# KVM installation

yum install kvm qemu-kvm libvirt libvirt-python virt-install bridge-utils libvirt-python

dnf group install Virtualization

libvirt is used to control/manage qemu/kvm instances and various associated features of VM domains.

Check if libvirtd is running..

pgrep -lfa libvirtd

If nothing shows up then:

systemctl start libvirtd.service

KVM notes:

How to undefined a domain

virsh destroy centos7.4patched

virsh undefine centos7.4patched

Create a domain from a given image file.

Example clone script:

[**clone-vm.sh**](https://gist.github.com/makuk66/5233855#file-clone-vm-sh)

|  |  |
| --- | --- |
|  | #!/bin/bash |
|  |  |
|  | VM=$1 |
|  | size=`sudo lvs -o lv\_size --unit=b --noheadings /dev/vg\_vms/ubuntu-base-vm | sed 's/^ \*//'` |
|  | echo size=$size |
|  | sudo lvcreate --size=$size --name=vms-$VM vg\_vms |
|  | sudo virt-resize --expand sda1 \ |
|  | /dev/vg\_vms/ubuntu-base-vm /dev/vg\_vms/vms-$VM |
|  | mkdir -p tmp |
|  | virsh dumpxml ubuntu-base-vm > tmp/ubuntu-base-vm.xml |
|  | mac=`egrep "^$VM"'\s' ips.txt | awk '{print $3}'`; echo $mac |
|  | python ./modify-domain.py \ |
|  | --name $VM \ |
|  | --new-uuid \ |
|  | --device-path=/dev/vg\_vms/vms-$VM \ |
|  | --mac-address $mac \ |
|  | < tmp/ubuntu-base-vm.xml > tmp/$VM.xml |
|  | virsh define tmp/$VM.xml |
|  | virsh dumpxml $VM |
|  |  |
|  | ip=`egrep "^$VM\s" ips.txt | awk '{print $2}'`; echo $ip |
|  | sed -e "s/IP\_ADDRESS\_GOES\_HERE/$ip/g" -e "s/VM\_NAME\_GOES\_HERE/$VM/g" < templates/hosts > tmp/hosts.$VM |
|  | sed -e "s/IP\_ADDRESS\_GOES\_HERE/$ip/g" -e "s/VM\_NAME\_GOES\_HERE/$VM/g" < templates/network-interfaces > tmp/network-interfaces.$VM |
|  | sed -e "s/IP\_ADDRESS\_GOES\_HERE/$ip/g" -e "s/VM\_NAME\_GOES\_HERE/$VM/g" < templates/configure.sh > tmp/configure.sh.$VM |
|  | chmod a+x tmp/configure.sh.$VM |
|  | sudo virt-sysprep -d $VM \ |
|  | --verbose \ |
|  | --enable udev-persistent-net,bash-history,hostname,logfiles,utmp,script \ |
|  | --hostname $VM \ |
|  | --script `pwd`/tmp/configure.sh.$VM |
|  | virsh start $VM |

[**Raw**](https://gist.github.com/makuk66/5233855/raw/030420dda4f2205e0f07703e973d6762e3bb48b7/reset-vm.sh)

[**reset-vm.sh**](https://gist.github.com/makuk66/5233855#file-reset-vm-sh)

snapshot

snapshots are taken before any upgrade changes to guestVM.

Then if anything goes wrong it can be reverted back to the snapshot that was taken.

To get out from a "virsh console": With an english keyboard try:

# Ctrl+]

# Ctrl+5

All those special

Fastest way to create VM

From an existing VM

Use virt-clone.

Step1: shutdown the VM that needs to be cloned.

Step2: use virt-clone

virt-clone --original centos7.4 --auto-clone

note: virt-clone requires python2.. modify /usr/bin/virt-clone with python2 path if needed.

