

EI313 课程大作业二

杨凯翔 519030910240, flying_feixiang@sjtu.edu.cn

October 22, 2021

1 作业内容

1. 从官网下载 QEMU5.2.0 并编译
2. 利用 QEMU 创建两个虚拟机并设置为 tap 网络模式，虚拟网卡类型分别为：e1000 和 virtio-net
3. 通过 VNC viewer 或者 SSH 连接虚拟机
4. 比较两种有着不同种虚拟网卡的虚拟机和主机之间的网络性能

由于 tap 网络模式通过网桥将虚拟网卡与主机网卡相连，所以虚拟机都是通过主机的网卡访问外网，所以我们需要测试两种虚拟机和主机间的网络性能。为了便日后学习与研究，我安装了ubuntu双系统，并在 ubuntu 上完成作业内容。

2 安装双系统

1. 选择 ubuntu-20.10 来安装，从官网下载镜像 (<https://ubuntu.com/download/desktop>) 并用 Rufus 工具制作系统盘如图 1 所示。
2. 划分磁盘空间，因为我的电脑有两块盘（C盘为固态，D盘为机械），所以采用如图 2 所示的分区方式，在C盘划分的空间作为EFI引导区（开机时可以选择进入哪个系统），在D盘分配256G的空间。
3. 从U盘启动安装系统，自定义安装在已划分好的空间（freespace）/home, /boot 等分区可以按需选择（这里我只分了swap交换区和根目录），并选择C盘的那部分空间作为引导，安装系统。



图 1: 制作系统盘

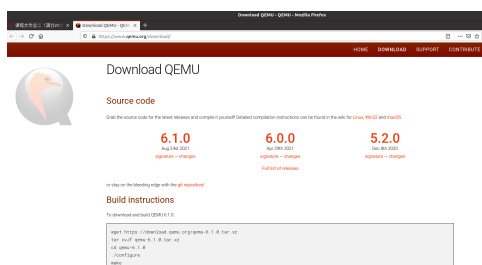
磁盘 0 基本 476.92 GB 联机	260 MB 状态良好 (E)	Windows (C-) 475.50 GB NTFS 状态良好 (启动, 页面文件, 故障转	200 MB 状态良好 (E)	1000 MB 状态良好 (恢复)
磁盘 1 基本 931.50 GB 联机	260 MB 状态良好 (E)	FILE (D-) 674.26 GB NTFS 状态良好 (基本数据分)	15.26 GB 状态良好 (主分)	240.74 GB 状态良好 (主分区)
				1000 MB 状态良好 (

图 2: 划分磁盘空间

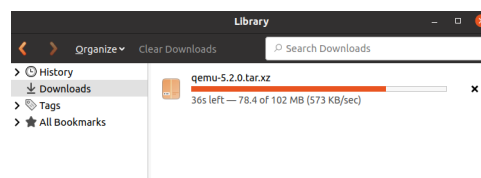
3 下载并编译安装QEMU

首先通过官网下载所需版本的 QEMU (<https://download.qemu.org/qemu-5.2.0.tar.xz>) 并通过命令行解压。

```
1 $ xz -d qemu-5.2.0.tar.xz
2 $ tar -xvf qemu-5.2.0.tar
```



(a) pic1.



(b) pic2.

图 3: Download

3.1 安装一些工具及依赖包

鉴于 ubuntu 系统是新安装的, 很多软件等都没有安装, 下面首先安装本次作业内容所需要的。

```
1 $ sudo apt-get update
2 $ sudo apt-get install gcc
3 $ sudo apt-get install make
4 $ sudo apt-get install python3
```

在尝试配置、编译 QEMU 时, 会报告一些错误, 针对这些错误安装相应的内容:

ERROR: Cannot find Ninja

```
1 $ sudo apt-get install ninja-build
2 $ sudo apt-get install build-essential
```

ERROR: glib-2.48 gthread-2.0 is required to compile QEMU

```
1 $ apt-cache search glib2
2 $ sudo apt-get install libglib2.0
```

Unable to locate package libpixman-1-0-dev

```
1 $ sudo apt-get install libpixman-1-dev
```

此时编译安装 qemu 安装虚拟机后会出现提示: VNC Server running on 127.0.0.1:5900...

