KVM on personal workstation

Discussion

WARNINIG: I'm still writing this, and most of it is coming off my (fallible) memory, so be careful if you use this before I've tested everything. I'll remove this warning when I have tested.

I have had great success with KVM on a workstation. If you like GUI's, virt-manager takes some of the pain out of learning a new system, at the loss of some capabilities. However, I was able to get very good response on an older Dell Inspiron 23 with 8G of RAM and a 4 core i5. I was able to run Windows 10, Windows 7, FreeBSD and a Devuan workstation on them (one at a time) and get very good response most of the time.

On my Lenovo ThinkStation D30, 32G RAM and 8 core Xeon E5-2637, I run multiple virtuals at the same time, depending on what I need.

In this case, want the virtuals visible on the LAN, so I set up a bridge for them and let them get their IP's from the LAN's DHCP server. However, on my laptop, which may not have a network connection, I use the internal NAT.

Installation

First, install the necessary Devuan packages. I always do my installation as the root user, so you won't see 'sudo' before any of the commands:

```
apt install bridge-utils qemu-kvm qemu-system-common qemu-system-x86 qemu-utils virt-manager virt-top virt-viewer virtinst xrdp xtightvncviewer libvirt-clients libvirt-daemon-system
```

Now, build out the network using a bridge (don't do this if you're going to use virt-manager's NAT). Edit /etc/network/interfaces. I put the whole file here, so you can just move the old one and create a new one with the contents.

interfaces

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
iface eth0 inet manual
```

```
iface eth0 inet6 manual

#set up bridge and give it a static ip
auto br0
iface br0 inet dhcp
        bridge_ports eth0
        bridge_stp off
        bridge_fd 0
        bridge_maxwait 0
```

Usage

virt-manager has a nice little GUI interface to running virtuals, but I prefer to connect to them using xtightvncviewer or rdesktop, so I install those.

```
apt install xtightvncviewer rdesktop
```

Windows Virtuals

On my Windows virtuals, I simply turn on Remote Desktop ('pro' version or above). I wrote a little script that I can run with some defaults I like, so I just call it with either the IP or the DNS name:

rdesktop.pl

```
#! /usr/bin/env perl

use warnings;
use strict;

# these will be available to the Windows machines as network shares.
The key
# is the name as seen by the Windows machine, the value is the actual
path.

my %shares = ( 'home' => '/home/me', 'installs' =>
'/map/to/some/other/share' );

my $target = shift or die "Usage: $0 hostname\n";
my $resolution = shift; $resolution = '1680x1050' unless $resolution;
# build my command
my $command = "/usr/bin/rdesktop -g $resolution -P ";
foreach my $share ( keys %shares ) {
   $command .= "-r disk:$share='$shares{$share} ";
}
$command .= $target;
# and execute it
```

```
    qx/$command/;
```

Just call it as

```
./rdesktop.pl targetname [resolution]
```

and the session is set for your machine.

Unix VNC Targets

Again, I have a quick and dirty script that I use to connect.

connectVNC

```
#! /usr/bin/env perl

use warnings;
use strict;

my $target = shift  or die "Usage: $0 hostname [port]\n";
my $port = shift; $port = 5900 unless $port;
`vncviewer -compresslevel 6 -encodings "copyrect tight hextile zlib
corre rre raw" -quality 5 $target:$port`;
```

Call it with ./connectVNC ip.or.dnsname 5900

For Linux, there is a cute little script that works well that you can install on the target virtual. I'll include it here later.