高级计算机网络课程报告

课题一

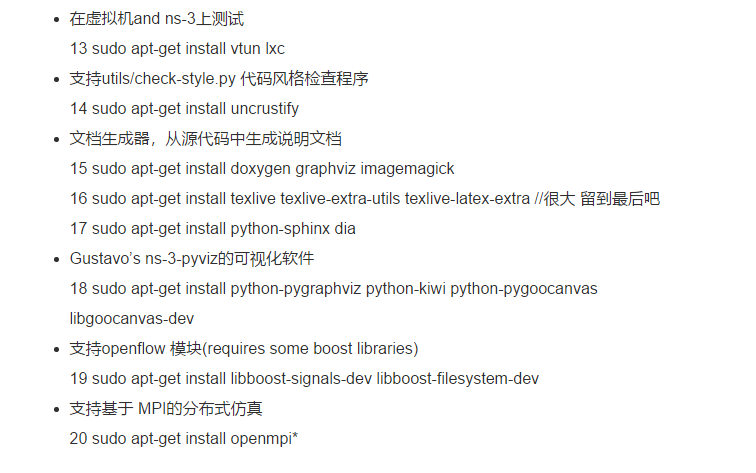
1. 实验环境

管理程序：VMWare

Linux系统：Ubuntu 16.04 内存为：4G 硬盘为：20G

1. 下载和安装ns3
   * 1. 预安装





* + 1. 安装ns3

使用tarball安装

1. mkdir workplace
2. cd worplace
3. wget [http://www.nsnam.org/release/ns-allinone-3.27.tar.bz2](http://www.nsnam.org/release/ns-allinone-3.26.tar.bz2)
4. tar xjf ns-allinone-3.27.tar.bz2
   * 1. 编译ns3
5. cd ns-allinone-3.27
6. ./build.py
7. cd ns-3.27
8. ./waf distclean
9. ./waf
10. ./waf -d optimized --enable-examples-examples --enable-tests configure

第6步执行之后可以看到以下的结果：

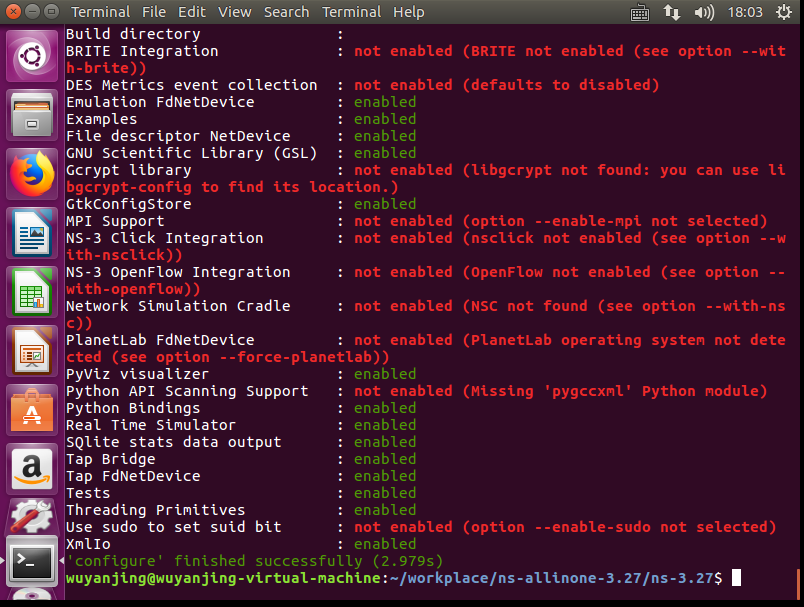


图1 Examples和Tests设置为enabled

* + 1. 测试

1. ./test.py -c core

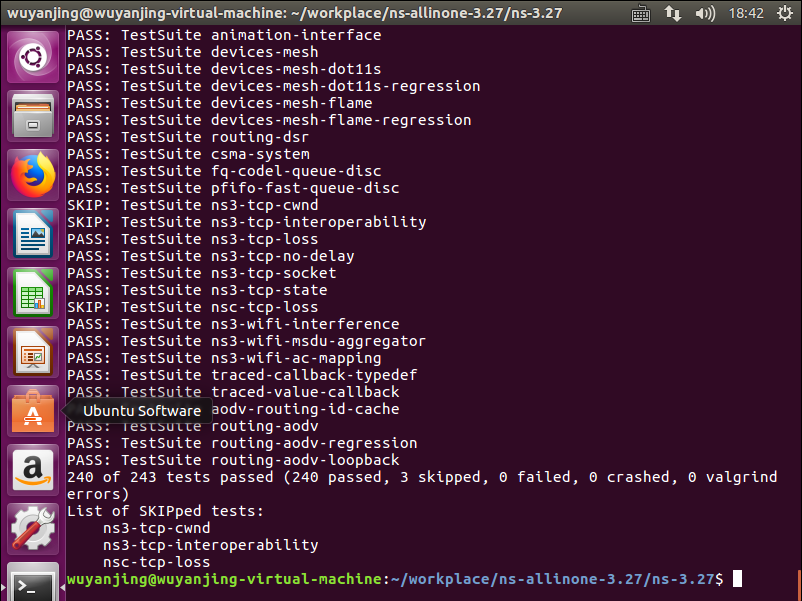


图2 测试simulator结果图

1. ./waf --run hello-simulator

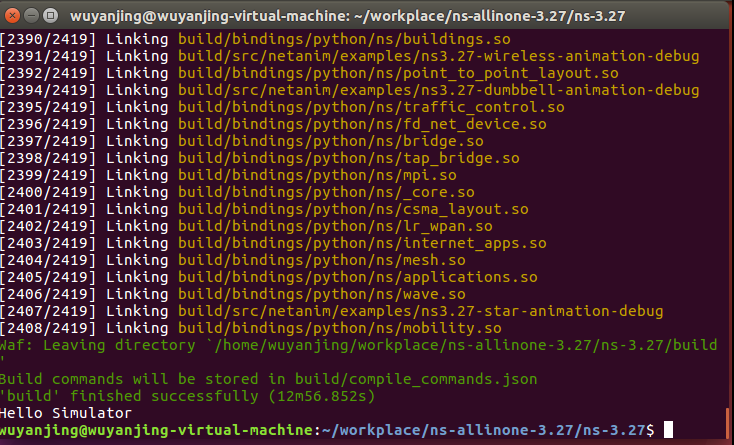


图3 hello simulator结果图

1. 下载wireshark

（1）安装安装包

1. dpkg -l | grep wireshark
2. sudo apt-get update
3. sudo apt-get install wireshark

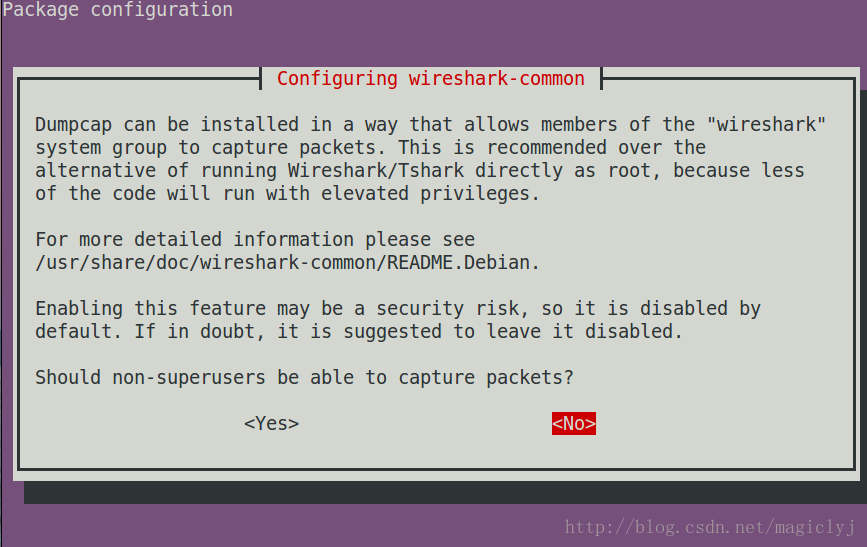
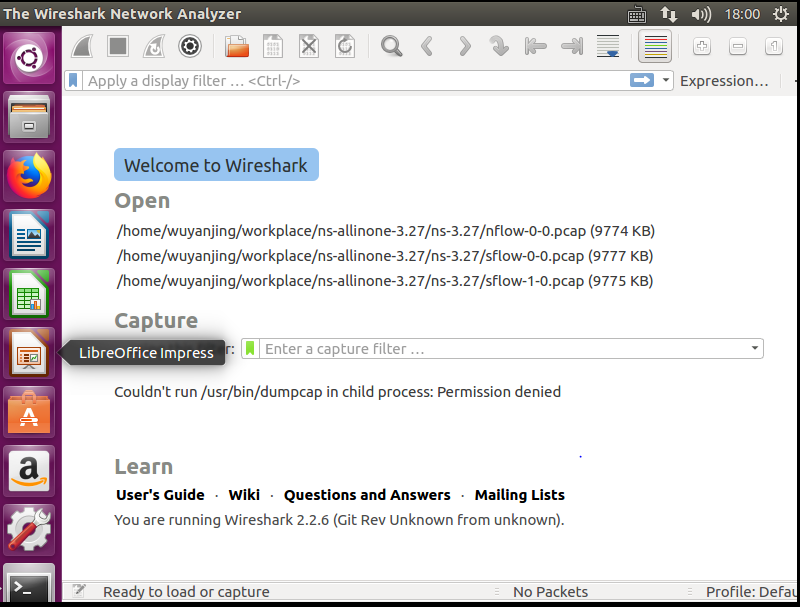


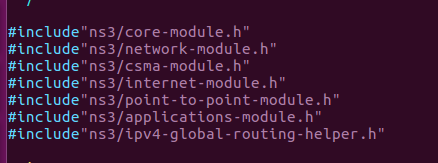
图4 是否安装wireshark确认图

执行完之后会出现以上图，选择<yes>

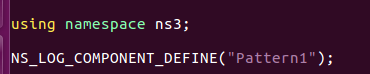
1. 修改wireshark的组权限
   1. sudo vim /etc/group
   2. 找wireshark，将当前的用户名加入
2. 测试
   1. Wireshark

