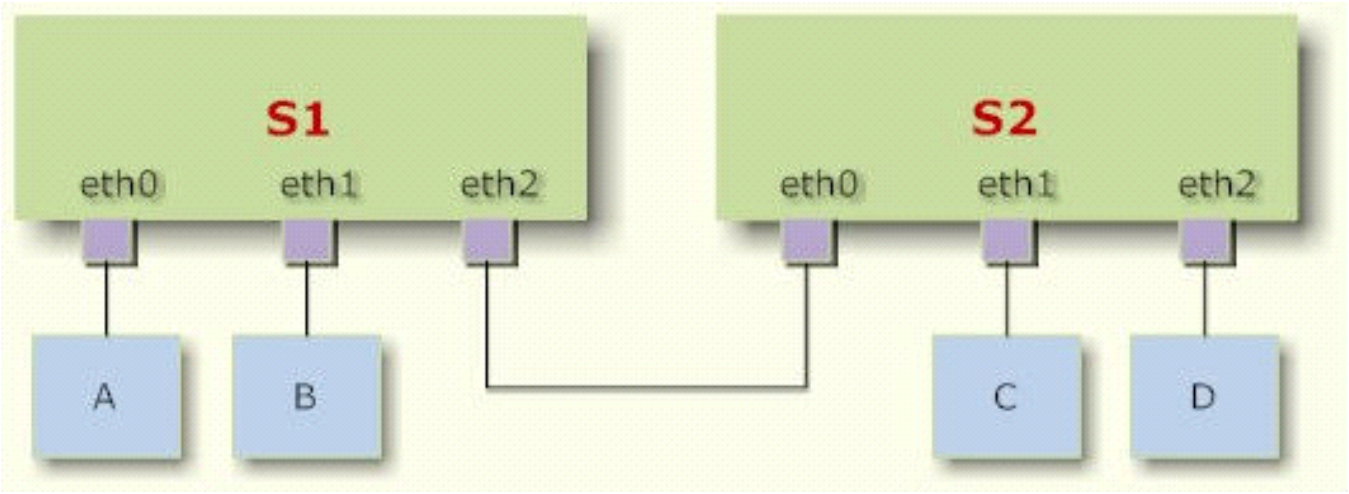
网桥

2017年5月5日 ♡菁 09:56

桥接就是把一台机器上的若干个网络接口“连接”起来。其中一个网口收到的报文会被复制给其他网口并发送出去。以使得网口之间的报文能够互相转发。

如下图所示主机A发送的报文被送到交换机S1的eth0口，由于eth0与eth1、eth2桥接在一起，故而报文被复制到eth1和eth2，并且发送出去，然后被主机B和交换机S2接收到。而S2又会将报文转发给主机C、D