这是由缺少SDL库或者在配置时没有使能SDL所导致的默认的vnc viewer无法形成可视化界面, 所以这里我们预先安装:

```
1 $ sudo apt-get install libsdl2-dev
```

3.2 编译安装QEMU

首先使用 ./configure 配置编译选项, 用于生成 Makefile, 可以通过命令 `./configure --help` 查看可用选项。注意到编译时可以指定 QEMU 支持的架构, 其余的可以不编译, 这样可以减少编译时间。

```
1 $ ./configure --enable-kvm --enable-debug --enable-vnc --enable-werror --enable-sdl --target-list="x86_64-softmmu"
```

这里指定了x86_64架构, 同时显式地使其支持 KVM 和 vnc, 编译时将 warning 当做 error 处理, 以及开启 SDL 支持。

配置完成后, 在输出信息中确认已经有 SDL 支持。下面直接编译安装即可。

```
1 $ make -j8
2 $ make install
```

4 创建虚拟机并配置网络

4.1 检查KVM是否可用

可以采用如下命令检查:

```
1 $ grep -E 'vmx|svm' /proc/cpuinfo
2 $ lsmod | grep kvm
```

有输出即可。比如第二个命令的输出类似于:

```
1 kvm_intel          *****    0
2 kvm                *****    1   kvm_intel
```

4.2 创建虚拟机并安装操作系统

首先创建虚拟机镜像文件:

```
1 $ qemu-img create -f qcow2 ubuntu.img 30G
2 $ qemu-img create -f qcow2 ubuntu3.img 30G
```

然后启动虚拟机并安装操作系统, 为加以区分, 虚拟机的操作系统使用 ubuntu-20.04.3, 同样从官网下载镜像(<https://ubuntu.com/download/desktop>)

```
1 $ qemu-system-x86_64 -m 4096 ubuntu.img -enable-kvm -cdrom ./ubuntu-20.04.3-desktop-amd64.iso
2 $ qemu-system-x86_64 -m 4096 ubuntu3.img -enable-kvm -cdrom ./ubuntu-20.04.3-desktop-amd64.iso
```

安装虚拟机过程如图 4所示:

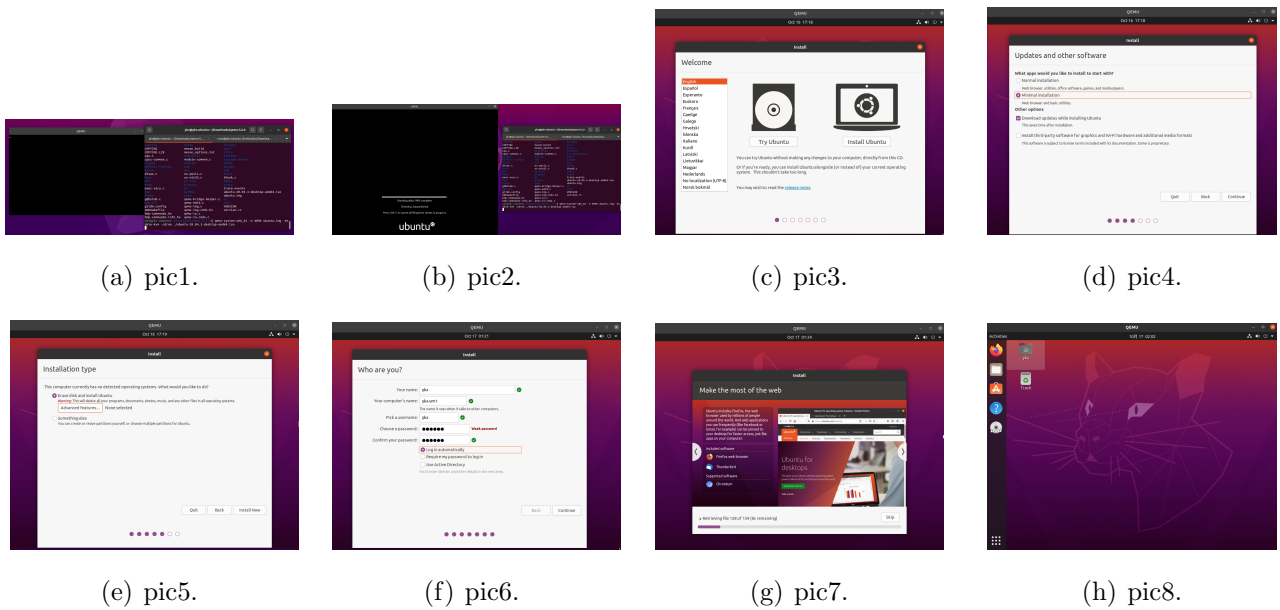


图 4: 虚拟机安装过程

4.3 配置tap网络

学习并了解了相关知识(<https://wiki.qemu.org/Documentation/Networking>), tap 网络模式是在主机中创建网桥和tap设备, 然后使网桥连接主机的网卡和tap设备, 再通过指定的tap设备和虚拟机的虚拟网卡相连接, 达到访问外部网络的目的 (tap设备是在数据链路层模拟的) 这种连接模式会为虚拟机分配一个独立的局域网IP地址。

这里会用到一些网络相关的工具, 如下安装:

```
1 $ sudo apt-get install net-tools
2 $ sudo apt-get install ethtool
```

使用 `ifconfig` 查看网络设备如5(a)所示

```

yxx@yxx-ubuntu:~/Desktop$ ifconfig
enp4s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.131 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::218a:5851:7d19:2902 prefixlen 64 scopeid 0x20<link>
    ether 00:2b:67:9f:59:58 txqueuelen 1000 (Ethernet)
    RX packets 61 bytes 9758 (9.7 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 92 bytes 10771 (10.7 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 2502 bytes 252440 (252.4 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2502 bytes 252440 (252.4 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 192.168.122.1 netmask 255.255.255.0 broadcast 192.168.122.255
    ether 52:54:00:c1:8e:99 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlp5s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.129 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::aefc:46e2:e15e:988d prefixlen 64 scopeid 0x20<link>
    ether f8:c6:5e:9d:5d:a9 txqueuelen 1000 (Ethernet)
    RX packets 4053 bytes 2049177 (2.0 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4361 bytes 853373 (853.3 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@yxx-ubuntu:/home/yxx/downloads/qemu-2.2.0# ifconfig
br0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.131 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::22b:67ff:fe9f:5958 prefixlen 64 scopeid 0x20<link>
    ether 00:2b:67:9f:59:58 txqueuelen 1000 (Ethernet)
    RX packets 90 bytes 10710 (10.7 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 131 bytes 10720 (10.7 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

enp4s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.131 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::218a:5851:7d19:2902 prefixlen 64 scopeid 0x20<link>
    ether 00:2b:67:9f:59:58 txqueuelen 1000 (Ethernet)
    RX packets 230 bytes 40691 (40.6 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 806 bytes 112229 (112.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 2766 bytes 282190 (282.1 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2766 bytes 282190 (282.1 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

tap0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether 72:34:5a:8d:50:44 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

tapi: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether f2:f9:bb:90:3b:69 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

(a) pic1.

(b) pic2.

图 5: 查看主机网络设备

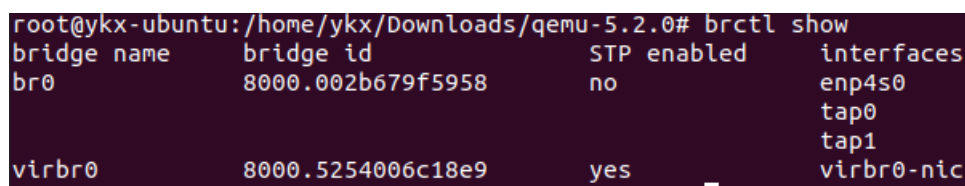
下面在主机创建网桥和两个tap设备，先安装所需工具

