisco Secure Firewall ASA 兼容性。

准备 Day 0 配置文件

在启动 ASA virtual之前，您可以准备一个 Day 0 配置文件。此文件是包含将在 ASA virtual 启动时应用的 ASA virtual 配置的文本文件。此初始配置将放入您选择的工作目录中名为“day0-config”的文本文件，并写入首次启动时安装和读取的 day0.iso 文件。Day 0 配置文件必须至少包含用于激活管理接口以及设置用于公共密钥身份验证的 SSH 服务器的命令，但它还可包含完整的 ASA 配置。

day0.iso 文件（自定义 day0.iso 或默认 day0.iso）必须在首次启动过程中可用：

- 要在初始部署过程中自动完成 ASA virtual 的许可过程，请将从思科智能软件管理器下载的智能许可身份 (ID) 令牌放入与 Day 0 配置文件处于同一目录且名为“idtoken”的文本文件。
- 如果需要从虚拟机监控程序的**串行端口**（而不是虚拟 VGA 控制台）访问和配置 ASA virtual，则 Day 0 配置文件中应包括 console serial 设置，才能在首次启动过程中使用串行端口。
- 如果要在透明模式下部署 ASA virtual，则必须在透明模式下将已知的运行 ASA 配置文件用作 Day 0 配置文件。这不适用于路由防火墙的 Day 0 配置文件。

注释 我们在本示例中使用的是 Linux，但对于 Windows 也有类似的实用程序。

过程

步骤 1 在名为“day0-config”的文本文件中输入 ASA virtual的 CLI 配置。添加三个接口的接口配置和所需的任何其他配置。

第一行应以 ASA 版本开头。day0-config 应该是有效的 ASA 配置。生成 day0-config 的最佳方式是从现有的 ASA 或 ASA virtual复制一个运行配置的相关部分。day0-config 中的行顺序很重要，应与现有的 **show running-config** 命令输出中看到的顺序相符。

示例：

```
ASA Version
!
interface management0/0
ipv6 enable
ipv6 address 2001:db8::a111:b220:0:abcd/96
nameif management
security-level 100
no shut

interface gigabitethernet0/0
ipv6 enable
ipv6 address 2001:db8::a111:b221:0:abcd/96
nameif inside
security-level 100
no shut

interface gigabitethernet1/0
ipv6 enable
ipv6 address 2001:db8::a111:b222:0:abcd/96
nameif outside
security-level 100
no shut

crypto key generate rsa general-keys modulus 4096
ssh ::/0 inside
ssh timeout 60
ssh version 2
aaa authentication ssh console LOCAL

dns domain-lookup management
```

```
dns server-group DefaultDNS
name-server 2001:4860:4860::8888
```

步骤 2 （可选）若要在初始 ASA virtual 部署过程中进行自动许可，请确保 day0-config 文件中包含以下信息：

- 管理接口 IP 地址
- （可选）要用于智能许可的 HTTP 代理
- 用于启用与 HTTP 代理（如果指定）或 tools.cisco.com 的连接的 **route** 命令
- 将 tools.cisco.com 解析为 IP 地址的 DNS 服务器
- 指定您正请求的 ASA virtual 许可证的智能许可配置
- （可选）更加便于 ASA virtual 在 CSSM 中进行查找的唯一主机名

步骤 3 （可选）将 Cisco Smart Software Manager 颁发的智能许可证身份令牌文件下载到您的计算机，从下载文件中复制 ID 令牌，然后将其置于名为“idtoken”的文本文件中，该文件只包含 ID 令牌。

步骤 4 通过将文本文件转换成 ISO 文件生成虚拟 CD-ROM：

示例：

```
stack@user-ubuntu:-/KvmAsa$ sudo genisoimage -r -o day0.iso day0-config idtoken
I: input-charset not specified, using utf-8 (detected in locale settings)
Total translation table size: 0
Total rockridge attributes bytes: 252
Total directory bytes: 0
Path table size (bytes): 10
Max brk space used 0
176 extents written (0 MB)
stack@user-ubuntu:-/KvmAsa$
```

身份令牌自动向智能许可服务器注册 ASA virtual。

步骤 5 重复步骤 1 到 5，使用相应的 IP 地址为要部署的每个 ASA virtual 创建单独的默认配置文件。

准备虚拟网桥 XML 文件

您需要设置将 ASA virtual 访客连接到 KVM 主机，以及将访客彼此连接的虚拟网络。

注释 此程序不会建立与 KVM 主机之外的外部环境的连接。

在 KVM 主机上准备虚拟网桥 XML 文件。对于准备 Day 0 配置文件，第 5 页所述的虚拟网络拓扑示例，您需要以下三个虚拟网桥文件：virbr1.xml、virbr2.xml 和 virbr3.xml（您必须使用这三个文件名；例如，不允许使用 virbr0，因为它已经存在）。每个文件具有设置虚拟网桥所需的信息。您必须为虚拟网桥提供名称和唯一的 MAC 地址。提供 IP 地址是可选的。

过程

步骤 1 创建三个虚拟网络网桥 XML 文件。例如，virbr1.xml、virbr2.xml 和 virbr3.xml：

示例：

```
<network>
<name>virbr1</name>
<bridge name='virbr1' stp='on' delay='0' />
<mac address='52:54:00:05:6e:00' />
<ip address='192.168.1.10' netmask='255.255.255.0' />
</network>
```