How to pin vcpu to specific logical cores in the host.

virsh vcpupin <domain-name> --vcpu <vpcuno> <actual logical core number from lscpu on the host>

#!/bin/bsh

virsh vcpupin redis-vm --vcpu 1 46

virsh vcpupin redis-vm --vcpu 2 47

virsh vcpupin redis-vm --vcpu 3 48

virsh vcpupin redis-vm --vcpu 4 49

virsh vcpupin redis-vm --vcpu 5 50

virsh vcpupin redis-vm --vcpu 6 51

virsh vcpupin redis-vm --vcpu 7 52

virsh vcpupin redis-vm --vcpu 8 53

virsh vcpupin redis-vm --vcpu 9 54

How to use SRIOV to create network interface for KVMs.

echo 16 >  /sys/class/net/enp24s0f0/device/sriov\_numvfs

change the device to your device. Above creating 16 functions to be put on the NIC.

virt-install -n tile16-web -r 12288 --vcpus=3 --cpuset=28-55,84-111 --os-type=linux --os-variant=rhel6 --accelerate --disk path=/opt/vmimages2/webserver16.img,format=raw,bus=virtio,cache=writeback --import --nonetworks  --host-device=pci\_0000\_86\_03\_7 --noautoconsole

SRIOV -> Single Root I/O Virtualization.

SRIOV : is a specification that allows a PCIe device to appear as multiple separate physical devices that can be shared between many virtual machines on the same host bypassing the hypervisor layer thus achieving network speed.

AWS expose this feature for its EC2 compute through API calls.

BUsinfo for network interface

lshw -c network –businfo

pci@0000:03:00.0 eth0 network I350 Gigabit Network Connection

pci@0000:05:00.1 eth3 network Ethernet Controller XL710 for 40GbE QSFP+

vnet0 network Ethernet interface

# BKM to enable sriov

Make sure Inte VT-X is enabled. Which by default us enabled in SKX.

Make sure Intel VT-D is enabled in BIOS. Can be done by pythonSV as below..

import misc.SvBios.SvBiosCli as cli

cli.CvProgKnobs("VTdSupport=1")

[root@bdx-prt-sys-1 device]# lshw -c network -businfo

Bus info Device Class Description

========================================================

pci@0000:03:00.0 eth0 network I350 Gigabit Network Connection

pci@0000:03:00.3 eth1 network I350 Gigabit Network Connection

pci@0000:05:00.0 eth2 network Ethernet Controller XL710 for 40GbE QSFP+

pci@0000:05:00.1 eth3 network Ethernet Controller XL710 for 40GbE QSFP+

virbr0-nic network Ethernet interface

vnet0 network Ethernet interface

[root@bdx-prt-sys-1 device]# echo 2 > sriov\_numvfs

[root@bdx-prt-sys-1 device]# lshw -c network -businfo

Bus info Device Class Description

========================================================

pci@0000:03:00.0 eth0 network I350 Gigabit Network Connection

pci@0000:03:00.3 eth1 network I350 Gigabit Network Connection

pci@0000:03:10.3 eth4 network Illegal Vendor ID

pci@0000:03:10.7 eth5 network Illegal Vendor ID

pci@0000:05:00.0 eth2 network Ethernet Controller XL710 for 40GbE QSFP+

pci@0000:05:00.1 eth3 network Ethernet Controller XL710 for 40GbE QSFP+

virbr0-nic network Ethernet interface

vnet0 network Ethernet interface

* From BIOS
* Enable IIO VT-D
* Then modify grub to include intel\_iommu=on
* Reboot system.

Domain=0x0

Bus=0x3

Slot=0x10

Function=0x3

Then launch KVMs with virt-install

Vm1

virt-install -n vm-sriov -r 12288 --vcpus=3 --cpuset=23-41 --os-type=linux --os-variant=rhel6 --accelerate --disk path=/home/vm/centos7.4-clone.img,format=raw,bus=virtio,cache=writeback --import --nonetworks --host-device=pci\_0000\_03\_10\_3 –noautoconsole

VM2

virt-install -n vm-sriov2 -r 12288 --vcpus=3 --cpuset=23-41 --os-type=linux --os-variant=rhel6 --accelerate --disk path=/home/vm/redis-vm.img,format=raw,bus=virtio,cache=writeback --import --nonetworks --host-device=pci\_0000\_03\_10\_7 –noautoconsole

Note: It create network interface ens5 inside the VM.