```
1 $ sudo apt-get install bridge-utils
2 $ sudo apt-get install iproute2
```

然后创建网桥、tap设备、网桥分别与主机有线网卡和tap设备相连、并为网桥分配 IP

```
1 $ modprobe tun tap #make sure module has built-in
2 $ ip link add br0 type bridge
3 $ ip tuntap add dev tap0 mode tap
4 $ ip tuntap add dev tap1 mode tap
5 $ ip link set dev enp4s0 master br0
6 $ ip link set dev tap0 master br0
7 $ ip link set dev tap1 master br0
8 $ ip link set dev br0 up
9 $ dhclient br0
10 $ ifconfig tap0 up
11 $ ifconfig tap1 up
```

再次通过 `ifconfig` 查看网络设备如5(b)所示，通过命令 `brctl show` 查看网桥连接状态：



```
root@yqx-ubuntu:/home/yqx/Downloads/qemu-5.2.0# brctl show
bridge name      bridge id        STP enabled      interfaces
br0               8000.002b679f5958  no               enp4s0
                  tap0
                  tap1
virbr0            8000.5254006c18e9  yes              virbr0-nic
```

图 6: 网桥连接情况

4.4 启动虚拟机并配置网络

执行如下命令启动虚拟机：

```
1 $ qemu-system-x86_64 -m 4096 ubuntu.img -netdev tap,id=mynet0,ifname=tap0,
   script=no,downscript=no -device e1000,netdev=mynet0,mac=52:55:00:d1
   :55:01 -enable-kvm
2 $ qemu-system-x86_64 -m 4096 ubuntu3.img -netdev tap,id=mynet0,ifname=tap1
   ,script=no,downscript=no -device virtio-net-pci,netdev=mynet0,mac
   =52:55:00:d1:55:02 -enable-kvm
```

主机的机器名字为 `yqx-ubuntu`，两个虚拟机的名字分别为 `yqx-vm1`, `yqx-vm3`，用来区分。

在两个虚拟机中分别安装工具 `net-tools` 和 `ethtool`，分别查看网络设备和网卡信息，如图7(a)、7(b)、7(c)、7(d)所示。

```
ykx@ykn-vm1:~$ ifconfig
ens3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.134 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::50d7:9a25:8bf6:3a60 prefixlen 64 scopeid 0x20<link>
    ether 52:55:00:d1:55:01 txqueuelen 1000 (Ethernet)
    RX packets 127 bytes 54873 (54.8 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 212 bytes 24327 (24.3 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 179 bytes 15843 (15.8 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 179 bytes 15843 (15.8 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

(a) vm1

```
ykx@ykn-vm3:~$ ifconfig
ens3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.135 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::b75b:a84b:9116:9813 prefixlen 64 scopeid 0x20<link>
    ether 52:55:00:d1:55:02 txqueuelen 1000 (Ethernet)
    RX packets 107 bytes 32004 (32.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 198 bytes 20379 (20.3 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 183 bytes 15477 (15.4 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 183 bytes 15477 (15.4 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

(b) vm3

```
ykx@ykn-vm1:~$ ethtool -i ens3
driver: e1000
version: 5.11.0-37-generic
firmware-version:
expansion-rom-version:
bus-info: 0000:00:03.0
supports-statistics: yes
supports-test: yes
supports-eeprom-access: yes
supports-register-dump: yes
supports-priv-flags: no
```

(c) vm1-e1000