示例：

```
<network>
<name>virbr2</name>
<bridge name='virbr2' stp='on' delay='0' />
<mac address='52:54:00:05:6e:01' />
<ip address='10.1.1.10' netmask='255.255.255.0' />
</network>
```

示例：

```
<network>
<name>virbr3</name>
<bridge name='virbr3' stp='on' delay='0' />
<mac address='52:54:00:05:6e:02' />
<ip address='198.51.100.10' netmask='255.255.255.0' />
</network>
```

步骤 2 创建包含以下内容的脚本（在本例中，我们将脚本命名为 virt_network_setup.sh）：

```
virsh net-create virbr1.xml
virsh net-create virbr2.xml
virsh net-create virbr3.xml
```

步骤 3 运行此脚本以设置虚拟网络。此脚本将生成虚拟网络。只要 KVM 主机运行，网络就会保持运行。

```
stack@user-ubuntu:-/KvmAsa$ virt_network_setup.sh
```

注释
如果重新加载 Linux 主机，则必须重新运行 virt_network_setup.sh 脚本。此脚本在主机重启期间即停止运行。

步骤 4 验证虚拟网络是否已创建：

```
stack@user-ubuntu:-/KvmAsa$ brctl show
bridge name bridge id STP enabled Interfaces
virbr0 8000.0000000000000 yes
virbr1 8000.5254000056eed yes virb1-nic
virbr2 8000.5254000056eee yes virb2-nic
virbr3 8000.5254000056eec yes virb3-nic
stack@user-ubuntu:-/KvmAsa$
```

步骤 5 显示分配给 virbr1 网桥的 IP 地址。这是您在 XML 文件中分配的 IP 地址。

```
stack@user-ubuntu:-/KvmAsa$ ip address show virbr1
S: virbr1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN
link/ether 52:54:00:05:6e:00 brd ff:ff:ff:ff:ff:ff
inet 192.168.1.10/24 brd 192.168.1.255 scope global virbr1
valid_lft forever preferred_lft forever
```

部署 ASA Virtual

使用基于 virt-install 的部署脚本启动 ASA virtual。

过程

步骤 1 创建名为“virt_install_asav.sh”的 virt-install 脚本。

ASA virtual机的名称在此 KVM 主机上的所有其他 VM 中必须是唯一的。

ASA virtual最多可以支持 10 个网络。此示例使用三个网络。网络网桥语句的顺序非常重要。第一个列出的始终是 ASA virtual的管理接口 (Management 0/0)，第二个列出的是 ASA virtual的 GigabitEthernet 0/0，第三个列出的是 ASA virtual的 GigabitEthernet 0/1，以此类推，直至 GigabitEthernet 0/8。虚拟 NIC 必须是 Virtio。

示例：

```
virt-install \
--connect=qemu:///system \
--network network=default,model=virtio \
--network network=default,model=virtio \
--network network=default,model=virtio \
--name=asav \
--cpu host \
--arch=x86_64 \
--machine=pc-1.0 \
--vcpus=1 \
--ram=2048 \
--os-type=linux \
--virt-type=kvm \
--import \
--disk path=/home/kvmperf/Images/desmo.qcow2,format=qcow2,device=disk,bus=virtio,cache=none \
--disk path=/home/kvmperf/asav_day0.iso,format=iso,device=cdrom \
--console pty,target_type=virtio \
--serial tcp,host=127.0.0.1:4554,mode=bind,protocol=telnet
```

步骤 2 运行 virt_install 脚本：

示例：

```
stack@user-ubuntu:-/KvmAsa$ ./virt_install_asav.sh

Starting install...
Creating domain...
```

此时将出现一个窗口，其中显示虚拟机的控制台。您可以看到虚拟机正在启动。启动虚拟机需要几分钟时间。在虚拟机停止启动后，您可以从控制台屏幕发出 CLI 命令。

性能调优

提高 KVM 配置的性能

在 KVM 环境中，通过更改 KVM 主机上的设置，可以提高 ASA virtual 的性能。这些设置与主机服务器上的配置设置无关。此选项适用于 Red Hat Enterprise Linux 7.0 KVM。

通过启用 CPU 固定，可以提高 KVM 配置的性能。

启用 CPU 固定功能

ASA virtual要求您使用KVM CPU 关联选项提高KVM 环境中 ASA virtual的性能。处理器关联或 CPU 固定可实现一个进程或线程与一个中央处理单元 (CPU) 或一系列 CPU 的绑定和取消绑定，以便该进程或线程仅在指定的一个或多个 CPU（而非任何 CPU）上执行。

配置主机聚合，将使用 CPU 固定的实例与不使用 CPU 固定的实例部署在不同主机上，以避免未固定实例使用已固定实例的资源要求。

注意 不要在相同主机上部署有 NUMA 拓扑的实例和没有 NUMA 拓扑的实例。

要使用此选项，请在 KVM 主机上配置 CPU 固定功能。

过程

步骤 1 在 KVM 主机环境中，验证主机拓扑以查明可用于固定的 vCPU 数量：

示例：

```
virsh nodeinfo
```

步骤 2 验证可用的 vCPU 数量：

示例：

```
virsh capabilities
```

步骤 3 将 vCPU 固定到处理器内核组：

示例：

```
virsh vcpupin <vm-name> <vcpu-number> <host-core-number>
```

对于 ASA virtual 上的每个 vCPU，都必须执行 **virsh vcpupin** 命令。以下示例显示当您的 ASA virtual 配置包含四个 vCPU 且主机包含八个内核时所需的 KVM 命令:

```
virsh vcpupin asav 0 2
virsh vcpupin asav 1 3
virsh vcpupin asav 2 4
virsh vcpupin asav 3 5
```