图5 wireshark安装成功结果图

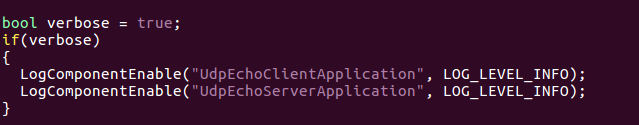
1. 使用ns3仿真datacenter network
2. 引入库函数



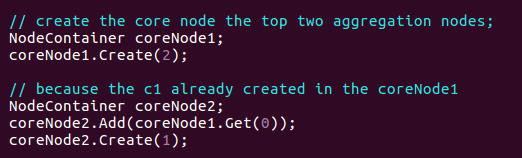
1. 使用ns3命名空间并且定义log component



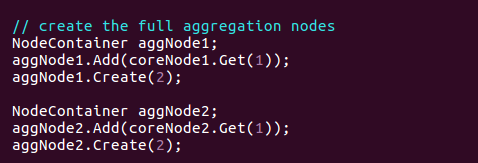
1. 主函数
   1. 实现两个log component



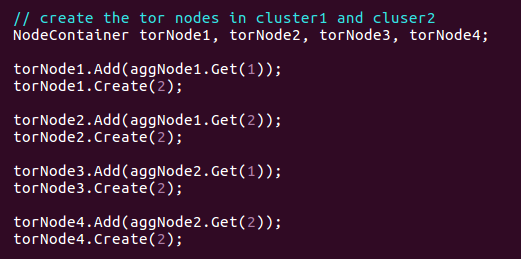
* 1. 创建c1,a1和a2节点和core与aggregation层的节点关系



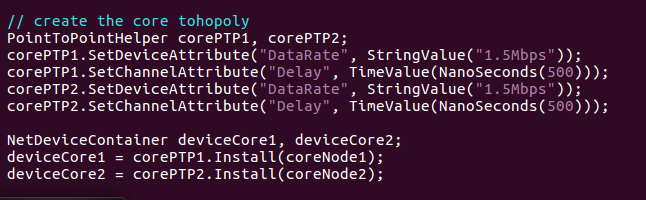
* 1. 创建t1,t2节点和aggregation层和Tor switch层的节点关系



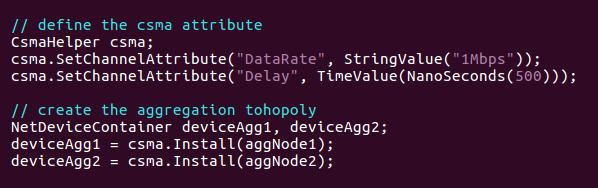
* 1. 创建n1,n2,n3,n4,n5,n6,n7,n8和Tor switch层与server层的节点关系



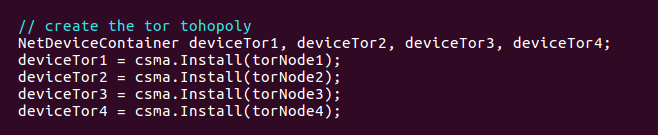
* 1. 创建core层的拓扑结构



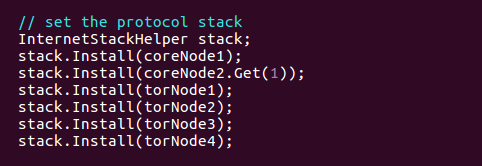
* 1. 创建aggregation层的拓扑结构



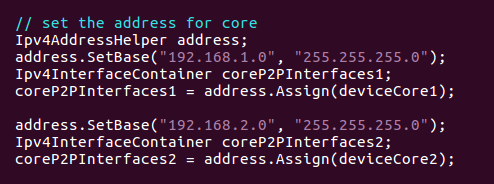
* 1. 创建Tor 层的拓扑结构



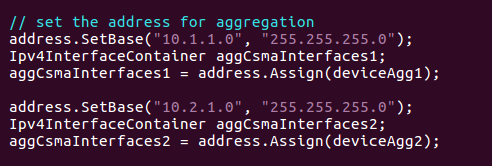
* 1. 为所有节点配置协议栈



* 1. 设置core层c1,a1,a2节点的地址



* 1. 设置aggregation层a1, t1, t2, a2, t3, t4节点的地址



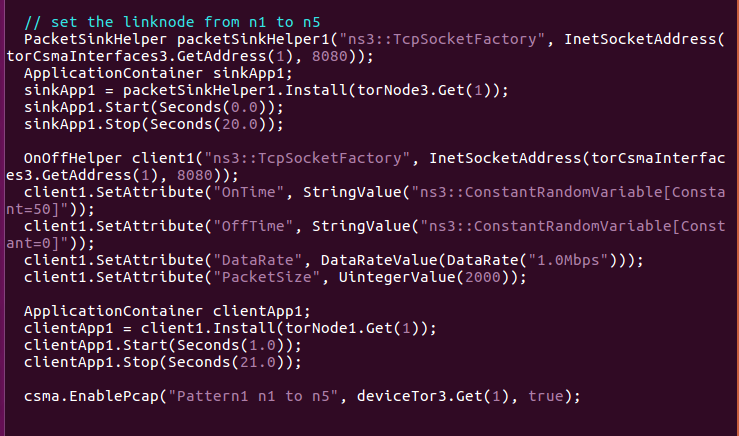
* 1. 设置Tor层t1, t2, t3, t4, n1, n2, n3, n4, n5, n6, n7, n8节点的地址



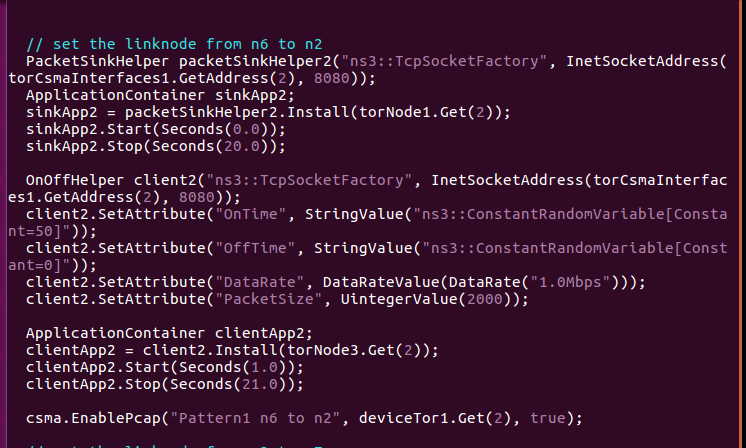
1. 测试两个traffic pattern
2. Pattern1模式

Pattern1模式实现了多到多的通信， 详细代码见pattern1.cc。我实现的通信方式是：n1-n5, n6-n2, n3-n7, n8-n4。以下是具体代码：

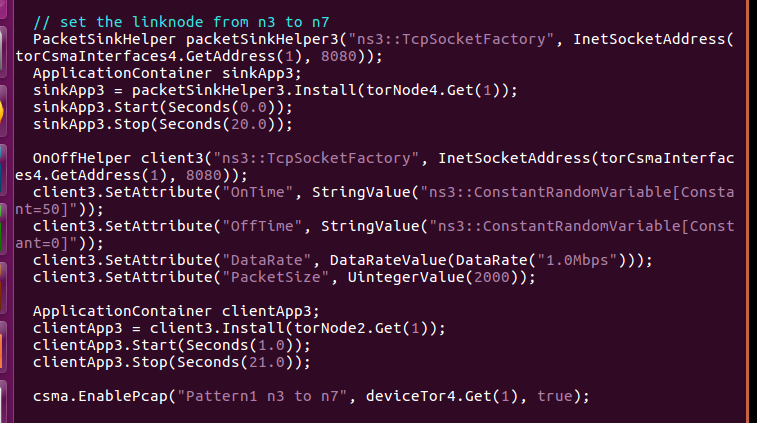
1. 网络的构建代码和四中datacenter network是一样的。
2. 为节点n1到节点n5设置链路，并记录trace



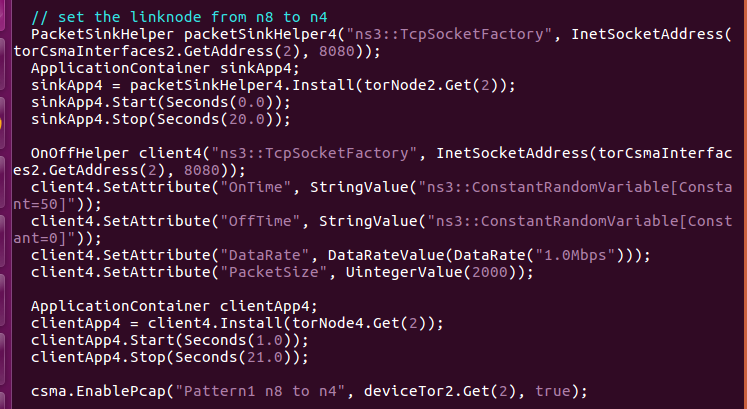
1. 为节点n6到节点n2设置链路，并记录trace



1. 为节点n3到节点n7设置链路，并记录trace



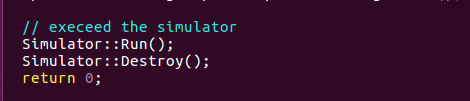
1. 为节点n8到节点n4设置链路，并记录trace



1. 记录core层PPP网络的trace



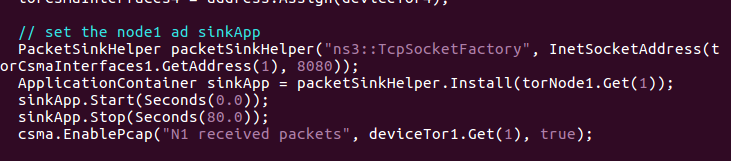
1. 进行仿真



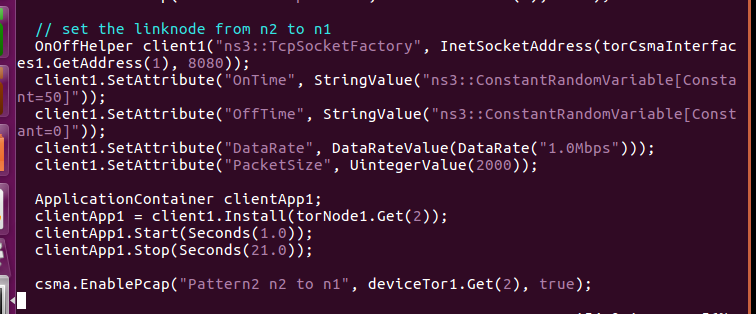
1. Pattern2模式

Pattern2模式实现了多到一的通信， 详细代码见pattern2.cc。我实现的通行方式是：n2-n1, n3-n1, n4-n1, n5-n1, n6-n1, n7-n1, n8-n1。以下是具体代码：

