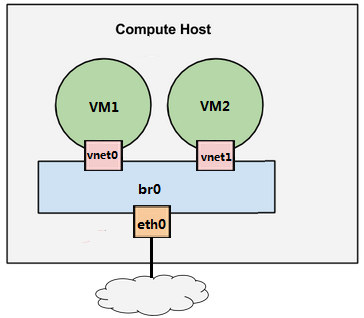


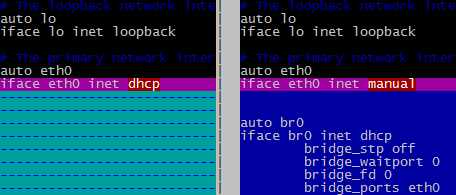
本节将演示如何在实验环境中实现下图所示的虚拟网络



**配置 Linux Bridge br0**

编辑 /etc/network/interfaces，配置 br0。

下面用 vimdiff 展示了对 /etc/network/interfaces 的修改



有两点需要注意：

1. 之前宿主机的 IP 是通过 dhcp 配置在 eth0 上的；创建 Linux Bridge 之后，IP 就必须放到 br0 上了

2. 在 br0 的配置信息中请注意最后一行 “**bridge\_ports eth0**”，其作用就是将 eth0 挂到 br0 上

重启宿主机，查看 IP 配置，可以看到 IP 已经放到 br0 上了

# ifconfig

br0       Link encap:Ethernet  HWaddr 00:0c:29:8d:ec:be

         inet addr:192.168.111.107  Bcast:192.168.111.255  Mask:255.255.255.0

         inet6 addr: fe80::20c:29ff:fe8d:ecbe/64 Scope:Link

         UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

         RX packets:22573 errors:0 dropped:0 overruns:0 frame:0

         TX packets:2757 errors:0 dropped:0 overruns:0 carrier:0

         collisions:0 txqueuelen:0

         RX bytes:4927580 (4.9 MB)  TX bytes:368895 (368.8 KB)

eth0      Link encap:Ethernet  HWaddr 00:0c:29:8d:ec:be

         inet6 addr: fe80::20c:29ff:fe8d:ecbe/64 Scope:Link

         UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

         RX packets:24388 errors:0 dropped:0 overruns:0 frame:0

         TX packets:2816 errors:0 dropped:0 overruns:0 carrier:0

         collisions:0 txqueuelen:1000

         RX bytes:5590438 (5.5 MB)  TX bytes:411558 (411.5 KB)

lo        Link encap:Local Loopback

         inet addr:127.0.0.1  Mask:255.0.0.0

         inet6 addr: ::1/128 Scope:Host

         UP LOOPBACK RUNNING  MTU:65536  Metric:1

         RX packets:146 errors:0 dropped:0 overruns:0 frame:0

         TX packets:146 errors:0 dropped:0 overruns:0 carrier:0

         collisions:0 txqueuelen:0

         RX bytes:10521 (10.5 KB)  TX bytes:10521 (10.5 KB)

virbr0    Link encap:Ethernet  HWaddr 72:db:fb:f2:19:91

         inet addr:192.168.122.1  Bcast:192.168.122.255  Mask:255.255.255.0

         UP BROADCAST MULTICAST  MTU:1500  Metric:1

         RX packets:0 errors:0 dropped:0 overruns:0 frame:0

         TX packets:0 errors:0 dropped:0 overruns:0 carrier:0

         collisions:0 txqueuelen:0

         RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

用 brctl show 查看当前 Linux Bridge 的配置。

eth0 已经挂到 br0 上了

# brctl show

bridge name     bridge id               STP enabled     interfaces

br0             8000.000c298decbe       no                    eth0

virbr0          8000.000000000000       yes

除了 br0，大家应该注意到还有一个 virbr0 的 Bridge，而且 virbr0 上已经配置了 IP 地址 192.168.122.1。

virbr0 的作用我们会在后面介绍。

在宿主机中 CloudMan 已经提前创建好了虚机 VM1 和 VM2，现在都处于关机状态

# virsh list --all

Id    Name                           State

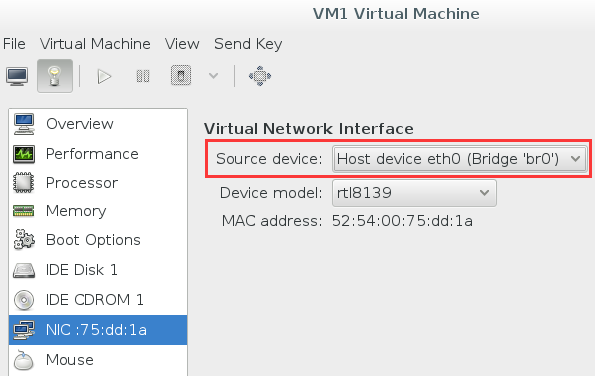
----------------------------------------------------