主机内核编号可以是 0 到 7 之间的任意数字。有关详细信息，请参阅 KVM 文档。

注释
在配置 CPU 固定功能时，请认真考虑主机服务器的 CPU 拓扑。如果使用配置了多个内核的服务器，请不要跨多个插槽配置 CPU 固定。

提高 KVM 配置性能的负面影响是，它需要专用的系统资源。

NUMA 准则

非一致内存访问 (NUMA) 是一种共享内存架构，描述了多处理器系统中主内存模块相对于处理器的位置。如果处理器访问的内存不在自己的节点内（远程内存），则数据通过NUMA连接以低于本地内存的访问速率传输。

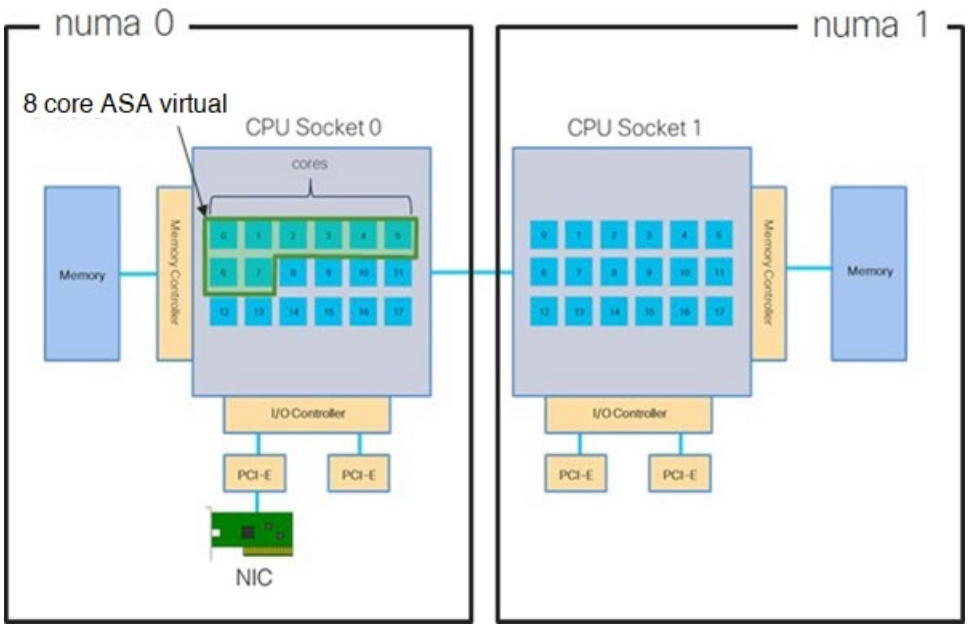
X86 服务器架构由多个插槽和每个插槽内的多个内核组成。每个 CPU 插槽及其内存和 I/O 均称为 NUMA 节点。要从内存高效读取数据包，来宾应用和关联的外围设备（例如 NIC）应位于同一个节点中。

为获得最佳 ASA virtual性能：

- ASA virtual VM 必须在单一 NUMA 节点上运行。如果部署了单个 ASA virtual以跨 2 个插槽运行，则性能将显著下降。
- 8 核 ASA virtual (图 2: 8 核 ASA Virtual NUMA 架构示例， 第 12 页) 要求主机 CPU 上的每个插槽至少有 8 个内核。必须考虑服务器上运行的其他虚拟机。
- 16 核 ASA virtual (图 3: 16 核 ASA Virtual NUMA 架构示例， 第 12 页) 要求主机 CPU 上的每个插槽至少有 16 个内核。必须考虑服务器上运行的其他虚拟机。
- NIC 应与 ASA virtual机位于同一 NUMA 节点上。

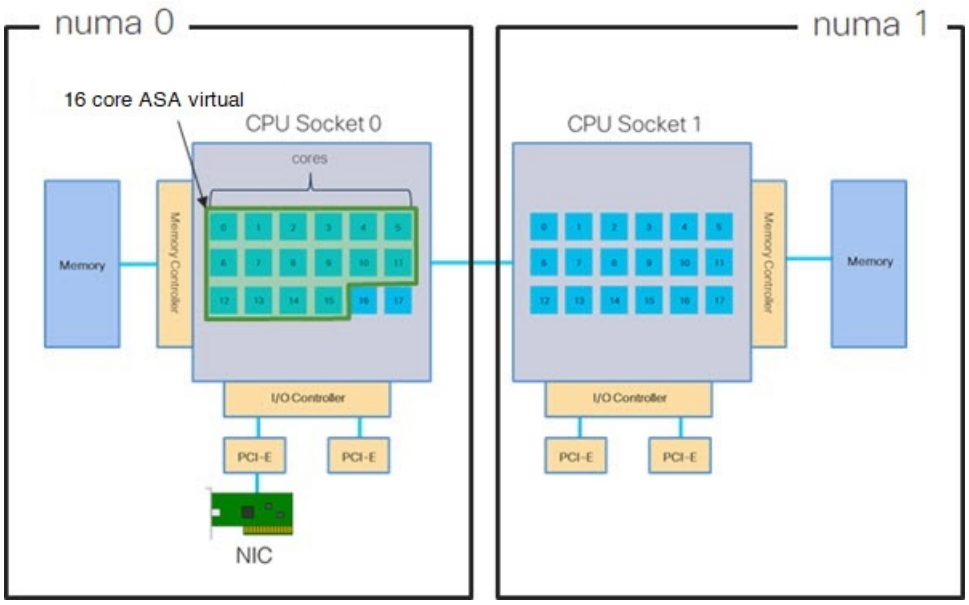
下图显示的服务器有两个 CPU 插槽，每个 CPU 有 18 个内核。8 核 ASA virtual要求主机 CPU 上的每个插槽至少有 8 个内核。