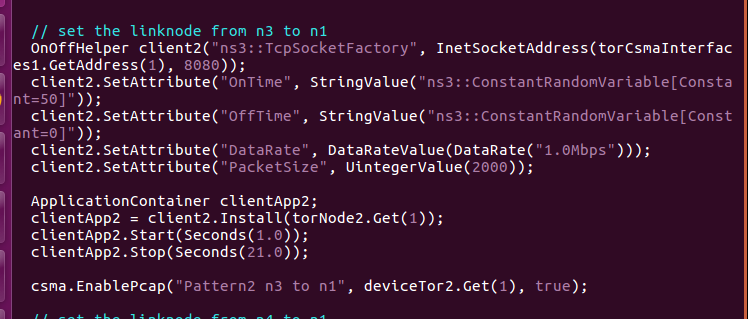
1. 构建网络结构的代码和datacenter network的代码是一样的
2. 设置节点n1的为server



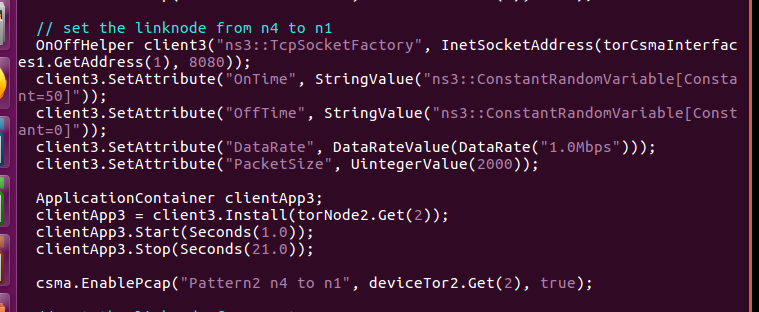
1. 设置节点n2到n1的通信链路



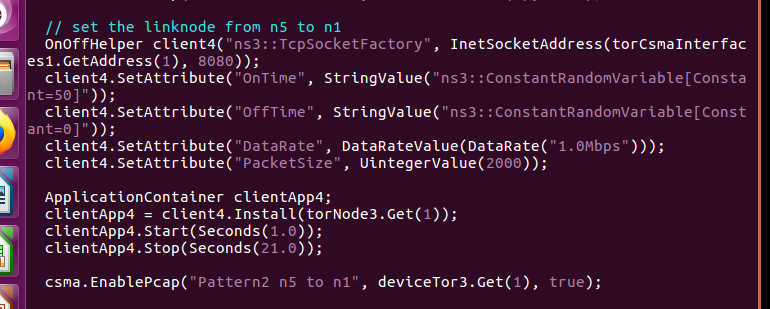
1. 设置节点n3到n1的通信链路



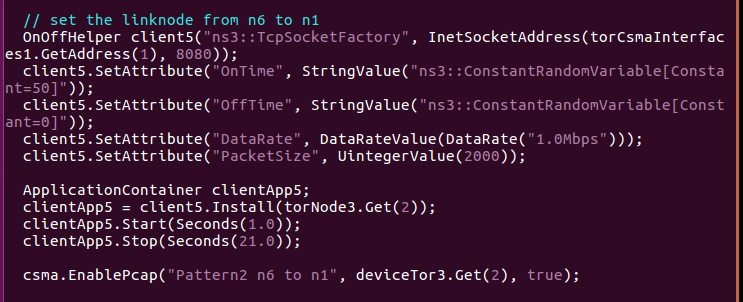
1. 设置节点n4到n1的通信链路



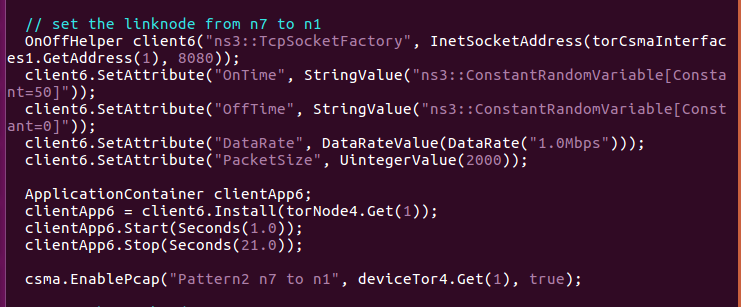
1. 设置节点n5到n1的通信链路



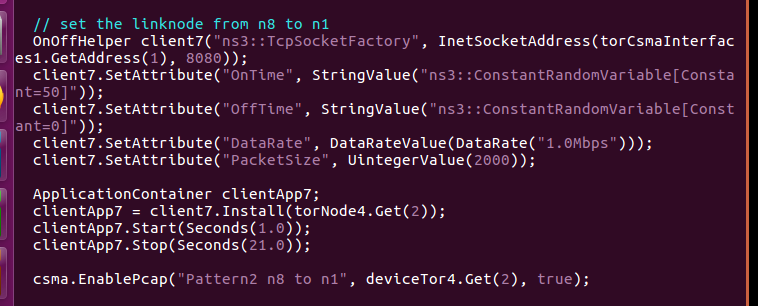
1. 设置节点n6到n1的通信链路



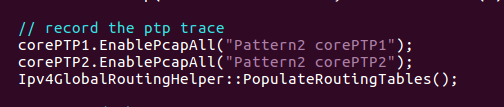
1. 设置节点n7到n1的通信链路



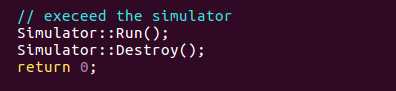
1. 设置节点n8到n1的通信链路



1. 记录PPP网络的trace

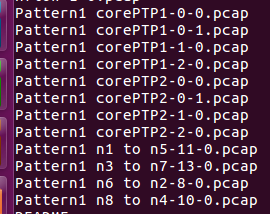


1. 执行仿真



1. 测试网络
2. Pattern1模式的测试

运行pattren1.cc，可输出12个文件， 文件分别如下所示：



其中的 Pattern1 n1 to n5 -11-0.pcap, Pattern1 n3 to n7-13-0.pcap, Pattern1 n6 to n2-8-0.pcap, Pattern1 n8 to n4-1-0.pcap是我们想要分析的文件，利用wireshark可以得到以下的throughput和RRT图像。