-     VM1                            shut off

-     VM2                            shut off

**配置 VM1**

下面我们在 virt-manager 中查看一下 VM1 的网卡配置（为了使大家能够熟练使用命令行工具 virsh 和图形工具 virt-manager，CloudMan 在演示的时候会同时用到它们，两个工具都很重要）



可以看到虚拟网卡的 source device 我们选择的是 br0

下面我们启动 VM1，看会发生什么

# virsh start VM1

Domain VM1 started

# brctl show

bridge name     bridge id               STP enabled     interfaces

br0             8000.000c298decbe       no                    eth0

                                                                                 vnet0

virbr0          8000.000000000000       yes

brctl show 告诉我们 br0 下面添加了一个 vnet0 设备，通过 virsh 确认这就是VM1的虚拟网卡。

# virsh domiflist VM1

Interface  Type       Source     Model       MAC

-------------------------------------------------------

vnet0      bridge     br0        rtl8139     52:54:00:75:dd:1a

VM1 的 IP 是 DHCP 获得的（设置静态 IP 当然也可以），通过 virt-manager 控制台登录 VM1，查看 IP。

# ifconfig

eth0      Link encap:Ethernet  HWaddr 52:54:00:75:dd:1a

         inet addr:192.168.111.106  Bcast:192.168.111.255  Mask:255.255.255.0

         inet6 addr: fe80::5054:ff:fe75:dd1a/64 Scope:Link

         UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

         RX packets:400 errors:0 dropped:0 overruns:0 frame:0

         TX packets:101 errors:0 dropped:0 overruns:0 carrier:0

         collisions:0 txqueuelen:1000

         RX bytes:41950 (41.9 KB)  TX bytes:12583 (12.5 KB)

VM1 通过 DHCP 拿到的 IP 是 192.168.111.106，与宿主机（IP为192.168.111.107）是同一个网段。Ping 一下外网

root@VM1:~# ping www.baidu.com

PING www.a.shifen.com (180.97.33.108) 56(84) bytes of data.

64 bytes from 180.97.33.108: icmp\_seq=1 ttl=53 time=34.9 ms

64 bytes from 180.97.33.108: icmp\_seq=2 ttl=53 time=36.2 ms

64 bytes from 180.97.33.108: icmp\_seq=3 ttl=53 time=38.8 ms

没问题，可以访问。

另外，在 VM1 中虚拟网卡是 eth0，并不是 vnet0。

vent0 是该虚拟网卡在宿主机中对应的设备名称，其类型是 TAP 设备，这里需要注意一下。

**配置 VM2**

跟 VM1 一样，VM2 的虚拟网卡也挂在 br0上，启动 VM1，查看网卡信息

# virsh start VM2

Domain VM2 started

# brctl show

bridge name     bridge id               STP enabled     interfaces

br0             8000.000c298decbe       no                    eth0

                                                                                 vnet0

                                                                                 vnet1

virbr0          8000.000000000000       yes

br0 下面多了 vnet1，通过 virsh 确认这就是 VM2 的虚拟网卡。

# virsh domiflist VM2

Interface  Type       Source     Model       MAC

-------------------------------------------------------

vnet0      bridge     br0        rtl8139     52:54:00:cf:33:a1

VM2 通过 DHCP 拿到的 IP 是 192.168.111.108，登录 VM2，验证网络的连通性

Ping VM1

root@VM2:~# ping VM1

PING VM1 (192.168.111.106) 56(84) bytes of data.

64 bytes from 192.168.111.106: icmp\_seq=1 ttl=64 time=4.54 ms

64 bytes from 192.168.111.106: icmp\_seq=2 ttl=64 time=1.63 ms

64 bytes from 192.168.111.106: icmp\_seq=3 ttl=64 time=2.16 ms

Ping 宿主机

root@VM2:~# ping 192.168.111.107

PING 192.168.111.107 (192.168.111.107) 56(84) bytes of data.

64 bytes from 192.168.111.107: icmp\_seq=1 ttl=64 time=1.02 ms

64 bytes from 192.168.111.107: icmp\_seq=2 ttl=64 time=0.052 ms

64 bytes from 192.168.111.107: icmp\_seq=3 ttl=64 time=0.064 ms

Ping 外网

root@VM2:~# ping www.baidu.com

PING www.a.shifen.com (180.97.33.107) 56(84) bytes of data.

64 bytes from 180.97.33.107: icmp\_seq=1 ttl=53 time=53.9 ms

64 bytes from 180.97.33.107: icmp\_seq=2 ttl=53 time=45.0 ms

64 bytes from 180.97.33.107: icmp\_seq=3 ttl=53 time=44.2 ms

可见，通过 br0 这个 Linux Bridge，我们实现了 VM1、VM2、宿主机和外网这四者之间的数据通信。

下节我们讨论 virbr0