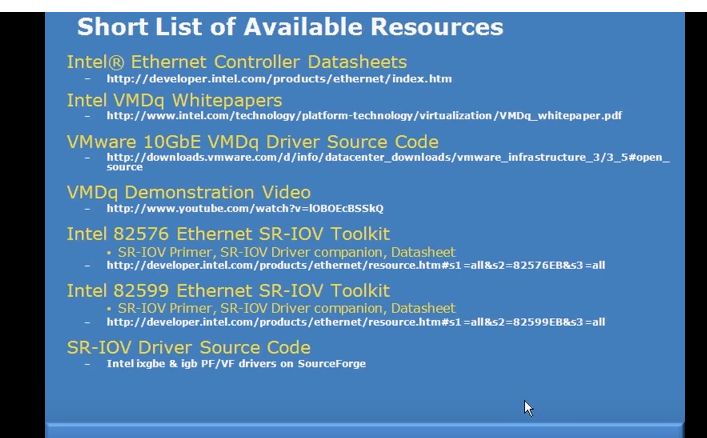
Use ifconfig to assign IP with possibly a different subnet then the other existing subnet.

On the host netstat –rn

Routetable gets automatically updated.

Note: two VMs were only able to ping each other only after the PF (physical interface eth1 where the two VFs were created) was enabled by plugging into a switch. Then both ens5

SRIOV documentation/resources

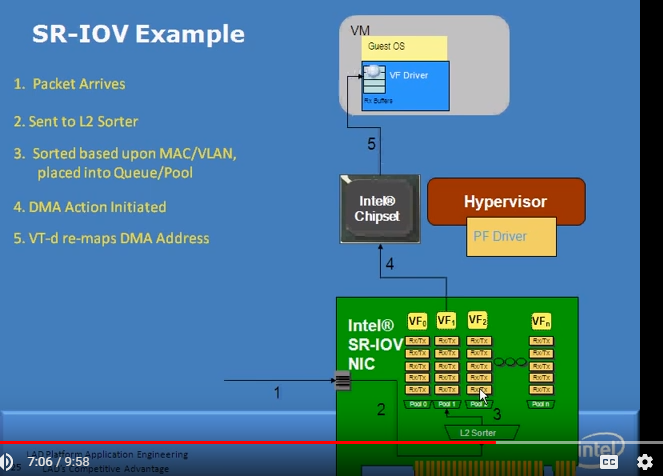


In short summary:

SRIOV creates VF which have direct access to resources ( memory descriptors ) from where and to where it can directly copy data without needing the hypervisor to copy it to and from specific guest OS.

Reduces interrupt from NIC to hypervisor.

Resources



DMA from NIC to VM needs Intel VF-driver to map virtual address to physical address.

That’s VT-d enabling is needed.

Attaching a virtual nic to a KVM domain live of virtio type.

First create a linux bridge with brctl if linux bridge not there.

**brctl addbr virbr0**

**ip addr add 192.168.122.1 dev virbr0**

**brctl stp virbr0 on**

**Next:**

virsh attach-interface --domain vm-sriov --type bridge --source virbr0 --target vsrx-mgmt --model virtio –live

From within the VM set ip to the interface vsrx-mg

Ifconfig eth0 192.168.122.42

# Some debug of no connection from the host to the VM.

Netstat –rn was showing

Two interfaces from 192.168.122 ( since both sriov and as well as this bridge was on he same subnet)..

Changed that.

tcpdump -i eth0 [ shows alla request coming to the interface eth0]