* 1. 节点n1到n5的throughput和RRT

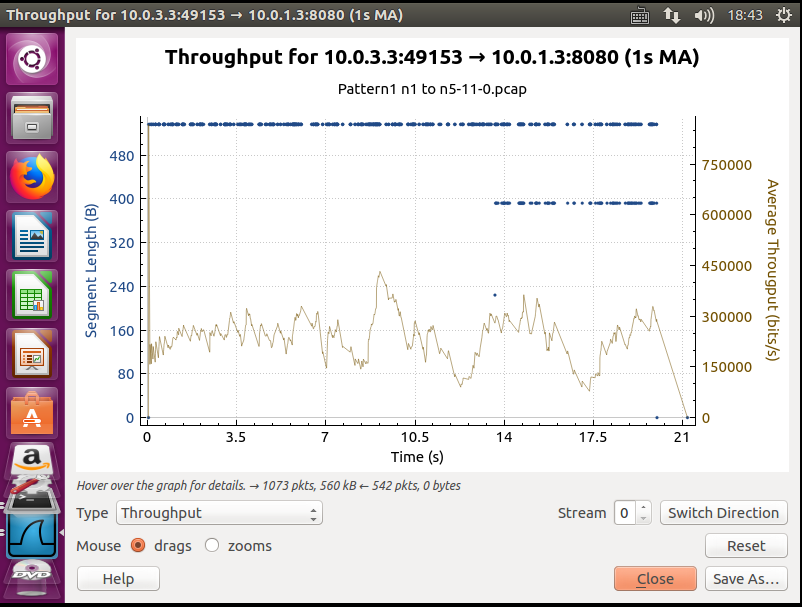


图6.1 n1到n5的throughput

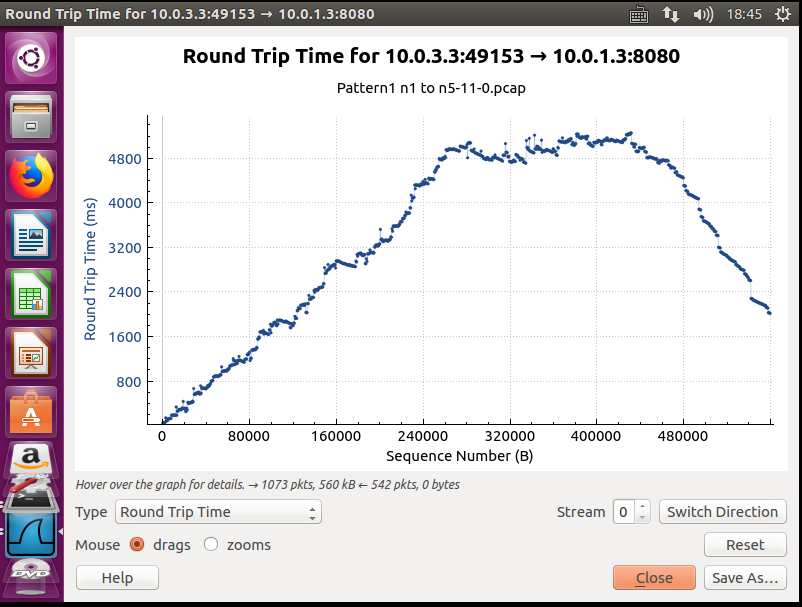


图6.2 n1到n5的RRT

* 1. 节点n6到n2的throughput和RRT

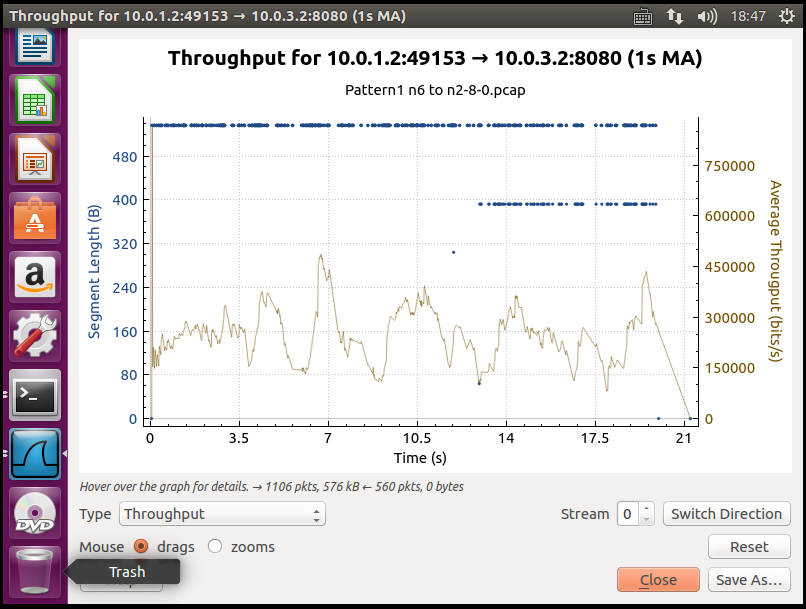


图6.3 n2到n6的throughput

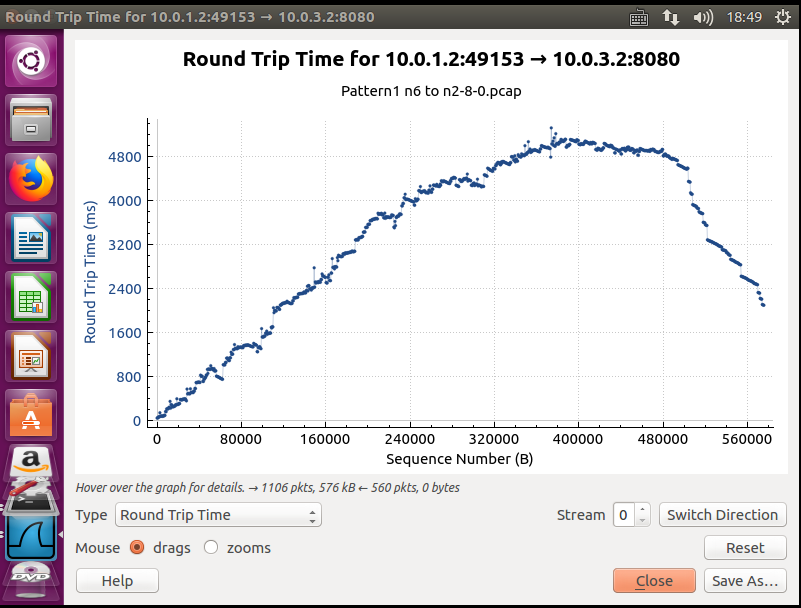


图6.4 n6到n2的RRT

* 1. 节点n3到n7的throughput和RRT

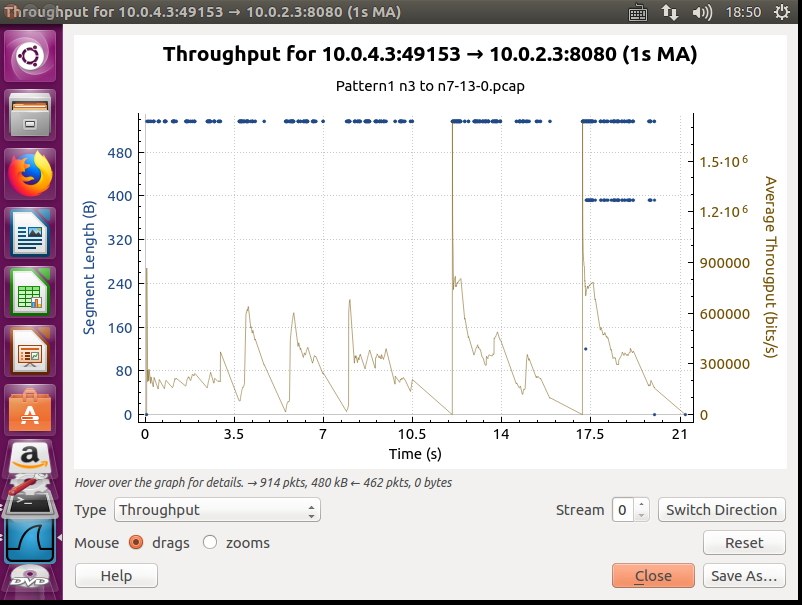


图6.5 n3到n7的throughput

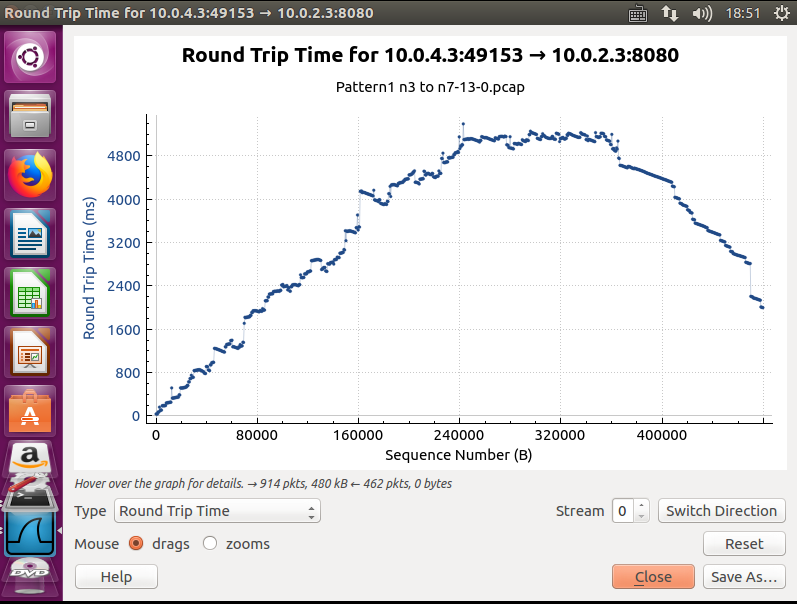


图6.6 n3到n7的RRT

* 1. 节点n8到n4的throughput和RRT

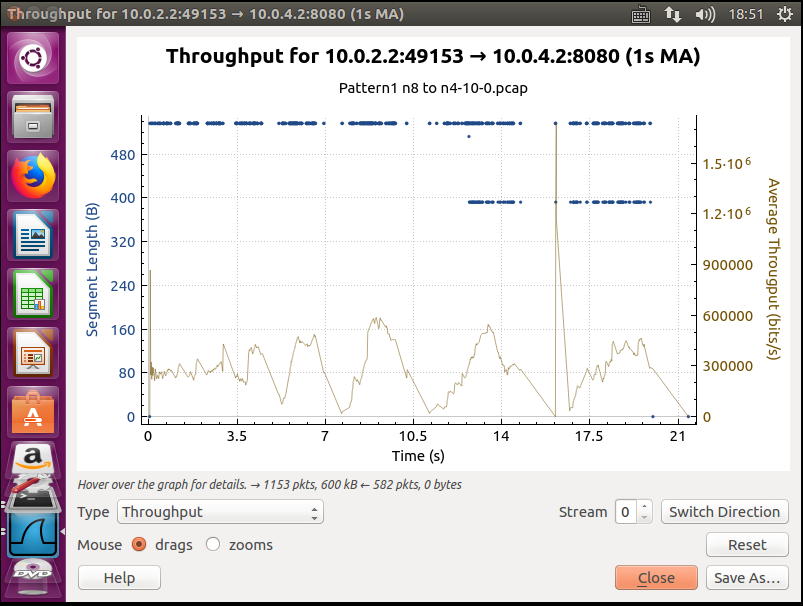


图6.7 n8到n4的throughput

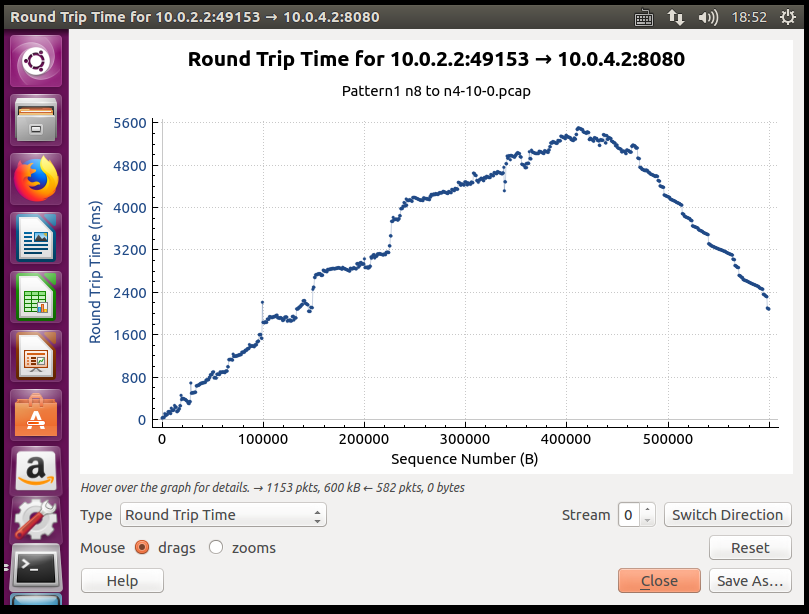
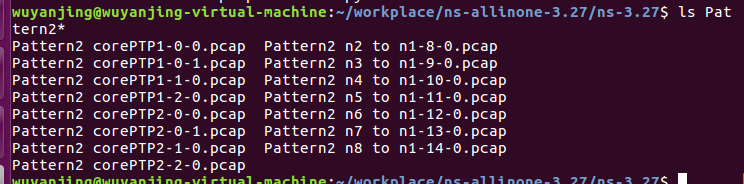


图6.8 n8到n4的RRT

* 1. Pattern1网络分析：
     1. 从数据包的流向上看，数据包是按照之前预想的方式进行发送和接收，所以数据包流向是正常的。
     2. 从TCP throughput上看四条链路的吞吐量变化情况较大，但是从观察上可以看出四条链路的RRT是差不多的。说明四条链路之间是存在着竞争的，并且该竞争是比较公平的。

1. Pattern2模式的测试

运行pattren2.cc，可输出14个文件， 文件分别如下所示：



其中的 Pattern2 n2 to n1 -8-0.pcap, Pattern2 n3 to n1-9-0.pcap, Pattern2 n4 to n1-10-0.pcap, Pattern2 n5 to n1-11-0.pcap, Pattern2 n6 to n1-12-0.pcap, Pattern2 n7 to n1-13-0.pcap, Pattern2 n8 to n1-14.0.pcap是我们想要分析的文件，利用wireshark可以得到以下的throughput和RRT图像。