图 *2: 8* 核 ***ASA Virtual NUMA*** 架构示例



下图显示的服务器有两个 CPU 插槽，每个 CPU 有 18 个内核。16 核 ASA virtual要求主机 CPU 上的每个插槽至少有 16 个内核。

图 *3: 16* 核 ***ASA Virtual NUMA*** 架构示例



NUMA 优化

最佳情况下，ASA virtual机应在运行 NIC 的同一 NUMA 节点上运行。为此：

1. 使用“lstopo”显示节点图，确定 NIC 所在的节点。找到 NIC 并记录它们连接的节点。
2. 在 KVM 主机上，使用 `virsh list` 查找 ASA virtual。
3. 编辑 VM：`virsh edit <VM Number>`。
4. 对齐所选节点上的 ASA virtual。以下示例以 18 核节点为前提。

   对齐节点 0：

   ```
   <vcpu placement='static' cpuset='0-17'>16</vcpu>
   <numatune>
       <memory mode='strict' nodeset='0'/>
   </numatune>
   ```

   对齐节点 1：

   ```
   <vcpu placement='static' cpuset='18-35'>16</vcpu>
   <numatune>
       <memory mode='strict' nodeset='1'/>
   </numatune>
   ```

5. 保存 .xml 更改并重启 ASA virtual机。
6. 为确保您的 VM 在所需的节点上运行，请执行 `ps aux | grep <name of your ASAv VM>` 以获取进程 ID。
7. 运行 `sudo numastat -c <ASAv VM Process ID>` 以查看 ASA virtual机是否正确对齐。

有关在 KVM 上使用 NUMA 调整的详细信息，请参阅 RedHat 文档 9.3. libvirt NUMA Tuning。

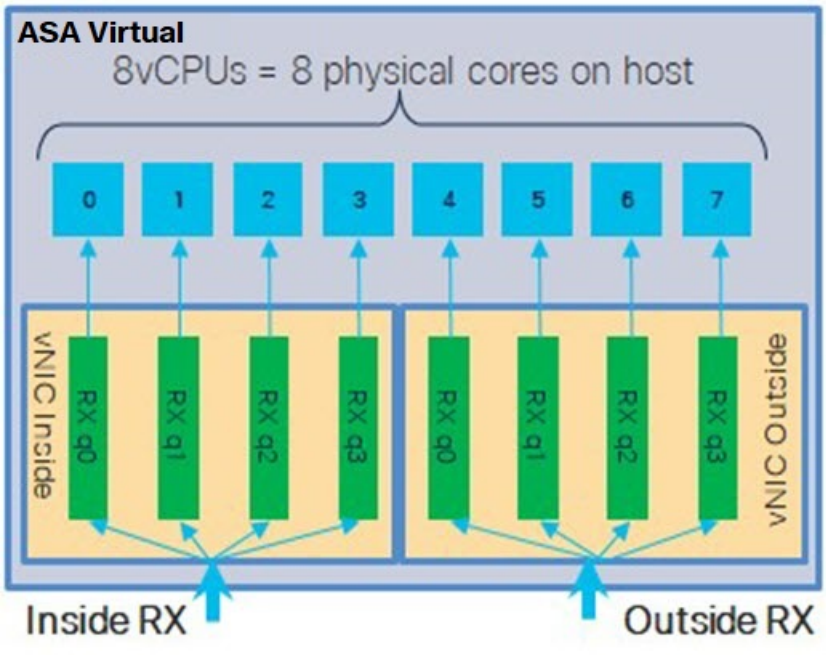
用于接收端扩展 (RSS) 的多个 RX 队列

ASA virtual支持接收端扩展 (RSS)，网络适配器利用这项技术将网络接收流量并行分发给多个处理器内核。为实现最大吞吐量，每个 vCPU（内核）都必须有自己的 NIC RX 队列。请注意，典型的 RA VPN 部署可能使用单一内部/外部接口对。

重要事项 您需要 ASA virtual版本 9.13(1) 或更高版本，才能使用多个 RX 队列。对于 KVM，*libvirt* 版本最低需要是 1.0.6。

