All the following is done on a base install of Debian Lenny running a x64 kernel (2.6.26-2-amd64). In the case below it was done on a remote server with only ssh access to the host. LVM was already setup on the Debian Lenny Host and is used to give a LV to the VMs.

Contents

Setup of KVM and Libvirt on Debian Lenny Host

```
apt-get install kvm libvirt-bin virtinst
#libvirt allows easy management of KVM, ensuring VMs start upon boot
```

(Default) Network Config

The default network config that comes out of the box with KVM gives a private IP address to each VM via local DHCP & DNS server (dnsmasq). Each VM can then NAT out through the host to get internet access. There is an issue with dnsmasq and possibly a bug with debian which is overcome below by preventing dnsmasq to start on boot of the host and allowing libvirt instead of control dnsmasq.

```
#Type "virsh" to enter the virsh console:
virsh
virsh #                          //You are now in the virsh console
virsh # nodeinfo                 //Shows info on the Host
virsh # list --all               //Shows all VMs. (Same as going ~# ls /etc/libvirt/qemu/)
virsh # net-list --all           //Shows all the network config files. (Same as going ~# ls /etc/libv
virsh # net-edit default         //It is BEST to edit network config files using vish.
                                 //If you edit /etc/libvirt/qemu/networks/default.xml manually, it wo
virsh # quit

#DHCP of the VMs is provided by dnsmasq. We need to stop the dnsmasq service from starting itself an
/etc/init.d/dnsmasq stop
sysv-rc-conf (apt-get install sysv-rc-conf if its not installed)
#untick dnsmasq for all runlevels
```

```
virsh
virsh # net-autostart default
virsh # net-start default
virsh # quit
```

Setup of Debian Lenny VM

```
wget http://ftp.debian.org/debian/dists/stable/main/installer-amd64/current/images/netboot/mini.iso
lvcreate -n deb01 --size 20g vg0   //For file based disk: dd if=/dev/zero of=/kvm/deb01.img bs=1024k
virt-install -d --name=deb01 --ram 512 --disk path=/dev/vg0/deb01 --network network=default --vnc --
# --accelerate IS required, otherwise it will be qemu and virtio won't work later when we config it
#After virt-install, you will see (virt-viewer:3501): Gtk-WARNING **: cannot open display:
#Domain installation still in progress. You can reconnect to the console to complete the installatio

#Connect with vncviewer. You may have to ssh to the server and port forward port 5901.
#netstat -tap //This will show the ports open and if they are bound to 127.0.0.1
#Complete the install. Debian will shutdown the VM after installation

virsh
virsh # list --all
virsh # start deb01
#Connect with VNCviewer and apt-get install ssh
```

Benchmark

I ran many dd tests and hdparm however the results seemed too good due to KVM caches etc. As a result the following phoronix benchmark tests proved to work very well for comparing IO operations.

```
vi /etc/apt/sources.list
#Add the following line:
deb http://www.phoronix-test-suite.com/releases/repo pts.debian/
apt-get update
apt-get upgrade
apt-get install phoronix-test-suite
phoronix-test-suite benchmark apache            //Scored Average: 8115.92 Requests Per Second
phoronix-test-suite benchmark compress-gzip     //Scored Average: 20.61 Seconds
```

Change from IDE to Virtio

I was mainly interested in IO performance of VMs. I've noticed a little iowait on my current Xen setup and wanted to see how KVM would work.

```
virsh
virsh # shutdown deb01
virsh # edit deb01
#change <target dev='hda' bus='ide'/> to:
<target dev='vda' bus='virtio'/>

#add in <model type='virtio'/> under the network interface to look like:
   <interface type='network'>
     <mac address='54:52:00:22:33:50'/>
     <source network='default'/>
     <model type='virtio'/>
```

```
     <target dev='vnet0'/>
   </interface>
:wq
```

Mount LVM VM Disk

```
See: http://wiki.kartbuilding.net/index.php/Mount_kvm_file_based_image_(disk.img)_on_host_computer
kpartx -av /dev/vg0/deb01
mount /dev/mapper/vg0-deb01p1 /mnt/
vi /mnt/boot/grub/menu.lst
#change /boot/vmlinuz-2.6.26-2-amd64 root=/dev/hda1 ro quiet to:
/boot/vmlinuz-2.6.26-2-amd64 root=/dev/vda1 ro
vi /mnt/etc/fstab
#Change hda* to vda*
umount /mnt/
kpartx -dv /dev/vg0/deb01
virsh start deb01
```

Custom Network Config

So my KVM host IP is x.x.x.16 My hosting provider also allow me 3 additional IPs: x.x.x.35, x.x.x.36 and x.x.x.61 I am using this config below AND the default KVM Network config at the top of this page.

```
vi /etc/network/interfaces
auto br0
iface br0 inet static
  address x.x.x.16
  netmask 255.255.255.255
  bridge_stp off
  bridge_fd 0
  pre-up brctl addbr br0
  #pre-up brctl addif br0 vnet3                    //If VMs are running and init.d/networking restar
  up ip route add x.x.x.35 dev br0
  up ip route add x.x.x.36 dev br0
  up ip route add x.x.x.61 dev br0
  post-down brctl delbr br0                       //Allows for smooth use of /etc/init.d/networking

ifup br0
#check with "route" to see the entry.
```

The virt-install config line is now:

```
virt-install -d --name=deb03 --ram 512 --disk path=/dev/vg0/deb03 --network bridge=br0 --vnc --accel
```

The resulting xml network config is:

```
virsh edit deb03
   <interface type='bridge'>
     <mac address=/>
     <source bridge='br0'/>
     <target dev='vnet2'/>
   </interface>
```

When completing the installation after virt-install using VNC, DHCP fails (of course) and Manual Network Configuration has to be done. Supply the fixed IP (x.x.x.35), the Default Gateway (Host IP x.x.x.16) and DNS server IP. That will then allow the netinstall to complete.