1. 节点n2到n1的throughput和RRT

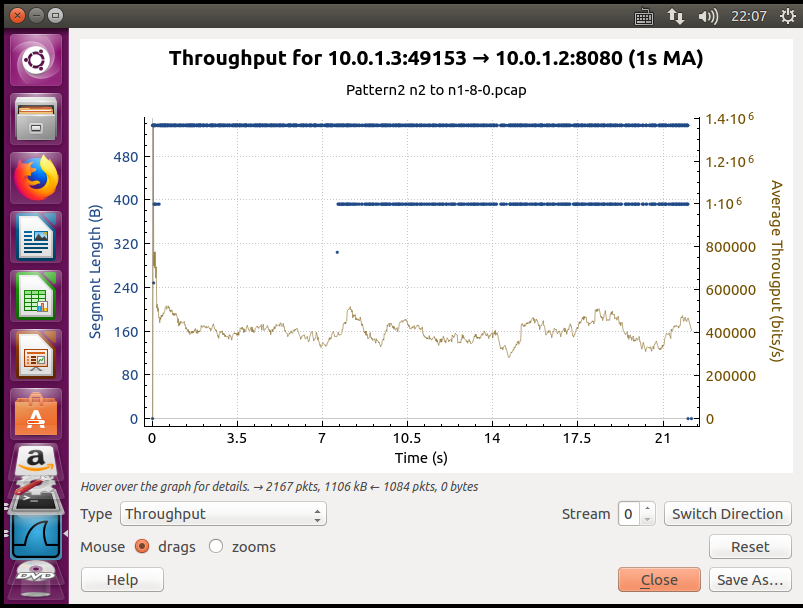


图7.1 n2到n1的throughput

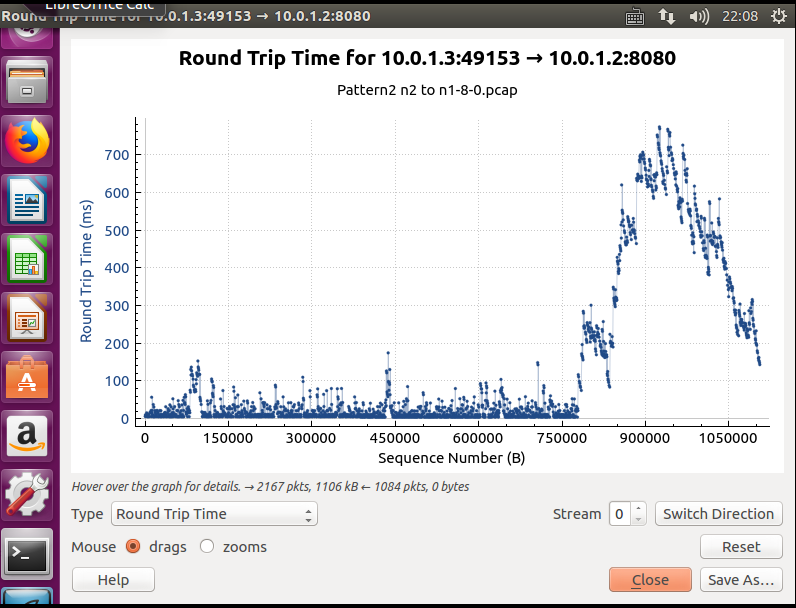


图7.2 n2到n1的RRT

1. 节点n3到n1的throughput和RRT

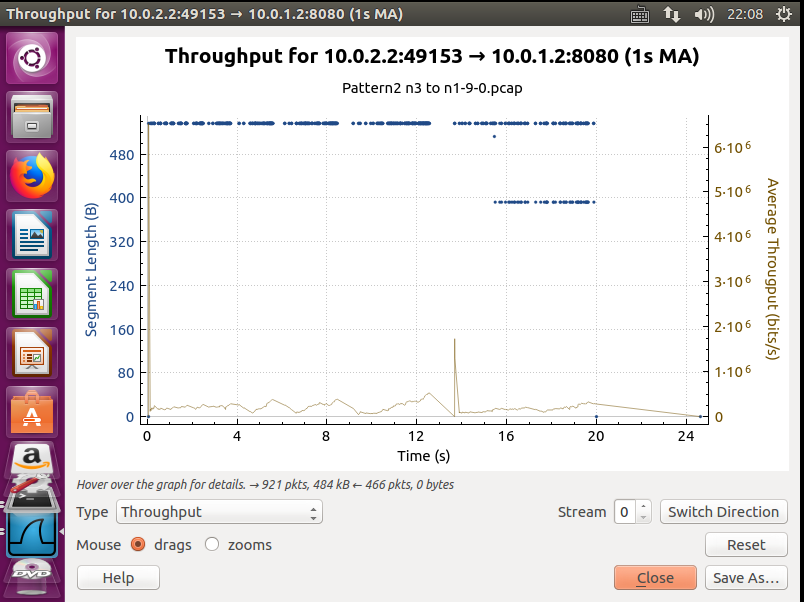


图7.3 n3到n1的throughput

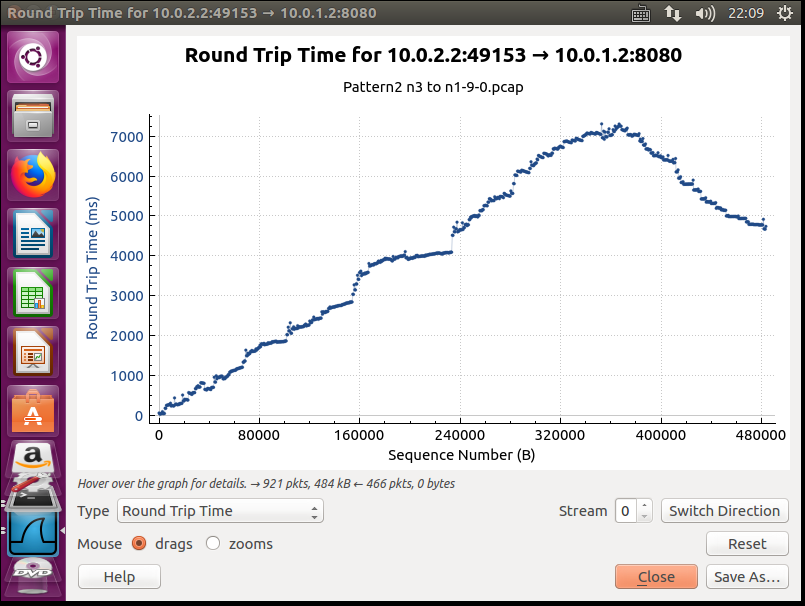


图7.4 n3到n1的RRT

1. 节点n4到n1的throughput和RRT

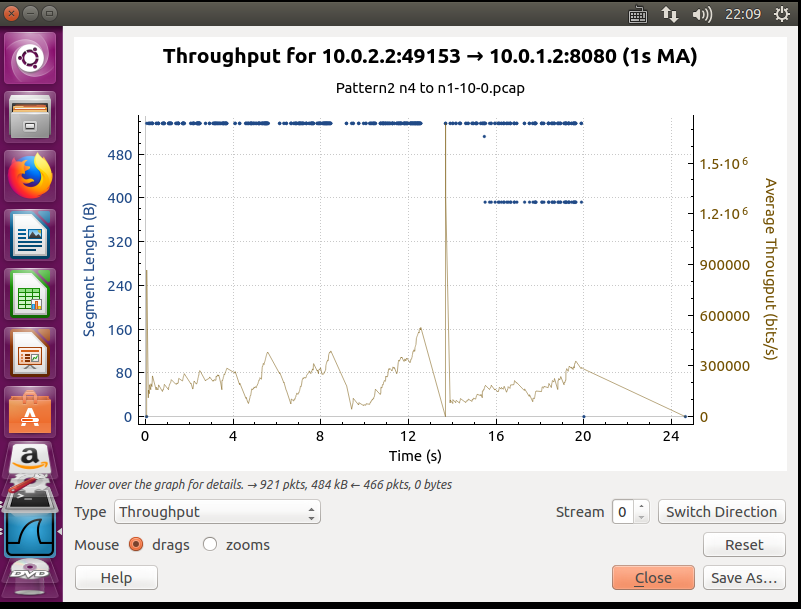


图7.5 n4到n1的throughput

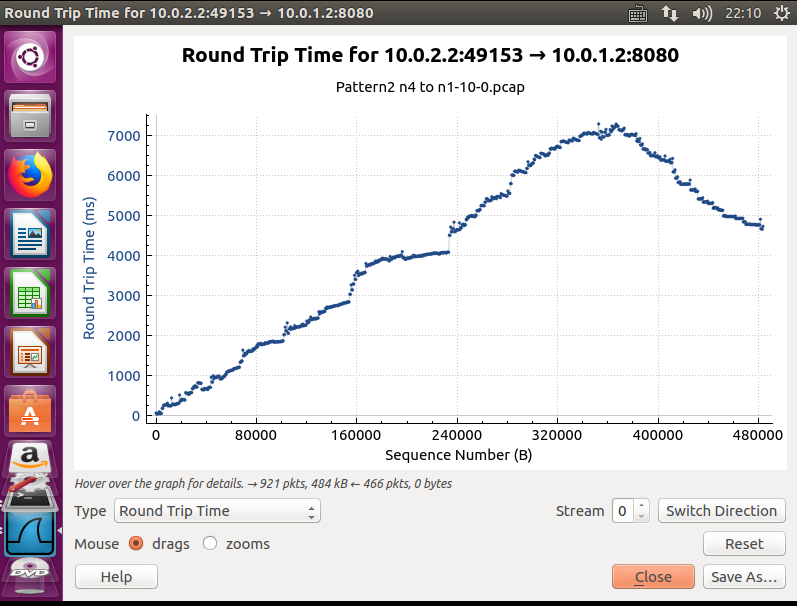


图7.6 n4到n1的RRT

1. 节点n5到n1的throughput和RRT

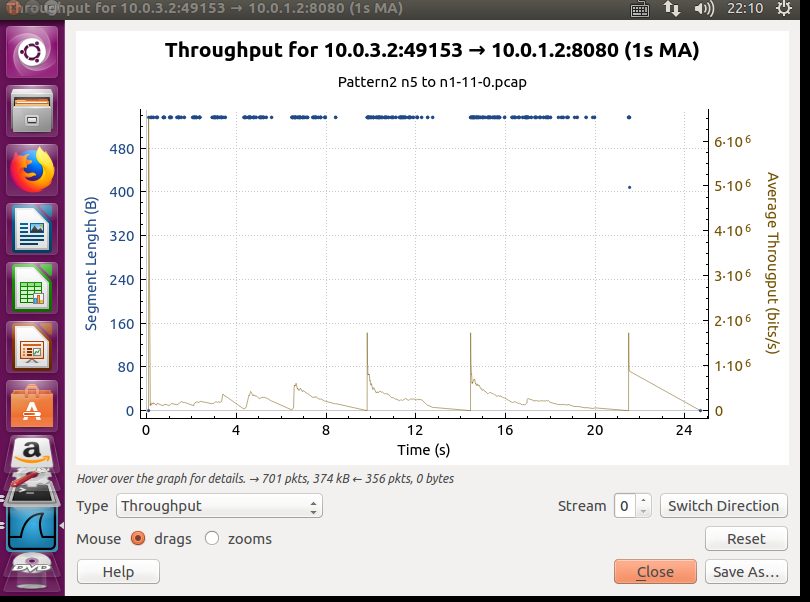


