Using KVM on vpsFree.cz

This information page contains manuals on how to install and run KVM on Alpine Linux 3.4+, CentOS 7 and Debian 8.

First, go the details of your VPS in vpsAdmin and turn on the following options:

- Bridge – enables the creation of a network bridge that the KVM virtual machines will be connected to later,
- iptables – enables iptables, an option necessary to configure the IP masquerade,
- KVM – enables KVM (for hardware support of virtualization).

KVM on Alpine Linux

Install the required packages (ip6tables is optional):

```
apk update
apk add qemu-system-x86_64 qemu-openrc qemu-img bridge iptables ip6tables
```

Configure the bridge for Qemu/KVM virtual machines – create the */etc/network/interfaces.tail* file (of course, you can choose any IP address from your private range):

```
auto br0
iface br0 inet static
     pre-up brctl addbr br0
     address 172.17.1.1
     netmask 255.255.255.0
     post-down brctl delbr br0
```

Since OpenVZ rewrites the */etc/network/interfaces* file rather clumsily (this is also why we add the configuration of the bridge to *interfaces.tail*), it is safest to restart the container at this point.

Give the user in the *qemu* group permissions to manage the newly-created bridge:

```
echo "allow br0" > /etc/qemu/bridge.conf
chown root:qemu /etc/qemu/bridge.conf
chmod 0640 /etc/qemu/bridge.conf
```

Configure the IP masquerade so that the Qemu/KVM virtual machines have access to the public Internet.

If you have configured iptables, all you need to add is this rule:

```
iptables -t nat -A POSTROUTING -s 172.17.1.0/24 ! -o br0 -j MASQUERADE
```

If not, you can follow our paragraph on configuring iptables.

Creating and Running a Virtual Machine

This manual presupposes that you will be using qemu-openrc – an OpenRC init script used to start Qemu/KVM. Of course, if you're on Alpine, you can use libvirt as well, but do you really want and need its clumsy XML configuration files and/or the click interface above it…? ;) Qemu-openrc is a much simpler and more transparent solution. Each virtual machine is represented by an init script. Just like with other programs, you can declare dependencies between virtual machines, etc.

Creating a new virtual machine consists only of preparing an image disk, creating a symlink for the init script and modifying a simple configuration script. Let's say that the new virtual machine is called "myvirt."

Prepare a raw image for myvirt with the required size:

```
mkdir -p /var/lib/qemu/myvirt/
qemu-img create -f raw /var/lib/qemu/myvirt/disk0.img 5G
chown qemu:qemu /var/lib/qemu/myvirt/disk0.img
chmod 0600 /var/lib/qemu/myvirt/disk0.img
```

Copy the default configuration file /etc/conf.d/qemu to /etc/conf.d/qemu.myvirt and modify it as needed:

```
cd /etc/conf.d
cp qemu qemu.jarvis
vi qemu.jarvis  # read comments and edit
```

Most importantly, add the prepared image:

```
disk1_file="/var/lib/qemu/myvirt/disk0.raw"
disk1_format="raw"
```

You will probably also need to add the installation CD of a distribution that you have already downloaded:

```
cdrom0_file="/var/lib/qemu/alpine-virt-3.4.1-x86_64.iso"
```

Create a symlink for the init script and run myvirt.

```
cd /etc/init.d
ln -s qemu qemu.myvirt

rc-service qemu.myvirt start
```

Configuring iptables

If you aren't using any tool to generate iptables rules (like e.g. ferm), I recommend basing them on [https://gist.github.com/jirutka/3742890|existing rule templates]]. These are already in the iptables format, which can be loaded using iptables-restore.

Download the modified rule template with the added masquerade for our bridge to /etc/iptables:

```
rmdir /etc/iptables
wget -O /etc/iptables http://haste.fit.cvut.cz/raw/iwuqoso
```

Modify the /etc/conf.d/iptables configuration file (IPv4):

```
# /etc/conf.d/iptables

IPTABLES_SAVE="/etc/iptables"
#SAVE_RESTORE_OPTIONS="-c"
SAVE_ON_STOP="no"
IPFORWARD="yes"
```

...and the /etc/conf.d/ip6tables configuration file (IPv6):

```
# /etc/conf.d/ip6tables

IP6TABLES_SAVE="/etc/iptables"
SAVE_RESTORE_OPTIONS="-T filter"
SAVE_ON_STOP="no"
IPFORWARD="yes"
```

Run iptables and ip6tables and add them to the runlevel boot:

```
rc-service iptables start
rc-service ip6tables start
rc-update add iptables boot
rc-update add ip6tables boot
```

Contacts

- Jakub Jirůtka (on #vpsfree under the name “jirutka”)

KVM on CentOS 7

This manual can only be used for CentOS 7.1. The internal network between VPS and VM currently isn’t working on CentOS 7.2. Until this problem is solved, please use either CentOS 7.1 or Debian 8.

I use KVM using libvirt on an updated CentOS 7.

I recommend fully updating CentOS 7, configuring it and installing the required software. Because of iptables permissions, it is necessary to either configure or turn off firewalld.

```
yum group install virtualization-host-environment
yum install virt-manager xauth
```

```
systemctl enable libvirtd
systemctl disable firewalld
reboot
```

Creating a Virtual Machine Using virt-manager on the Side of the Server

Motivation: When you're working on a slow connection (which the ADSL that the O2 provides in villages definitely is), you will need to minimize the data flow through your primary computer. The local virt-manager would download at least the kernel and initramdisk using the relatively slow download speed. The typically extremely slow upload speed would be used to upload to the host container.

```
ssh root@your-host-name -Y virt-manager
```

You can start the installation on the remote running instance. However, displaying the installer using the default SPICE doesn't work for me. Since I think switching libvirt to VNC is extremely impractical, I recommend running virt-manager locally and adding the path to the server for both the installation and further use.

KVM on Debian 8

This manual was made using Debian 8. It should function without issues on CentOS as well (with different commands, see above). I have tried to run KVM on Ubuntu 14.04, but sadly, to no avail.

If you're using Windows, you need to install and turn on Xming (if you at least want to install and configure the virtual machine using a graphical interface, like I did). Then we start Putty (don't forget to run Xming and select the "Enable X11 forwarding" option) and let the magic begin.

We'll start with the usual:

```
apt-get update
apt-get upgrade
```

Then we install the libvirt library:

```
apt-get install qemu-kvm libvirt-bin
apt-get install virt-manager
```

Altogether, these two packages should be around 320 MB. The next step is downloading the image of the distribution that we want to install on the virtual machine. I installed Ubuntu Server 14.04. Choose the folder where you want to download the image and download the ISO image using "wget".

```
cd /home
wget http://releases.ubuntu.com/14.04.3/ubuntu-14.04.3-server-amd64.iso
```

Run virt-manager.

```
virt-manager
```

Using Xming, you can now use Windows to configure and start the installation of the virtual server on your VPS remotely. Note that the SPICE display doesn't work under X11, so if you want to use a graphical interface to install the virtual machine, you need to select the "Display VNC" option in the settings and restart the virtual machine. If you do this and restart the server, it can lose the information that the .iso image for the installation has been connected, so you will have to go back to the settings of the "CD-ROM" tab and in "Source-path" connect the .iso image for the installation again. The next step is installing the chosen Linux distro. The only action left is setting up the correct port forwarding on your VPS so that you can access your virtual machine using the VPS from the outside.

Contacts

- pavlix