Custom Iptables Firewall with libvirt

As I wanted to use libvirt to create a default network with dhcp, I had to work around the fact that libvirt wanted to append its own iptables rules which were a little restrictive. I wanted VMs on the Private LAN to see other VMs on br0 and visa versa. As libvirt appended iptables rules, there was an issue. I tried removing the <forward mode='nat'> however libvirt still added its own iptables rules.

Firewall Script

```
vi /etc/init.d/firewall
#!/bin/sh
#This gets called by /etc/rc.local
IPTABLES=/sbin/iptables

EXTBR=br0
INTBR=virbr0

PRIVATE=192.168.1.0/24

$IPTABLES -F INPUT
$IPTABLES -F OUTPUT
$IPTABLES -F FORWARD
$IPTABLES -F POSTROUTING -t nat
$IPTABLES -F PREROUTING -t nat

######################
# DNS and DHCP
######################
$IPTABLES -A INPUT -i $INTBR -p udp -m udp --dport 53 -j ACCEPT
$IPTABLES -A INPUT -i $INTBR -p tcp -m tcp --dport 53 -j ACCEPT
$IPTABLES -A INPUT -i $INTBR -p udp -m udp --dport 67 -j ACCEPT
$IPTABLES -A INPUT -i $INTBR -p tcp -m tcp --dport 67 -j ACCEPT

####################
# FORWARDS
####################
$IPTABLES -A FORWARD -d $PRIVATE -o $INTBR -m state --state RELATED,ESTABLISHED -j ACCEPT
$IPTABLES -A FORWARD -s $PRIVATE -i $INTBR -j ACCEPT
$IPTABLES -A FORWARD -i $INTBR -o $INTBR -j ACCEPT
$IPTABLES -A FORWARD -i $EXTBR -o $EXTBR -j ACCEPT

####################
# NATTING
####################
$IPTABLES -t nat -A POSTROUTING ! -d $PRIVATE -s $PRIVATE -j MASQUERADE
$IPTABLES -t nat -A POSTROUTING ! -s $PRIVATE -d $PRIVATE -j MASQUERADE

####################
# PORT FORWARDING (Remote Desktop)
####################
$IPTABLES -t nat -A PREROUTING -p tcp --dport 8883 -j DNAT --to 192.168.1.42:3389
$IPTABLES -t nat -A PREROUTING -p tcp --dport 8884 -j DNAT --to 192.168.1.152:3389
```

```
###################
# BLOCKING
###################
$IPTABLES -A FORWARD -j REJECT --reject-with icmp-port-unreachable
```

Control the Firewall

We need to let libvirt start the default network and then run our firewall. Note: this is much less than ideal. The alternative would be not to use a libvirt network config, create a second bridge, do the dnsmasq manually and iptable rules.

```
vi /etc/rc.local

#add in the following
while [ `ps -e | grep -c libvirtd` -lt 1 ]; do
       sleep 1
done
sleep 10
/etc/init.d/firewall

exit 0
```

The firewall can also be called manually anytime by going /etc/init.d/firewall

Ref: http://efreedom.com/Question/2-170079/Forwarding-Ports-Guests-Libvirt-KVM

Munin Plugins for Libvirt / KVM

Debian squeeze has plugins for libvirt available via apt. Lenny does not however. The following worked ok for me on Lenny:

```
//taken from http://honk.sigxcpu.org/projects/libvirt/
wget http://honk.sigxcpu.org/projects/libvirt/monitor/releases/munin-libvirt-plugins-0.0.6.tar.gz
gunzip munin-libvirt-plugins-0.0.6.tar.gz
tar -xvf munin-libvirt-plugins-0.0.6.tar
#explore the directory and move the libvirt-cpu -ifstat etc. into /usr/share/munin/plugins and chmod
```

Setup of Windows 7 VM

```
lvcreate -n win01 --size 20g vg0
virt-install -d --name=win01 --ram 512 --disk path=/dev/vg0/win01 --network network=default --vnc --
```

Correct Mouse Pointer issue with VNC

```
virsh shutdown win01
virsh edit win01
//add in the following line above the mouse config:
<input type='tablet' bus='usb'/>
```

```
//afterwards it will look like:
   <input type='tablet' bus='usb'/>
   <input type='mouse' bus='ps2'/>
   <graphics type='vnc' port='5905' autoport='yes' keymap='en-us'/>
 </devices>
</domain>
```

Refs: http://wiki.libvirt.org/page/UbuntuKVMWalkthrough

IPv6 Config on Host

So my hosting company gave me a ipv6 /64 subnet. They also gave me a default gateway to which a route had to be added. Also debian lenny required "pre-up modprobe ipv6" to work correctly.

- IPv6 Subnet: 2a01:4f8:100:xxx1::
- IPv6 Host IP: 2a01:4f8:100:xxx1::1
- IPv6 Default Gateway: 2a01:4f8:100:xxx0::1

```
iface eth0 inet6 static
  address 2a01:4f8:100:xxx1::1
  netmask 64
  gateway 2a01:4f8:100:xxx0::1
  pre-up modprobe ipv6
  pre-up ip -6 route add 2a01:4f8:100:xxx0::1 dev eth0
```