Homework #4 Solution

Contact TAs: `vegetable@csie.ntu.edu.tw`

System Administration

1. (20%)

Install packages

```
yum install virt-install qemu-kvm libvirt
```

and start with (1%)

```
systemctl start libvirtd
```

and verify hardware virtualization support with `grep vmx /proc/cpuinfo` or `grep svm /proc/cpuinfo` (1%).

Uncomment the following lines in `/etc/libvirt/libvirtd.conf`. (3%) The first line is to set the group of socket file of libvirt. (2%) The second line is to set the permissions of the socket file. (3%) The third line is to allow a user that has access to the socket file of libvirtd to use libvirt. (3%)

```
unix_sock_group = "libvirt"
unix_sock_rw_perms = "0770"
auth_unix_rw = "none"
```

Then restart `libvrtd` to apply the configuration. (2%)

```
systemctl start libvirtd
```

To enable user `user` to create a VM, do (2%)

```
usermod -aG libvirt user
```

Then for user `user`, to create a VM as usual, simply add `--connect qemu:///system` after original commands. (3%)

(Actually, simply add user to group libvirt is enough.)

2. (25%)

First create a logical volume in volume group `vm-pool` for the VM. (2%)

```
sudo lvcreate vm-pool -n vm-image -L 10G
```

Get a kickstart script from previously installed CentOS and make it accessible through http. (2%) Then create the VM with `virt-install`. (2%)

```
virt-install \
   --name=vm-test \
   --disk path=/dev/vm-pool/vm-image \
   --graphics spice,password=pwd \
   --vcpus=2 --ram=512 \
   --location=/var/lib/libvirt/images/CentOS7-7-x86_64-Minimal-1611.iso \
   --network network=default \
   --extra-args 'ks=[url of ks]'
```

- –name: The name of the VM, as shown in the list of virt-manager. (1%)
- –disk: Specify the attributes of the disk used by the VM. Here only one attribute is used. That is, the path of the virtual disk on the VM host. (3%)
- –graphics: Specify how clients can access the graphic console of the VM. (2%) Here spice is specified, which means that clients can access the graphic console with spice protocol and the password is pwd. (2%)
- –vcpu: Number of “virtual CPU”the VM has. (1%)
- –ram: Number of MB of ram the VM has. (1%)
- –location: Location of installation media. Here an iso file is used. (2%)
- –network: Use the virtual network named “default”, which is NAT. (3%)
- –extra-args: Extra arguments that will be passed to kernel for installation. (2%) Here argument `ks` is used to make the CentOS installer follow the kickstart script. (2%)

If your commands can not be run, you will get at most 15pts for this problem.

Reference

3. (10%)

`virsh console` simulates an accessing serial port of a server (4%), and usually it is RS232. Usually we use a converter to convert it into USB interface and connect it with another computer to read it. (4%)

Picture of the RS232 cable (2%)

創用 CC 姓名標示-相同方式分享 3.0, https://commons.wikimedia.org/w/index.php?curid=33732
(If one does not give credit to the author of the image, they will not get the 2 points.)

4. (15%)

Add three lines to /etc/default/grub file: (5%)

```
GRUB_CMDLINE_LINUX_DEFAULT="console=tty0 console=ttyS0,38400n8"
GRUB_TERMINAL="console serial"
GRUB_SERIAL_COMMAND="serial --speed=38400 --unit=0 --word=8 --parity=no --stop=1"
```

where the first line is to tell kernel to use both console `tty0` and serial console `ttyS0` when GRUB bring up the system (2%); the second line is to tell GRUB to use both the graphical console and the serial console to show menu (2%); the third line is to tell GRUB what arguments to use for the serial console (2%).

Then run `grub2-mkconfig -o /boot/grub2/grub.cfg` (2%) to update the grub configuration. (2%)

5. (15%)

Open `nmtui`, delete all profiles except `virbr0`. Then `Add` > `Bridge`. Fill device name with `br0`, and then `Add` > `Ethernet`, where fill field device with `eth0`, which is the interface you want to bridge. Run `systemctl restart network` to apply those configurations. To get libvirtd notice the existence of the new bridge, run `systemctl restart libvirtd`. Now a bridge has been created. Then run `virsh edit [VM]`, find where interfaces are defined, insert lines

```
<interface type='bridge'>
    <source bridge='br0'/>
</interface>
```

that specify an interface for the VM.

6. (10%)

- virsh list
- virsh undefine [VM]
- virsh domiflist [VM]
- virsh detach-interface [VM]
- virsh edit [VM]

7. (10% bonus)

- NAT: The function of NAT is achieved by iptables. Compared with the other modes, it is the simplest and can not be accessed from outside the network.
- macvtap: Macvtap is the combination of macvlan and tap. What macvlan does is to let an physical iterface have multiple MAC addresses, so there are like multiple virtual interfaces on a single physical interface. What TAP does is kind of connection between VM's virtual interface and the virtual interface simulated on the phycal interface. So traffic from a VM is forwarded with TAP, and then is sent out from MAC VLAN. Compared with the other modes, if a VM host wants to communicate with its guest VM, support of "hairpin mode"for switch connected to the VM host is required.

- routed: Traffic from a VM is routed outside. Compared with the other modes, the traffic is of IP layer.
- Linux bridge: Linux bridge simply forwords L2 frames for interfaces connected to it. Compared with macvtap mode, if a VM host wants to communicate with its guest VM, no hairpin mode support is required.

If you want your VM to connect a VLAN trunk, both macvtap mode and bridge mode can be choosed.