1. 执行Mininet脚本时，报“.../scripts/disable\_offloading.sh should be set executable by ...”错误，然后脚本终止运行。
   * Linux下文件至少有三种可能的权限属性：可读、可写、可执行。脚本disable\_offloading.sh需要可执行权限，才能在终端下直接执行脚本。
   * 课程附件是以.tar.gz打包压缩格式分发的，其内部保存有各个文件的权限属性。如果在Windows下打开该压缩文件，文件的可执行权限会丢失，再移入到Linux下，就会报该错误。
   * 可以使用chmod +x ./scripts/disable\_offloading.sh给文件加上可执行权限。该问题和解决方案同样适用于./scripts/disable\_ipv6.sh。
2. 实现了broadcast\_packet函数后，使用three\_nodes\_bw.py脚本进行测试，可以ping通，但iperf不连通。

* 可以ping通说明广播网络节点实现的没问题。使用iperf不通，有两种可能：

1、iperf服务器节点没有运行；

2、节点h1-h3, b1上没有执行disable\_offloading.sh脚本。因为我们的虚拟网卡不支持Offloading，该脚本的作用就是禁止协议栈将TCP校验和计算Offload到网卡。可以在相应节点终端上手动执行该脚本。

1. 实验中的Hub、Switch不能达到预期，表现为：1）、两个iperf流的吞吐率之和达不到瓶颈带宽；2）、两个iperf流的吞吐率相差很多，大都是h2的高于h3。为什么？
2. 如果在虚拟机中运行程序，并且给虚拟机分配的CPU资源较少时，两个iperf流的（CPU、I/O）竞争导致两流吞吐率之和难以达到瓶颈带宽。要解决该问题，可以给虚拟机多分配一些资源，或者如果物理机是Linux，可以直接跑在物理机上。
3. 两个iperf流的吞吐率相差很多，本质上还是虚拟机性能较差导致的，与TCP传输控制无关。具体影响因素包括：“数据包收包处理顺序”和“事件处理所需时间”。对于前者，可以试试main.c中遍历端口时采用逆序遍历方式，会发现两个流的吞吐率大小会发生反转；对于后者，Hub/Switch收到每个包后会先申请内存，复制数据包，处理数据包，释放内存。对于这两个实验，不申请内存也能正常工作。可以删掉内存申请的相关代码，两个流的吞吐率会更接近。
4. 只要集线器/交换机处理流程实现没问题，在OJ上提交是不会出错的。如果在自己机器上总有两者不一致或者总带宽很低的现象，可以用自己的代码甚至编译好的程序在别的同学机器上试试。