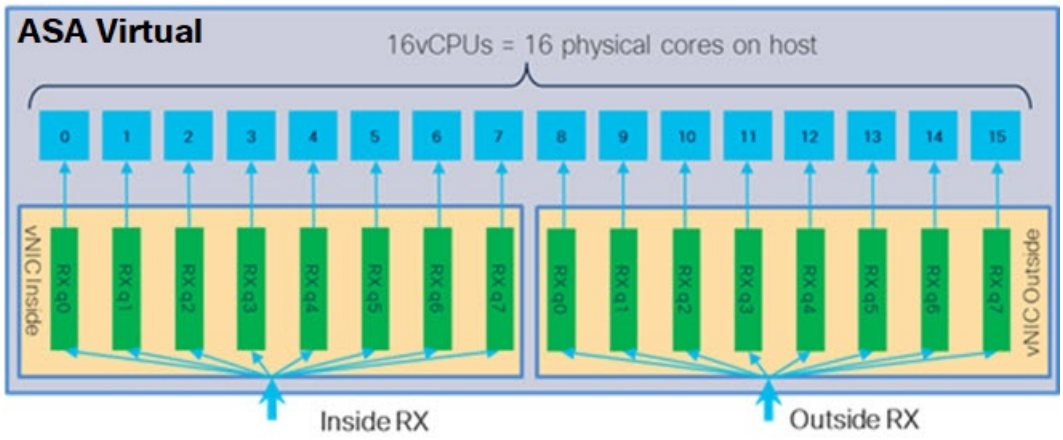
对于具有内部/外部接口对的 8 核 VM，每个接口将有 4 个 RX 队列，如图 4: 8 核 ASA Virtual RSS RX 队列 ，第 14 页中所示。

图 *4: 8* 核 *ASA Virtual RSS RX* 队列



对于具有内部/外部接口对的 16 核 VM，每个接口将有 8 个 RX 队列，如图 5: 16 核 ASA Virtual RSS RX 队列，第 14 页中所示。

图 *5: 16* 核 *ASA Virtual RSS RX* 队列



下表显示了 ASA virtual的适用于 KVM 的 vNIC 以及支持的 RX 队列数量。有关支持的 vNIC 的说明，请参阅建议的 vNIC，第 2 页。

表 *1: KVM* 建议的 *NIC/vNIC*

NIC 卡	vNIC 驱动程序	驱动程序技术	RX 队列数	性能
x710	i40e	PCI 直通	最多 8 个	X710 的 PCI 直通和 SR-IOV 模式性能最佳。SR-IOV 通常是虚拟部署的首选，因为 NIC 可在多个 VM 之间共享。
	i40evf	SR-IOV	8	
x520	ixgbe	PCI 直通	6	x520 NIC 性能比 x710 低 10% 到 30%。x520 的 PCI 直通和 SR-IOV 模式性能相似。SR-IOV 通常是虚拟部署的首选，因为 NIC 可在多个 VM 之间共享。
	ixgbe-vf	SR-IOV	2	
不适用	virtio	并行虚拟化	最多 8 个	不建议用于 ASAv100。 有关其他部署，请参阅为 Virtio on KVM 启用多队列支持，第 15 页。

为 Virtio on KVM 启用多队列支持

以下示例说明如何使用 virsh 编辑 libvirt xml，将 Virtio NIC RX 队列的数量配置为 4：

```
<interface type='bridge'>
 <mac address='52:54:00:43:6e:3f'/>
 <source bridge='clients'/>
 <model type='virtio'/>
 <driver name='vhost' queues='4'/>
 <address type='pci' domain='0x0000' bus='0x00' slot='0x04' function='0x0'/>
</interface>
```



重要事项 *libvirt* 版本最低需要 1.0.6 以支持多个 RX 队列。

VPN 优化

以下是使用 ASA virtual优化 VPN 性能的一些其他注意事项。

- IPSec 的吞吐量比 DTLS 更高。
- 密码 - GCM 的吞吐量大约为 CBC 的两倍。

SR-IOV 接口调配

SR-IOV 允许多个 VM 共享主机内的单一 PCIe 网络适配器。SR-IOV 定义了下列功能：

- 物理功能 (PF) - PF 指所有 PCIe 功能，包括 SR-IOV 功能。这些功能在主机服务器上显示为常规静态 NIC。
- 虚拟功能 (VF) - VF 是有助于数据传输的轻型 PCIe 功能。VF 源自于 PF，并通过 PF 进行管理。

VF 在虚拟化操作系统框架下，最高可以 10 Gbps 的速度连接 ASA virtual机。本节介绍如何在 KVM 环境下配置 VF。ASA Virtual和 SR-IOV 接口调配中介绍了 ASA virtual上对 SR-IOV 的支持信息。

在 ASAv5 和 ASAv10 上，强烈建议使用 VMXNET3 驱动程序以实现最佳性能。此外，SR-IOV 接口与 ASA virtual组合使用时（混合接口），尤其是在分配更多 CPU 核心和资源时。

SR-IOV 接口调配的要求

如果您有一个支持 SR-IOV 的物理 NIC，可以将支持 SR-IOV 的 VF 或虚拟 NIC (vNIC) 连接到 ASA virtual实例。此外，SR-IOV 还需要支持 BIOS 以及硬件上运行的操作系统实例或虚拟机监控程序。下面列出了对 KVM 环境中运行的 ASA virtual执行 SR-IOV 接口调配的一般准则：

- 在主机服务器中需要具有支持 SR-IOV 的物理 NIC；请参阅 SR-IOV 接口准则和限制。
- 您需要在主机服务器的 BIOS 中启用虚拟化。有关详细信息，请参阅供应商文档。
- 您需要在主机服务器的 BIOS 中启用 IOMMU 对 SR-IOV 的全局支持。有关详细信息，请参阅硬件供应商文档。
- KVM 上使用 SR-IOV 接口的 ASA virtual 支持混合接口类型。您可以将 SR-IOV 或 VMXNET3 用于管理接口，并将 SR-IOV 用于数据接口。