图7.7 n5到n1的throughput

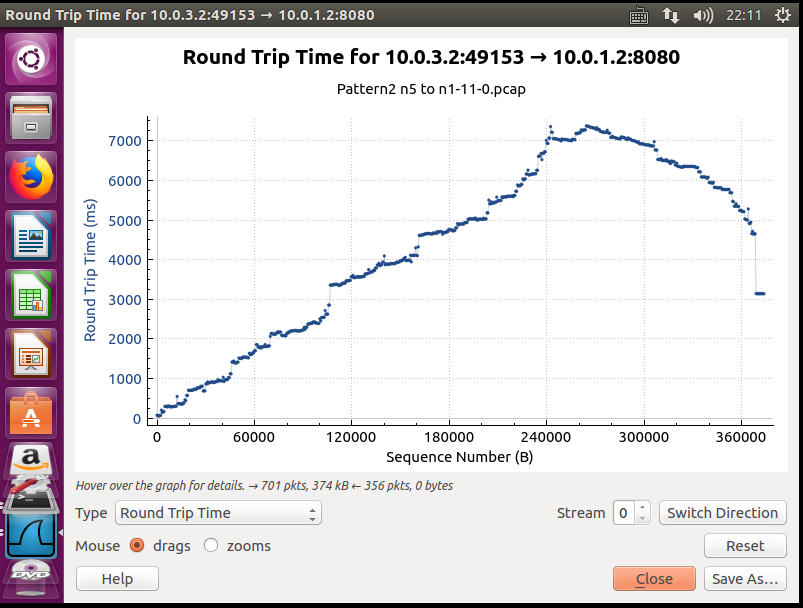


图7.8 n5到n1的RRT

1. 节点n6到n1的throughput和RRT

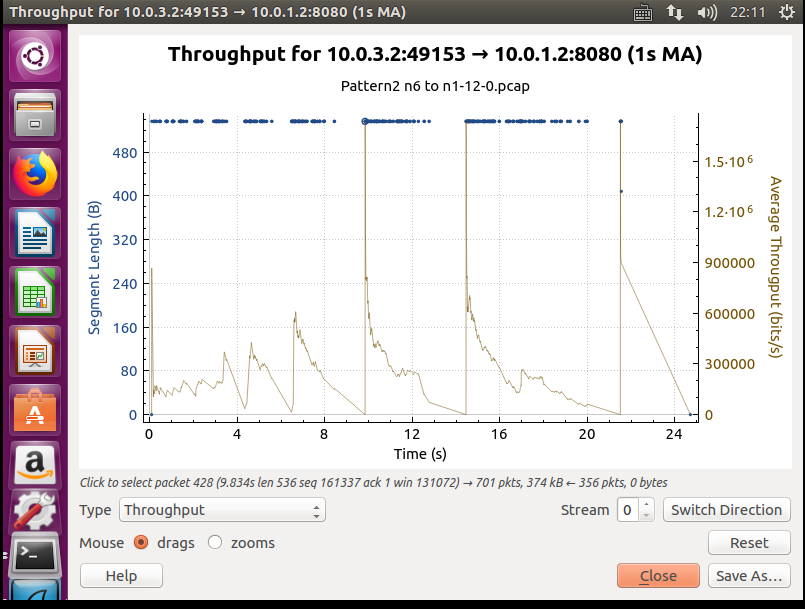


图7.9 n6到n1的throughput

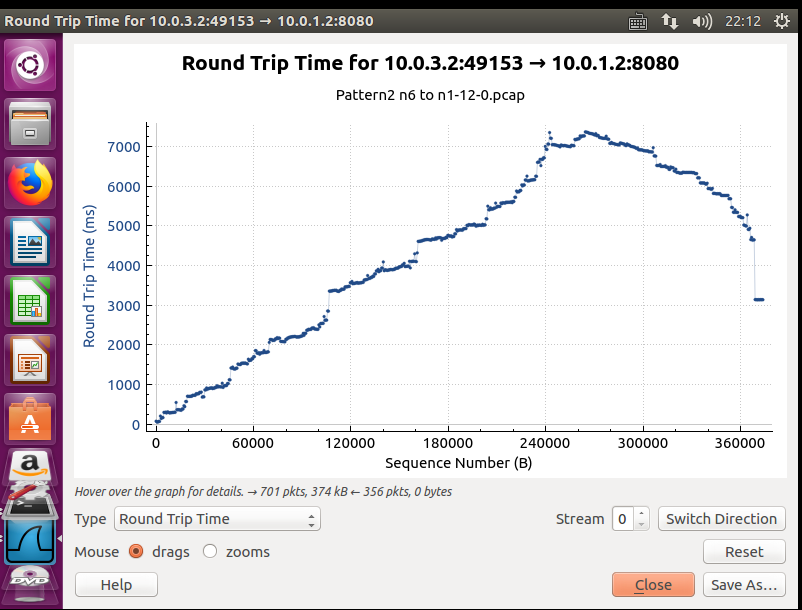


图7.10 n6到n1的RRT

1. 节点n7到n1的throughput和RRT

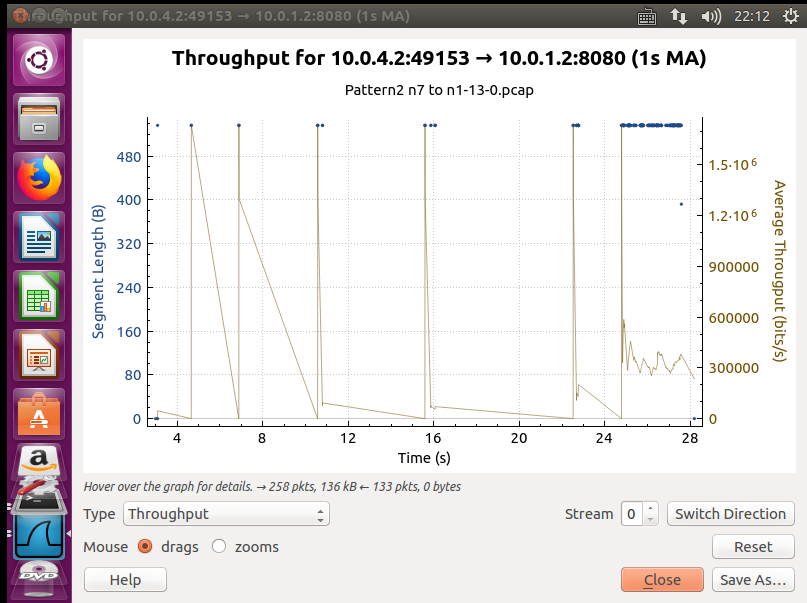


图7.11 n7到n1的throughput

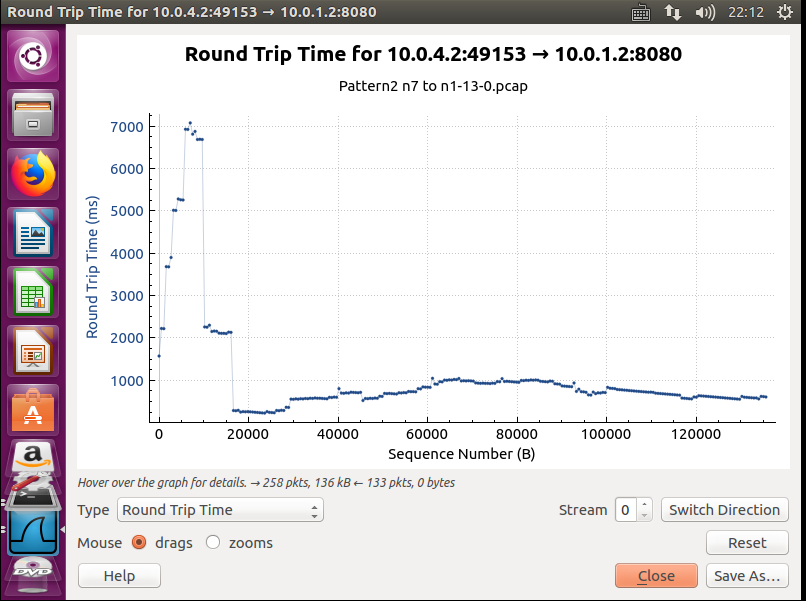


图7.12 n7到n1的RRT

1. 节点n8到n1的throughput和RRT



图7.13 n8到n1的throughput

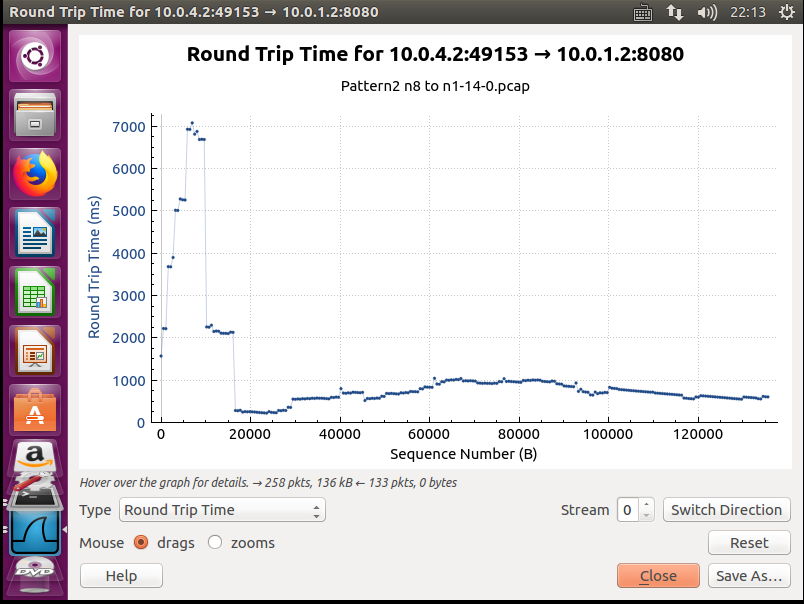