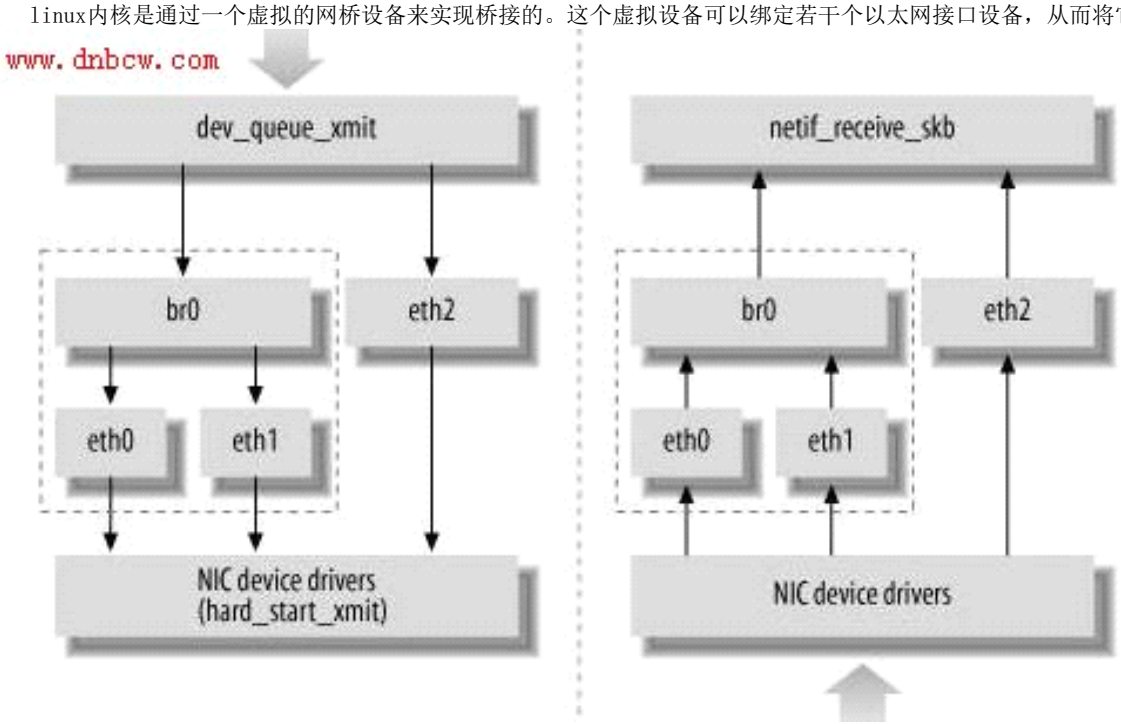


交换机在报文转发的过程中并不会篡改报文数据。交换机会关心填写在报文的数据链路层头部中的Mac地址信息（包括源地址和目的地址），以便了解每个Mac地址所代表的主机都在什么位置（与本交换机的哪个网口相连）。在报文转发时，交换机就只需要向特定的网口转发即可，从而避免不必要的网络交互。这个就是交换机的“地址学习”。但是如果交换机遇到一个自己未学习到的地址，就不会知道这个报文应该从哪个网口转发，则只好将报文转发给所有网口（接收报文的那个网口除外）。

Linux桥接

Linux内核支持网口的桥接（目前只支持以太网接口）。但是与单纯的交换机不同，交换机只是一个二层设备，对于接收到的报文，要么转发、要么丢弃。小型的交换机里面只需要一块交换芯片即可，并不需要CPU。而运行着linux内核的机器本身就是一台主机，有可能就是网络报文的目的地。其收到的报文除了转发和丢弃，还可能被送到网络协议栈的上层（网络层），从而被自己消化。

linux内核是通过一个虚拟的网桥设备来实现桥接的。这个虚拟设备可以绑定若干个以太网接口设备，从而将它们桥接起来。如下图：



网桥设备br0绑定了eth0和eth1。对于网络协议栈的上层来说，只看得到br0，因为桥接是在数据链路层实现的，上层不需要关心桥接的细节。于是协议栈上层需要发送的报文被送到br0，网桥设备的处理代码再来判断报文该被转发到eth0或是eth1apt-get，或者两者皆是；反过来，从eth0或从eth1接收到的报文被提交给网桥的处理代码，在这里会判断报文该转发、丢弃、或提交到协议栈上层。而有时候eth0、eth1也可能会作为报文的源地址或目的地址，直接参与报文的发送与接收（从而绕过网桥）。

```
[root@test network-scripts]# rm -rf ifcfg-ens33 # 删除网卡的配置文件
```

```
[root@test network-scripts]# systemctl restart network
[root@test network-scripts]# nmcli device status
```

创建网桥

```
[root@test ~]# nmcli connection add type bridge con-name br0 ifname br0
[root@test ~]# nmcli connection modify br0 ipv4.addresses 192.168.88.139/24 ipv4.gateway 192.168.88.2 ipv4.dns 114.114.114.114
ipv4.method manual
[root@test ~]# systemctl restart network
[root@test ~]# nmcli connection add type bridge-slave con-name br0-port0 ifname ens33 master br0
[root@test ~]# brctl show
[root@test ~]# systemctl restart network
[root@test ~]# ip addr

[root@test ~]# cat /etc/sysconfig/network-scripts/ifcfg-br0 # 自动创建配置文件
[root@test ~]# cat /etc/sysconfig/network-scripts/ifcfg-br0-port0
```

删除网桥

```
[root@test ~]# brctl delif eth0
[root@test ~]# ifconfig br0 down          # 只有该命令可以关闭br0，ifdown命令不可以
[root@test ~]# brctl delbr br0

[root@test ~]# systemctl restart network # 重启网络之后发现br0又出现了，这是因为bridge的配置文件依然存在
```

网桥功能测试

```
[root@test pipework]# docker run -itd --net=none busybox sh
[root@test pipework]# git clone https://github.com/jpetazzo/pipework
[root@test pipework]# ./pipework br0 24 192.168.88.111/24@192.168.88.2
```