```
ykx@ykn-vm3:~$ ethtool -i ens3
driver: virtio_net
version: 1.0.0
firmware-version:
expansion-rom-version:
bus-info: 0000:00:03.0
supports-statistics: yes
supports-test: no
supports-eeprom-access: no
supports-register-dump: no
supports-priv-flags: no
```

(d) vm3-virtio

图 7: 查看虚拟机网络设备和网卡信息

4.5 通过ssh连接虚拟机或者主机

分别在主机和两个虚拟机中安装 ssh，下图展示主机连接 ykx-vm3 和 ykx-vm3 连接主机。

```
ykx@ykn-ubuntu:~/Downloads/cent-7.2-0$ ssh ykx@192.168.1.135
The authenticity of host '192.168.1.135 (192.168.1.135)' can't be established.
ECDSA key fingerprint is SHA256:HHST5d08530h106ABcb2pMnEsnQ0WuQyX3qj8YBHjPY.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.135' (ECDSA) to the list of known hosts.
ykx@192.168.1.135's password:
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.11.0-38-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

93 updates can be applied immediately.
36 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
ykx@ykn-vm3:~$ ls
```

(a) 主机连接虚拟机

```
ykx@ykn-vm3:~$ ssh ykx@192.168.1.131
The authenticity of host '192.168.1.131 (192.168.1.131)' can't be established.
ECDSA key fingerprint is SHA256:P9p+0aDRcGQjMDwEcByd1nLodVpIhEC+wk0h0LAaso.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.131' (ECDSA) to the list of known hosts.
ykx@192.168.1.131's password:
Welcome to Ubuntu 20.10 (GNU/Linux 5.8.0-25-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

283 updates can be installed immediately.
156 of these updates are security updates.
To see these additional updates run: apt list --upgradable

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
ykx@ykn-ubuntu:~$
```

(b) 虚拟机连接主机

图 8: ssh连接

5 网络性能测试和比较

5.1 连接外网

在 subsection 4.5 中已经成功通过 ssh 从主机连接虚拟机以及从虚拟机连接主机。再测试虚拟机是否可以访问外网


```

jkykx-vm1:~$ ping baidu.com
PING baidu.com (220.181.38.251) 56(84) bytes of data:
64 bytes from 220.181.38.251: icmp_seq=1 ttl=43 time=28.2 ms
64 bytes from 220.181.38.251: icmp_seq=2 ttl=43 time=29.2 ms
64 bytes from 220.181.38.251: icmp_seq=3 ttl=43 time=29.0 ms
64 bytes from 220.181.38.251: icmp_seq=4 ttl=43 time=36.7 ms
64 bytes from 220.181.38.251: icmp_seq=5 ttl=43 time=29.2 ms

```

(a) vm1

```

jkykx-vm3:~$ ping baidu.com
PING baidu.com (220.181.38.251) 56(84) bytes of data:
64 bytes from 220.181.38.251: icmp_seq=1 ttl=43 time=28.6 ms
64 bytes from 220.181.38.251: icmp_seq=2 ttl=43 time=28.5 ms
64 bytes from 220.181.38.251: icmp_seq=3 ttl=43 time=29.0 ms
64 bytes from 220.181.38.251: icmp_seq=4 ttl=43 time=28.8 ms
64 bytes from 220.181.38.251: icmp_seq=5 ttl=43 time=28.6 ms

```

(b) vm3

图 9: 虚拟机连接外网

5.2 测试虚拟机到主机的网络性能，并进行比较

因为虚拟机访问外网都是通过主机的网卡，所以为比较两种虚拟网卡设备的性能，我们分别测试两个虚拟机到主机的速度来比较。这里使用 iperf 工具来测试网络性能，在主机和虚拟机中分别安装 iperf 工具，(`sudo apt-get install iperf3`)。使用方法参考 <https://www.cnblogs.com/saneri/p/11169926.html>

首先以主机作为服务端监听一个窗口：