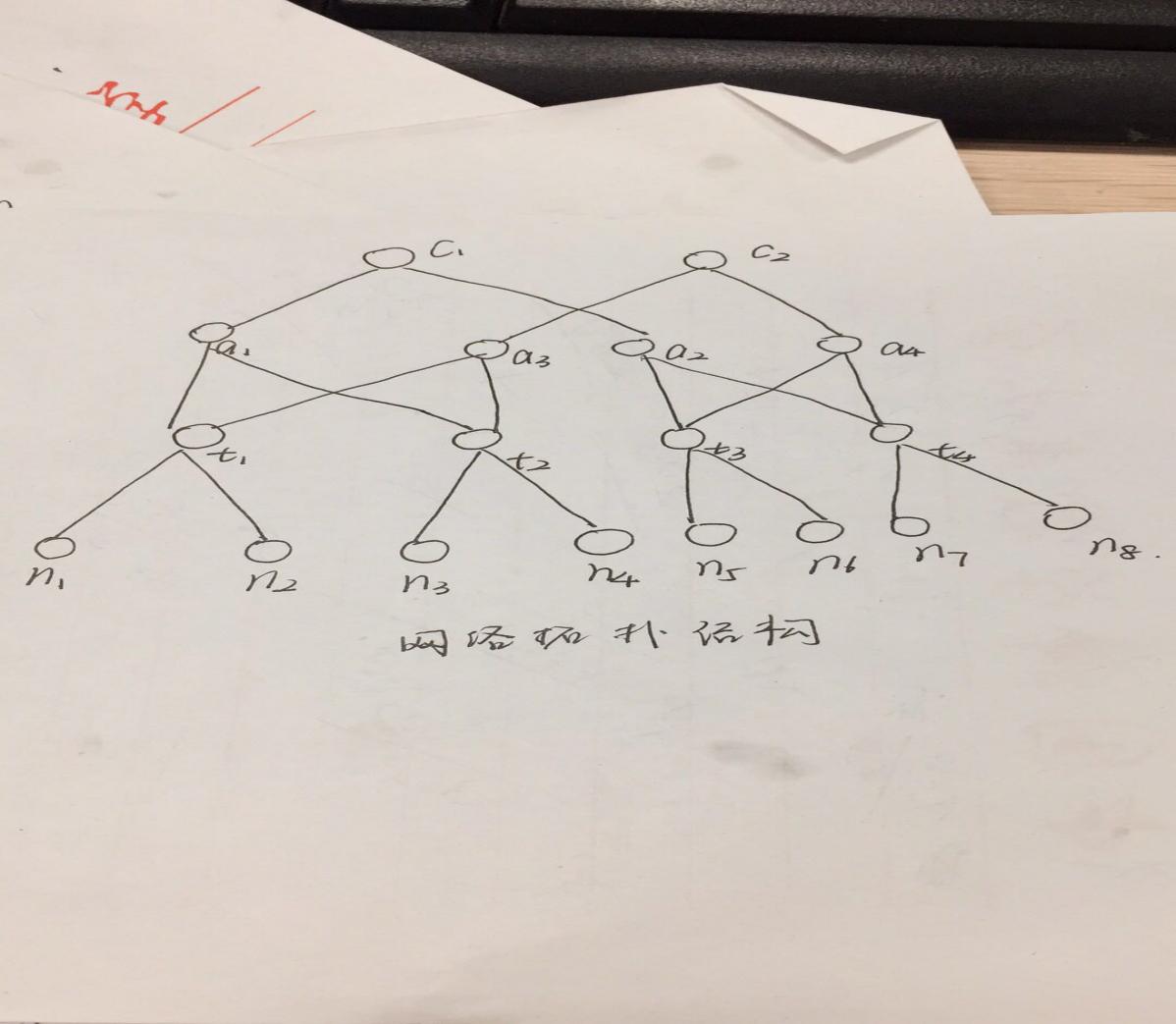
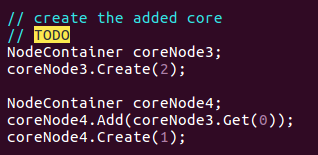


图7.14 n8到n1的RRT

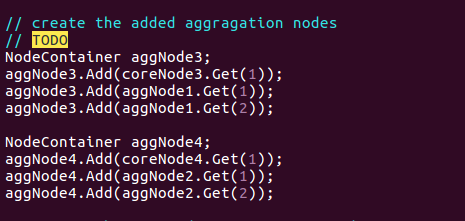
1. Pattern2的网络分析
   1. 从数据包的流向上看，数据包是按照先前预想的进行发送和接收，所以数据包流向是正常的
   2. 从TCP throughput来分析，可以看出四条链路的吞吐量的变化情况较大，且有些地方的吞吐量会接近链路的物理上界，但是从观察可以看出四条链路的平均带宽是差不大的。说明四条链路之间虽然存在竞争，但是该竞争是较为公平的。
2. 改善网络

要改善此网络主要从两个方面入手，首先是增加网络的core节点，其次是将网络的路由选择方式变为ECMP。代码保存在Pattern3.cc文件。以下是它的网络拓扑图：  


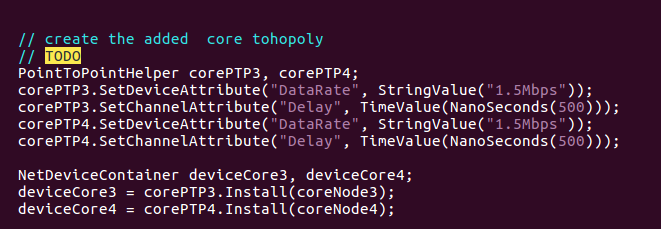
1. 在网络中增加一个core节点和两个aggregation节点
   1. 增加一个core节点



