Xen网络解析

2014/5/13

renyl

# 1 Guest网桥通信方式

## 1.1 能够与外部网络通讯

Guest要和网桥通信，必须借助Tap设备与网桥建立连接，如图1-1所示。

xenbr0

eth0

Guest1

tap1

tap2

eth0

Guest2

eth0

图1-1 Guest与网桥通信方式

分析：

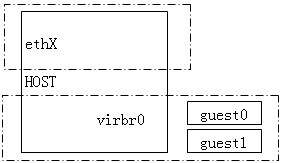
1. Tap设备是Linux kernel为用户空间提供的一种虚拟网络设备。
2. Tap设备可以理解为网桥上的一个接口，Guest的虚拟网卡eth0与这些接口相连，进而连接到网桥。
3. 同时，网桥与Host的物理网卡eth0相连，保证了Guest与外部网络的正常通信。

注：使用Tap设备的原因：

Guest的虚拟网卡对于Host来说是不可见的。也就是说Host上看不见Guest里面的eth0。所以Host的网桥必须借助一种机制，来区分连接在网桥上的各个Guest，这就是Tap设备。

## 1.2 不能够与外部网络通讯

使用virbr0作为guest与host通信的网桥，网络拓扑如下图所示：



# 2 Xen网络连接原理

Xen将Host也看作一个Guest，称为Dom0。真正的Guest称为DomU，如Dom1，Dom2，Dom3等。因此Host（Dom0）的网络连接从逻辑结构上说，和其他的Guest是一样的，如图2-1所示。

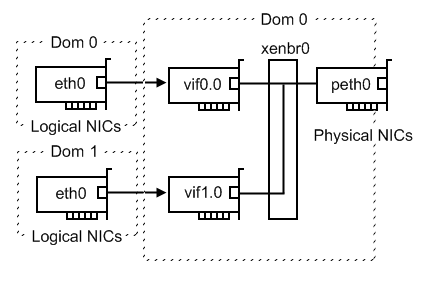


图2-1 Xen网络连接原理

注：

vif0.0 和vif1.0为tap设备的命名。vifx.x与guest的网卡对应规则：vif后的第一个数字为guest的Dom编号，第二个数字为guest中的网卡编号，如vif1.0对应Dom1上的eth0。

分析：

由上图可知，Host作为Dom0和其他的Guest一样，需要使用虚拟网卡eth0，经过tap设备vif0.0与网桥通信。Xen系统启动时，会创建以下两个虚拟设备，Dom0虚拟网卡的实现如下：

1）创建veth0，用于模拟Dom0中的eth0

2）创建Tap设备vif0.0，它是Dom0的eth0接入网桥所必须的Tap设备

Xen系统启动时通过/etc/xen/scripts/network-bridge脚本自动进行网络配置的流程如下：

1. 创建网桥xenbr0；

2） 关闭host的物理网卡eth0，将eth0的IP和MAC地址复制到veth0中（执行命令：ip addr add IPADDR dev veth0; ip link set veth0 addr MACADDR）；

3）将host的物理网卡eth0重命名为peth0（执行命令：ip link set eth0 name peth0），并且修改peth0的mac地址为FE:FF:FF:FF:FF:FF

4）将创建的虚拟网卡veth0重命名为eth0(Dom0的虚拟网卡)（执行命令：ip link set veth0 name eth0）；

5）将peth0和vif0.0加入xenbr0；

6）启动所有网络接口：xenbr0，eth0，vif0.0，peth0。

因此，Xen默认的网络配置如图2-2所示。

(原来的veth0)

xenbr0

Dom0

vif0.0

eth0

(原来的eth0)

peth0

图2-2 Xen默认网络配置

这里，eth0通过vif0.0间接地接入xenbr0，Xen kernel会将发送到xenbr0的数据包自动转发给eth0，同样，从eth0发出的数据包，将会通过vif0.0转发出去。

可以使用以下命令检查网络配置：

# brctl show

bridge name bridge id STP enabled interfaces

xenbr0 8000.feffffffffff no peth0

vif0.0 #表示eth0通过vif0.0接入到xenbr0

# 3 环路问题

由于Xen kernel将Host看作特殊的Guest：Dom0，因此导致Dom0中的网卡eth0在Host上可见。如下所示：

# ifconfig

eth0 Link encap:Ethernet HWaddr 00:19:99:0E:DE:39

inet addr:133.162.10.15 Bcast:133.162.10.255 Mask:255.255.255.0

……

peth0 Link encap:Ethernet HWaddr FE:FF:FF:FF:FF:FF

inet6 addr: fe80::fcff:ffff:feff:ffff/64 Scope:Link

……

vif0.0 Link encap:Ethernet HWaddr FE:FF:FF:FF:FF:FF

inet6 addr: fe80::fcff:ffff:feff:ffff/64 Scope:Link

……

xenbr0 Link encap:Ethernet HWaddr FE:FF:FF:FF:FF:FF

UP BROADCAST RUNNING NOARP MTU:1500 Metric:1

……

但必须注意，此时的eth0是Dom0的虚拟网卡，peth0才是Host的物理网卡。这是两个不同的设备。

* + **误操作：将eth0接入网桥xenbr0**

导致本次网络故障的原因是，使用brctl命令将eth0接入了xenbr0。

# brctl addif xenbr0 eth0

# brctl show

bridge name bridge id STP enabled interfaces

xenbr0 8000.feffffffffff no peth0

vif0.0 #表示eth0间接接入xenbr0

eth0 #表示eth0直接接入xenbr0

1. 由于brctl命令以及Linux平台上的虚拟网桥是独立于Xen kernel实现的
2. 而且eth0并不是直接接入xenbr0的
3. 所以执行这一步操作时brctl命令无法检测出eth0已经与xenbr0连通
4. 进而可以第二次将eth0接入xenbr0，形成了环路。

将eth0接入xenbr0以后，网络连接如图3-1中的红色线路所示:

xenbr0

peth0

Dom0

Vif0.0

eth0

图3-1 误操作后的网络配置

当Dom0向外部网络发送数据包时，会形成图3-2所示的环路。

xenbr0

peth0

Dom0

Vif0.0

eth0

图3-2 网络环路

* + **网络阻塞的根本原因**

网络内部形成环路，造成数据流量过大，网络阻塞。