```
1 $ iperf3 -s -p 8080
```

然后分别在两个虚拟机上测试与主机的连接：

```
1 $ iperf3 -c 192.168.1.131 -p 8080 -i 1
```

测试结果如下图所示：

```

jkykx-vm1:~$ iperf3 -c 192.168.1.131 -p 8080 -i 1
Connecting to host 192.168.1.131 port 8080
[ ID] Interval      Transfer      Bitrate      Retr  Cwnd
[ 0] 0.00-1.00 sec  154 Mbytes    1.29 Gbits/sec  0    1.71 Mbytes
[ 1] 1.00-2.00 sec  156 Mbytes    1.31 Gbits/sec  0    1.92 Mbytes
[ 2] 2.00-3.00 sec  155 Mbytes    1.30 Gbits/sec  0    2.01 Mbytes
[ 3] 3.00-4.00 sec  152 Mbytes    1.28 Gbits/sec  0    2.01 Mbytes
[ 4] 4.00-5.00 sec  150 Mbytes    1.25 Gbits/sec  0    2.01 Mbytes
[ 5] 5.00-6.00 sec  151 Mbytes    1.27 Gbits/sec  0    2.01 Mbytes
[ 6] 6.00-7.00 sec  149 Mbytes    1.25 Gbits/sec  0    2.01 Mbytes
[ 7] 7.00-8.00 sec  149 Mbytes    1.25 Gbits/sec  0    2.01 Mbytes
[ 8] 8.00-9.00 sec  150 Mbytes    1.25 Gbits/sec  0    2.01 Mbytes
[ 9] 9.00-10.00 sec 152 Mbytes    1.28 Gbits/sec  0    2.10 Mbytes
[ ID] Interval      Transfer      Bitrate      Retr  sender receiver
[ 0] 0.00-10.00 sec 1.48 Gbytes    1.27 Gbits/sec  0      sender receiver
iperf Done.

```

(a) vm1

```

jkykx-vm3:~$ iperf3 -c 192.168.1.131 -p 8080 -i 1
Connecting to host 192.168.1.131 port 8080
[ ID] Interval      Transfer      Bitrate      Retr  Cwnd
[ 0] 0.00-1.00 sec  3.09 Gbytes    26.5 Gbits/sec  0    3.00 Mbytes
[ 1] 1.00-2.00 sec  2.78 Gbytes    23.7 Gbits/sec  0    3.00 Mbytes
[ 2] 2.00-3.00 sec  2.76 Gbytes    23.7 Gbits/sec  0    3.00 Mbytes
[ 3] 3.00-4.00 sec  2.48 Gbytes    20.8 Gbits/sec  0    3.00 Mbytes
[ 4] 4.00-5.00 sec  2.71 Gbytes    23.3 Gbits/sec  0    3.00 Mbytes
[ 5] 5.00-6.00 sec  2.61 Gbytes    22.4 Gbits/sec  0    3.00 Mbytes
[ 6] 6.00-7.00 sec  2.72 Gbytes    23.4 Gbits/sec  0    3.00 Mbytes
[ 7] 7.00-8.00 sec  2.77 Gbytes    23.8 Gbits/sec  0    3.00 Mbytes
[ 8] 8.00-9.00 sec  2.57 Gbytes    22.1 Gbits/sec  0    3.00 Mbytes
[ 9] 9.00-10.00 sec 2.39 Gbytes    20.5 Gbits/sec  0    3.00 Mbytes
[ ID] Interval      Transfer      Bitrate      Retr  sender receiver
[ 0] 0.00-10.00 sec 26.8 Gbytes    23.0 Gbits/sec  0      sender receiver
iperf Done.

```

(b) vm3

