BKM for Creating the Virtual Machine

By: Rohan Tabish

Manager: Ripan Das

Group: Data Center Group

# Introduction:

This document describes the simple steps to create VM using qemu KVM and shows how to have internet connectivity in the QEMU KVM.

# Creating VM and Updating the Kernel inside the VM for BPF:

The NIC card interface ens1f0 on the host has internet connectivity in this example. You should use the one your computer has.

1. Create the configuration files for the bridge you want to attach to your VM. In example below corp is the bridge and ens1f0 is the Ethernet interface.

|  |
| --- |
| # vim /etc/sysconfig/network-scripts/ifcfg-corp  BOOTPROTO=dhcp  TYPE=Bridge  ONBOOT=yes  DEVICE=corp  NAME=corp  #BRIDGE=corp  NM\_CONTROLLED=no |

|  |
| --- |
| # vim /etc/sysconfig/network-scripts/ifcfg-ens1f0  ONBOOT=yes  DEVICE=ens1f0  NAME=ens1f0  BRIDGE=corp  NM\_CONTROLLED=no |

1. Create the bridge and attach the interface to the bridge

|  |
| --- |
| # brctl addbr corp  # brctl addif corp ens1f0  # ifup ens1f0  # ifup corp |

1. Download the fedora 30 Server image from the website and create the virtual machine using command below.

|  |
| --- |
| # virt-install --name fed30 --ram 16384 --vcpus 8 --disk path=/vmdisk/fed30.img,size=100 --os-variant fedora28 --os-type linux --network bridge=corp --network=default --graphics none --console pty,target\_type=serial --location /vmdisk/Fedora-Server-dvd-x86\_64-30-1.2.iso --extra-args 'console=ttyS0,115200n8 serial' |

1. The name of the VM in above example is fed30 and bridge attached is corp. Follow steps for installation on command line. After installation, the VM will boot.
2. Once you are booted into the virtual machine you can run following commands to resize

|  |
| --- |
| # vgs  # lvresize -L +75G --resizefs /dev/mapper/fedora-root |

1. If VM machine did not get the ip, run the following command:

|  |
| --- |
| # dhclient <etheret interface> |

1. Proxy settings for the VM

|  |
| --- |
| # vim /etc/envirnoment  # http\_proxy=http://proxy-chain.intel.com:922  # https\_proxy=http://proxy-chain.intel.com:912  # ftp\_proxy=http://proxy-chain.intel.com:911  # socks\_proxy=http://proxy-us.intel.com:1080  # no\_proxy=intel.com,.intel.com,localhost,127.0.0.1 |

1. In order to update the kernel inside the VM you can run following commands. After reboot from the grub menu select the new kernel.

|  |
| --- |
| # yum install vim  # yum install make  # yum install gcc  # yum install flex  # yum install bison  # yum install openssl  # yum install libssl-dev  # yum install libssl-devel  # yum search libssl  # yum install openssl-devel  # yum install libelf-dev libelf-devel  # yum install elfutils-libelf-devel  # yum install perl |

|  |
| --- |
| # cd /usr/src/  # wget https://cdn.kernel.org/pub/linux/kernel/v5.x/linux-5.2.7.tar.xz  # tar -xvf linux-5.2.7.tar.xz  # uname -r  # cp /boot/config-4.19.8-300.fc29.x86\_64 .config  # make oldconfig  # make bzImage -j50  # make modules -j50  # make modules\_install -j50  # make install  # reboot |

1. Once the kernel is installed, we can install the bpf tools by running the following commands:

|  |
| --- |
| cd /usr/src/linux-5.2.7/tools/bpf  yum install llvm7.0-devel.x86\_64  yum install clang  yum install binutiles-devel  yum install binutils-devel  yum install readline-devel  yum install llvm  yum install gmp-devel  make install  cd ../../samples/bpf  make |

# Adding SRIOV from host and attaching a sriov virtual as a host device to the VM

Host interface connected to the switch ( all the client2-client31 are connected to the switch). On local subnet 192.168.232.\*

1. Enabling SRIOV: From pythonSV
2. import misc.SvBios.SvBiosCli as cli
3. cli.CvProgKnobs("VTdSupport=1")
4. Then if grub is not mounted on the right mount /dev/sda1 to /tmp and then modifiy
5. /EFI/fedora/grub.cfg file to include the following line and rebootlinuxefi /vmlinuz-4.19.8-300.fc29.x86\_64 root=UUID=84062c66-403d-- 4d6b-8d33-8f557ae86693 ro selinux=0 console=ttyS0,115200 modprobe.blacklist=qat\_\_ c62x LANG=en\_US.UTF-8 **intel\_iommu=on**
6. Attaching sriov hostdevice to the VM
   1. After enabling VT-D and iommou as above, identify the interface ( physical NIC)
   2. In this case enp109s0f
   3. echo 2 > /sys/class/net/enp109s0f2/device/sriov\_numvfs
   4. virsh nodedev-list --cap pci | grep 6d\_0a [ will show the pci device ]
   5. create a sriov\_interface.xml file
   6. bus, slot etc info can be found from virsh nodedev-dumpxml pci\_0000\_6d\_0a\_0
   7. virsh nodedev-detach pci\_0000\_6d\_0a\_0
   8. <interface type='hostdev' managed='yes'>
   9. <source>
   10. <address type='pci' domain='0' bus='109' slot='10' function='0'/>
   11. </source>
   12. </interface>
7. virsh attach-device fed30 sriov\_interface.xml
8. go inside the VM
   1. Create a ifcfg-<ens9> file
      1. NAME=ens9
      2. DEVICE=ens9
      3. IPADDR=192.168.232.235
      4. NETMASK=255.255.255.0
      5. ONBOOT=yes
      6. USERCTL=no
      7. BOOTPROTO=static
9. Reboot. SRIOIV interface with routetable and ip updated should be avaialable now.

# Update VM GCC

1. Install gcc (GCC) 8.3.0 in the VM.