修改 KVM 主机 BIOS 和主机操作系统

本节介绍在KVM 系统上调配SR-IOV接口的各种安装和配置步骤。本节中的信息基于特定实验室环境中的设备创建，这些设备使用的是思科 UCS C 系列服务器上的 Ubuntu 14.04（配备有 Intel 以太网服务器适配器 X520 - DA2）。

开始之前

- 请确保已安装兼容 SR-IOV 的网络接口卡 (NIC)。
- 确保已启用 Intel 虚拟化技术 (VT-x) 和 VT-d 功能。



注释 有些系统制造商默认禁用这些扩展。我们建议您通过供应商文档验证该过程，因为不同的系统使用不同的方法来访问和更改 BIOS 设置。

- 确保在操作系统安装过程中已安装所有 Linux KVM 模块、库、用户工具和实用程序；请参阅前提条件，第 4 页。
- 确保物理接口处于“开启”状态。使用 ifconfig <*ethname*> 进行确认。

过程

步骤 1 使用“根”用户帐户和密码登录系统。

步骤 2 验证 Intel VT-d 是否已启用。

示例：

```
kvmuser@kvm-host:/$ dmesg | grep -e DMAR -e IOMMU
[    0.000000] ACPI: DMAR 0x000000006F9A4C68 000140 (v01 Cisco0 CiscoUCS 00000001 INTL 20091013)
[    0.000000] DMAR: IOMMU enabled
```

最后一行表示 VT-d 已启用。

步骤 3 通过将 *intel_iommu=on* 参数附加到 */etc/default/grub* 配置文件的 GRUB_CMDLINE_LINUX 条目，在内核中激活 Intel VT-d。

示例：

```
# vi /etc/default/grub
...
GRUB_CMDLINE_LINUX="nofb splash=quiet console=tty0 ... intel_iommu=on"
...
```

注释
如果您使用的是 AMD 处理器，则应改为将 *amd_iommu=on* 附加到引导参数。

步骤 4 重新启动服务器，以使 iommu 更改生效。

示例：

```
> shutdown -r now
```

步骤 5 创建 VF，具体方法为：通过 *sysfs* 接口向 *sriov_numvfs* 参数写入适当的值，格式如下：

```
#echo n > /sys/class/net/device name/device/sriov_numvfs
```

为了确保每次服务器通电时创建所需数量的 VF，请将上面的命令附加到 *rc.local* 文件中，该文件位于 */etc/rc.d/* 目录下。Linux 操作系统会在启动过程结束时执行 *rc.local* 脚本。

例如，下面显示了为每个端口创建一个 VF 的过程。适合您特定设置的接口不尽相同。

示例：

```
echo '1' > /sys/class/net/eth4/device/sriov_numvfs
echo '1' > /sys/class/net/eth5/device/sriov_numvfs
echo '1' > /sys/class/net/eth6/device/sriov_numvfs
echo '1' > /sys/class/net/eth7/device/sriov_numvfs
```

步骤 6 重新启动服务器。

示例：

```
> shutdown -r now
```

步骤 7 使用 *lspci* 确认是否已创建 VF。

示例：

```
 > lspci | grep -i "Virtual Function"
kvmuser@kvm-racetrack:~$ lspci | grep -i "Virtual Function"
```

```
0a:10.0 Ethernet controller: Intel Corporation 82599 Ethernet Controller Virtual Function (rev 01)
0a:10.1 Ethernet controller: Intel Corporation 82599 Ethernet Controller Virtual Function (rev 01)
0a:10.2 Ethernet controller: Intel Corporation 82599 Ethernet Controller Virtual Function (rev 01)
0a:10.3 Ethernet controller: Intel Corporation 82599 Ethernet Controller Virtual Function (rev 01)
```

注释
使用 **ifconfig** 命令，您会看到其他接口。

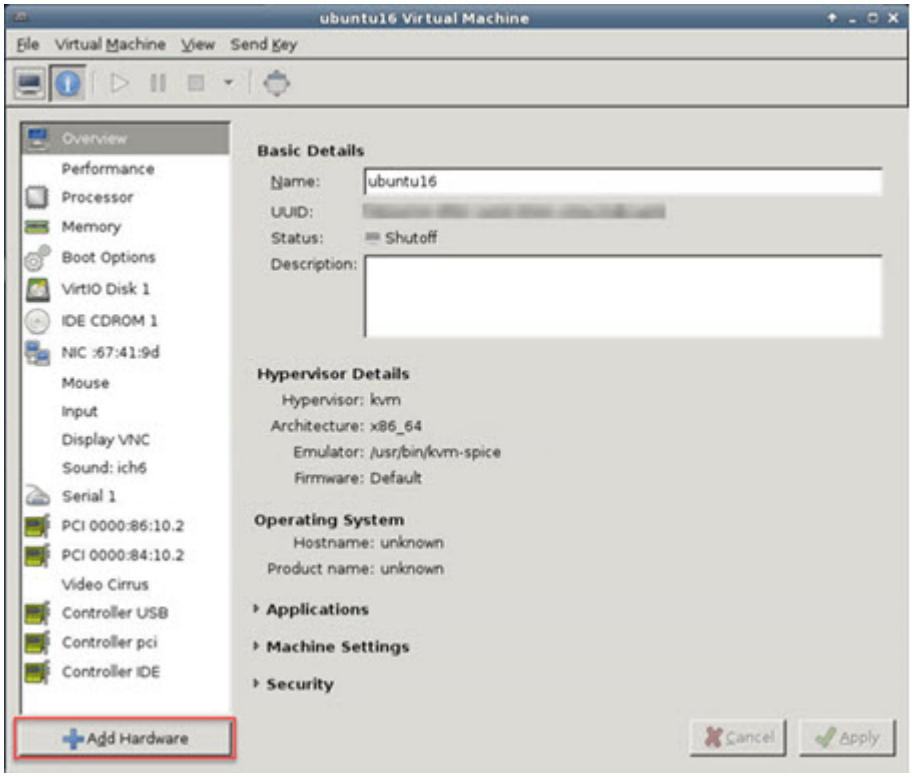
将 PCI 设备分配给 ASA Virtual

在创建 VF 后，您可以将它们添加到 ASA virtual中，就像添加任何 PCI 设备一样。以下示例说明如何使用图形 **virt-manager** 工具将以太网 VF 控制器添加到 ASA virtual。