```

jkykx-server:~$ iperf3 -s -p 8080
Server listening on 8080
Accepted connection from 192.168.1.131 port 40214
[ ID] Interval      Transfer      Bitrate
[ 0] 0.00-1.00 sec  152 Mbytes    1.27 Gbits/sec
[ 1] 1.00-2.00 sec  158 Mbytes    1.33 Gbits/sec
[ 2] 2.00-3.00 sec  153 Mbytes    1.30 Gbits/sec
[ 3] 3.00-4.00 sec  152 Mbytes    1.28 Gbits/sec
[ 4] 4.00-5.00 sec  149 Mbytes    1.25 Gbits/sec
[ 5] 5.00-6.00 sec  152 Mbytes    1.28 Gbits/sec
[ 6] 6.00-7.00 sec  149 Mbytes    1.25 Gbits/sec
[ 7] 7.00-8.00 sec  148 Mbytes    1.25 Gbits/sec
[ 8] 8.00-9.00 sec  149 Mbytes    1.25 Gbits/sec
[ 9] 9.00-10.00 sec 153 Mbytes    1.28 Gbits/sec
[ ID] Interval      Transfer      Bitrate
[ 0] 0.00-10.01 sec 1.48 Gbytes    1.27 Gbits/sec
Server listening on 8080
Accepted connection from 192.168.1.135 port 60234
[ ID] Interval      Transfer      Bitrate
[ 0] 0.00-1.00 sec  3.09 Gbytes    26.5 Gbits/sec
[ 1] 1.00-2.00 sec  2.78 Gbytes    23.7 Gbits/sec
[ 2] 2.00-3.00 sec  2.76 Gbytes    23.7 Gbits/sec
[ 3] 3.00-4.00 sec  2.48 Gbytes    20.8 Gbits/sec
[ 4] 4.00-5.00 sec  2.71 Gbytes    23.3 Gbits/sec
[ 5] 5.00-6.00 sec  2.61 Gbytes    22.4 Gbits/sec
[ 6] 6.00-7.00 sec  2.72 Gbytes    23.4 Gbits/sec
[ 7] 7.00-8.00 sec  2.77 Gbytes    23.8 Gbits/sec
[ 8] 8.00-9.00 sec  2.57 Gbytes    22.1 Gbits/sec
[ 9] 9.00-10.00 sec 2.39 Gbytes    20.5 Gbits/sec
[ ID] Interval      Transfer      Bitrate
[ 0] 0.00-10.00 sec 26.8 Gbytes    23.0 Gbits/sec

```

(c) server

图 10: 比较网络性能

总结如表 1:

表 1: 比较结果

	Bitrate
e1000	1.27 Gbits/sec
virtio-net	23.0 Gbits/sec

即 virtio-net 的性能远好于 e1000

6 补充

6.1 关于与无线网卡的网桥连接

参考 <https://wiki.qemu.org/Documentation/Networking/NAT>, 在这部分内容中介绍了建立网桥连接无向网卡的方法, 这里提供了一个脚本来代替命令行参数设定来启动虚拟机。

6.2 virt-manager尝试

virt-manager 是一个虚拟机管理器, 提供了图形化界面来创建和管理虚拟机, 通过它可以便捷地创建虚拟机。但是在使用的时候, 它会默认使用 qemu-system-i386 的模拟器来创建虚拟机, 这个默认设置需要通过在软件内改写 XML 来设置。另一方面, 在用其创建虚拟机时对网络的设置时, 如果想要使用在命令行创建的网桥, 也要修改 XML 来设置, 但在我的尝试下, 它无法识别我在主机创建的tap设备。

7 心得体会

在这项大作业中, 经历过诸多试错过程, 最终完成了全部的内容。在此期间了解了更多的网络设备虚拟化的知识和 qemu 的使用, 对 linux 系统的使用更加得心应手 (双系统重装了一次)。另一方面, 在配置虚拟机的网络时, 对虚拟机连接网络的各种方式都作以了解和学习, 同时联系计算机网络课程中学到的知识, 对这项作业帮助很多。