过程

步骤 1 打开 ASA virtual，点击**添加硬件 (Add Hardware)** 按钮以将新设备添加到虚拟机中。

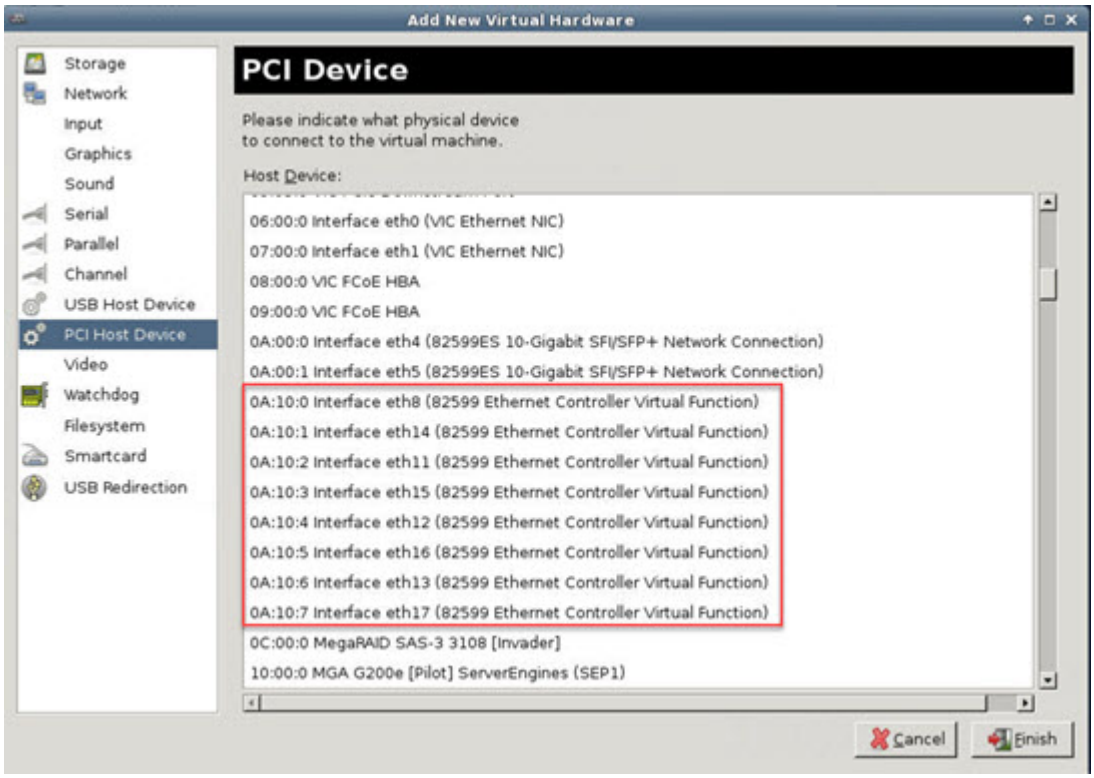
图 *6:* 添加硬件



步骤 2 点击左窗格**硬件 (Hardware)** 列表中的 **PCI 主机设备 (PCI Host Device)**。

PCI 设备列表（包括 VF）将出现在中心窗格中。

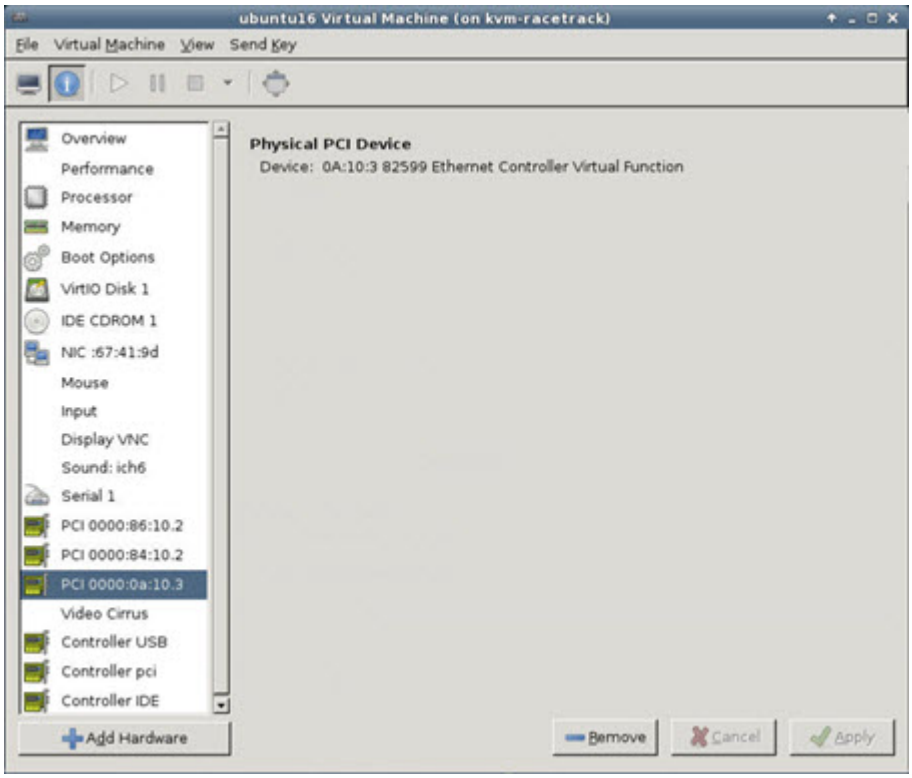
图 *7:* 虚拟功能列表



步骤 3 选择可用的虚拟功能之一，然后点击**完成 (Finish)**。

该 PCI 设备将出现在硬件列表中；请注意该设备被描述为以太网控制器虚拟功能。

图 *8:* 添加的虚拟功能



下一步做什么

- 使用 ASA virtual命令行中的 **show interface** 命令验证新配置的接口。
- 使用 ASA virtual 上的接口配置模式配置并启用该接口，以便传输和接收流量；有关详细信息，请参阅《思科 Cisco Secure Firewall ASA 系列常规操作 CLI 配置指南》的基本接口配置一章。

CPU 使用情况和报告

“CPU 利用率”(CPU Utilization) 报告汇总了指定时间内使用的 CPU 百分比。通常，核心在非高峰时段运行大约 30% 至 40% 的总 CPU 容量，在高峰时段运行大约 60% 至 70% 的容量。

重要事项 从 9.13(1) 开始，可以在任何支持的 ASA 虚拟 vCPU/内存配置上使用任何 ASA 虚拟许可证。这可让 ASA 虚拟客户在各种各样的 VM 资源中运行。

ASA 虚拟中的 vCPU 使用率

ASA 虚拟 vCPU 使用率显示了用于数据路径、控制点和外部进程的 vCPU 用量。

vSphere 报告的 vCPU 使用率包括上述 ASA 虚拟使用率，及：

- ASA 虚拟空闲时间
- 用于 ASA 虚拟机的 %SYS 开销
- 在 vSwitch、vNIC 和 pNIC 之间移动数据包的开销。此开销可能会非常大。

CPU 使用率示例

show cpu usage 命令可用于显示 CPU 利用率统计信息。

示例

```
Ciscoasa#show cpu usage

CPU utilization for 5 seconds = 1%; 1 minute: 2%; 5 minutes: 1%
```

在以下示例中，报告的 vCPU 使用率截然不同：

- ASAv 虚拟报告：40%
- DP：35%
- 外部进程：5%
- ASA（作为 ASA 虚拟报告）：40%
- ASA 空闲轮询：10%
- 开销：45%

开销用于执行虚拟机监控程序功能，以及使用 vSwitch 在 NIC 与 vNIC 之间移动数据包。

KVM CPU 使用情况报告

在传出数据包通过以太网微处理器退出前，此

`virsh cpu-stats` *`domain`* `--total start count`

命令提供有关指定访客虚拟机的 CPU 统计信息。默认情况下，它会显示所有 CPU 的统计信息以及总数。`--total` 选项将仅显示总统计信息。`--count` 选项将仅显示计数 CPU 的统计信息。

OProfile、top 等工具可提供特定 KVM VM 的总 CPU 使用率，其中包括虚拟机监控程序和 VM 的 CPU 使用率。同样，XenMon 等特定于 Xen VMM 的工具会提供 Xen 虚拟机监控程序的总 CPU 使用率（即 Dom 0），但不会将其划分为每个虚拟机的虚拟机监控程序使用情况。

除此之外，云计算框架中还提供了某些工具，例如 OpenNebula，它仅提供 VM 使用的虚拟 CPU 百分比的粗略信息。

ASA 虚拟和 KVM 图形

ASA 虚拟与 KVM 之间的 CPU 使用率 (%) 存在差异：

- KVM 图表值始终大于 ASA 虚拟值。
- KVM 称之为 %CPU 使用率；ASA 虚拟称之为 %CPU 利用率。

术语“%CPU 利用率”和“%CPU 使用率”表示不同的东西：

- CPU 利用率提供了物理 CPU 的统计信息。
- CPU 使用率提供了基于 CPU 超线程的逻辑 CPU 统计信息。但是，由于只使用一个 vCPU，因此超线程未打开。

KVM 按如下方式计算 CPU 使用率 (%)：

当前使用的虚拟 CPU 的用量，以总可用 CPU 的百分比表示

此计算值是基于主机的 CPU 使用率，而不是基于来宾操作系统，是虚拟机中所有可用虚拟 CPU 的平均 CPU 利用率。

例如，如果某个带一个虚拟 CPU 的虚拟机在一个具有四个物理 CPU 的主机上运行且 CPU 使用率为 100%，则该虚拟机已完全用尽一个物理 CPU。虚拟 CPU 使用率计算方式为：以 MHz 为单位的使用率/虚拟 CPU 数量 x 核心频率

当地语言翻译版本说明

思科可能会在某些地方提供本内容的当地语言翻译版本。请注意，翻译版本仅供参考，如有任何不一致之处，以本内容的英文版本为准。