* 1. 增加两个aggregation节点



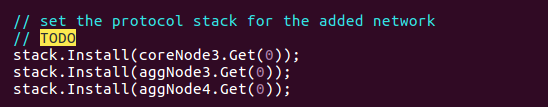
* 1. 增加core节点的拓扑结构



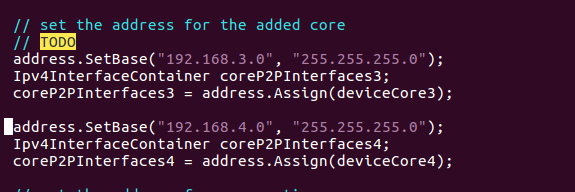
* 1. 增加aggregation节点的拓扑结构



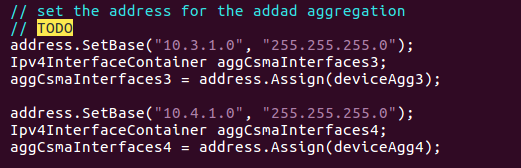
* 1. 为增加的节点设置网络协议



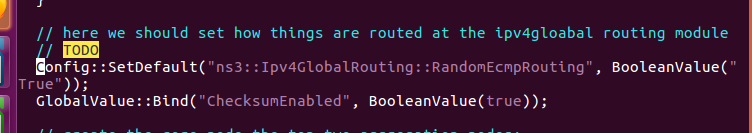
* 1. 为增加的core节点设置地址



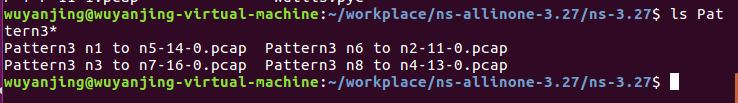
* 1. 为增加的aggregation节点设置地址



1. 采用ECMP的方式



1. 运行Pattern3.cc可以得到以下的文件：
   1. 运行程序之后会有以下的文件：



* 1. n1到n5的throughout和RRT

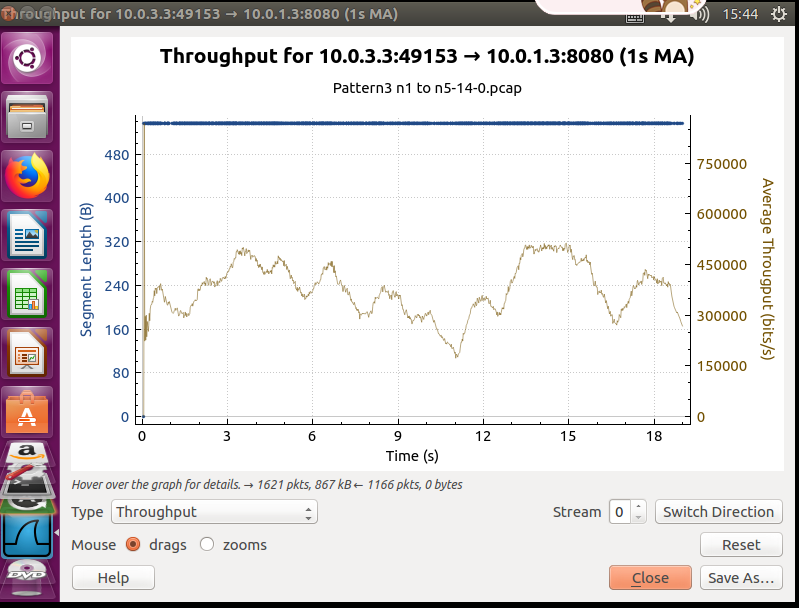


图8.1 n1到n5的throughput

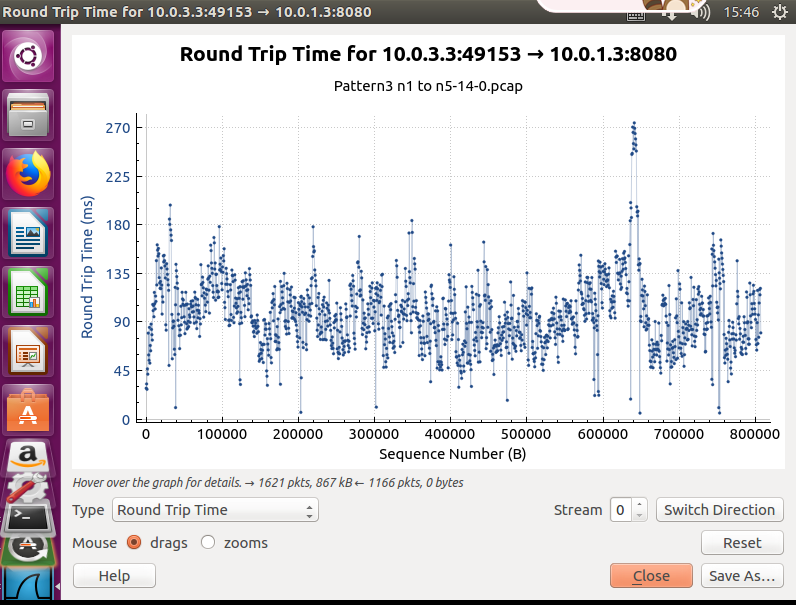


图8.2 n1到n5的RRT

* 1. n2到n6的throughout和RRT

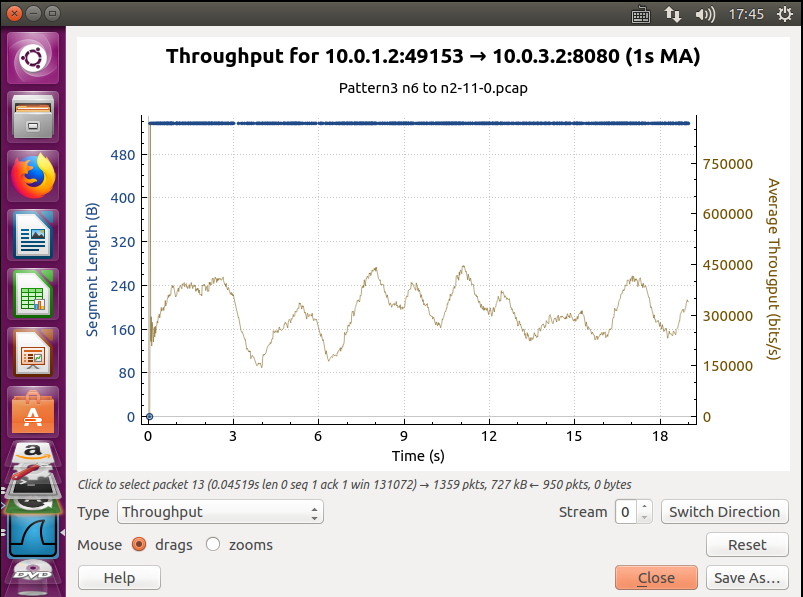


图8.3 n2到n6的throughout

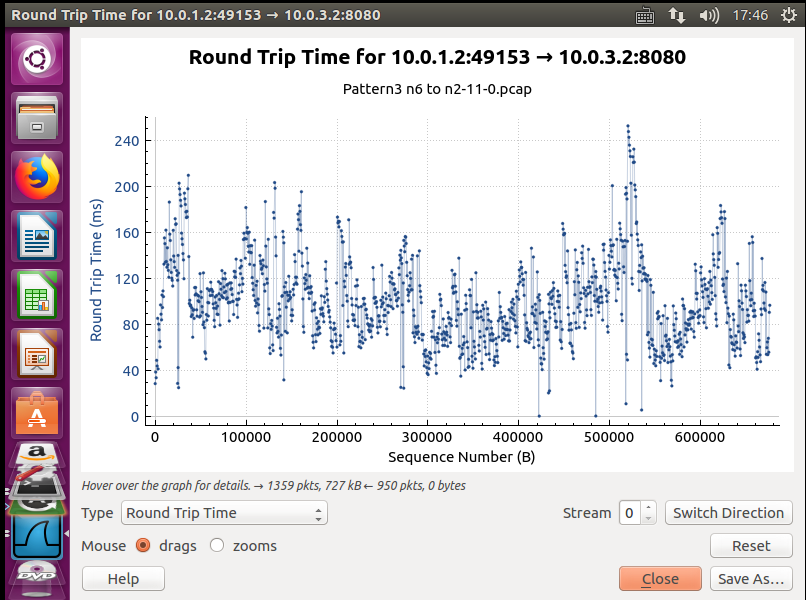


图8.4 n2到n6的RRT

* 1. n3到n7的throughput和RRT

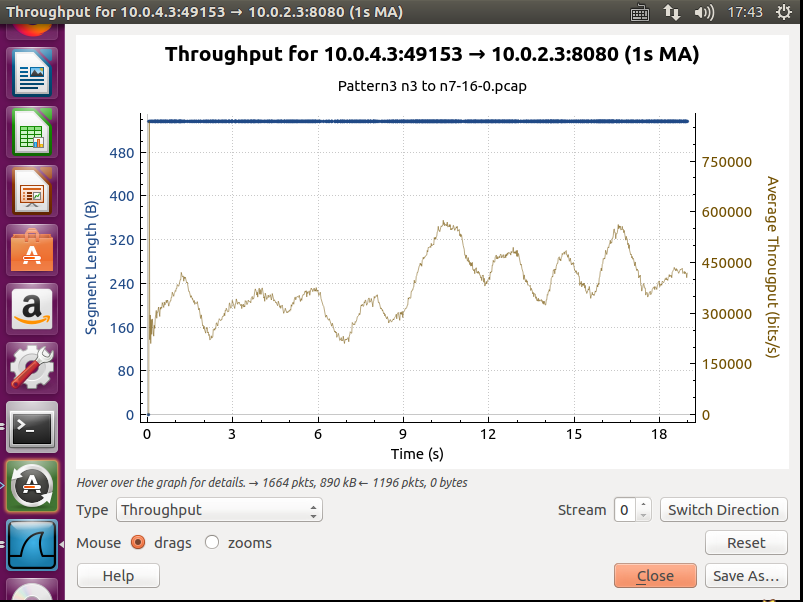


图8.5 n3到n7的throughout

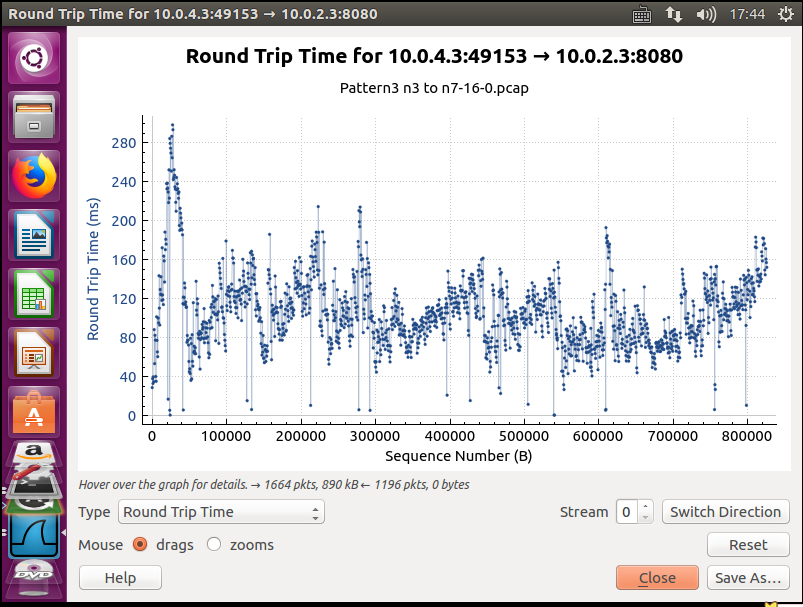


图8.6 n3到n7的RRT

* 1. n4到n8的throughout和RRT

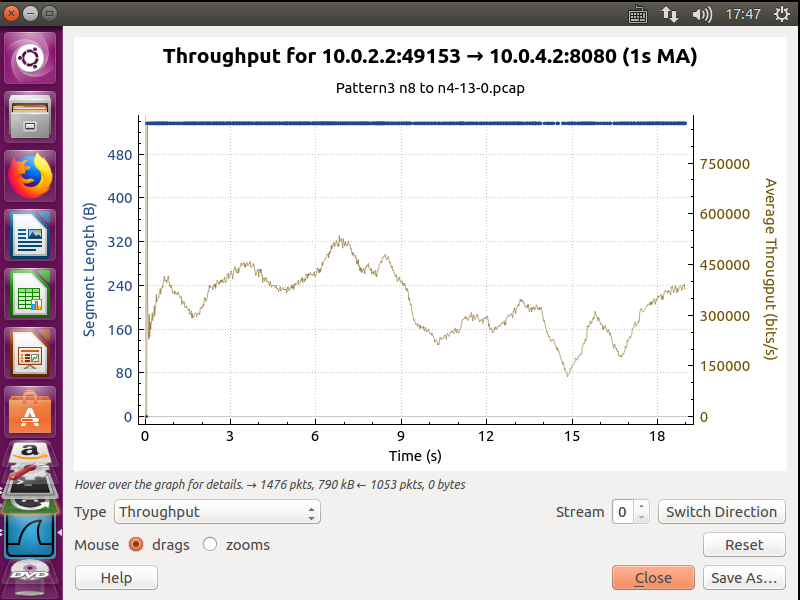


图8.7 n4到n8的throughout

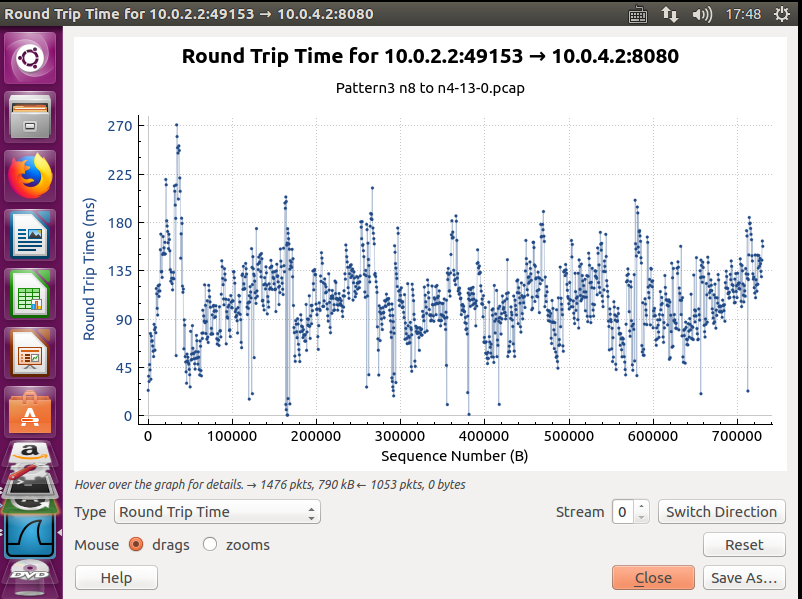


图8.8 n4到n8的RRT

1. 分析
   1. 从数据包的流向上看，数据包是按照之前预想的方式进行发送和接收，所以数据包流向是正常的。
   2. 从TCP throughput上看四条链路的吞吐量变化很小，从观察上可以看出四条链路的RRT是差不多的。说明四条链路之间是存在着竞争的，并且该竞争是比较公平的。
   3. 将改善后的网络和之前的网络进行对比，可以发现，改善之后的网络更加的稳定，竞争更弱。
2. 实验总结
3. 通过这次实验，我知道了应该如何利用ns3来仿真一个计算机网络。
4. 通过对fat tree网络的编码，我更加深刻的认识到了fat tree网络的特点。
5. 还学会了利用wireshark软件进行抓包，并利用wiresshark进行throughput和RRT的分析。
6. 当一个网络不够好的时候，我们可以通过适当的增加core节点和改变路由选择路径为ecmp的方式来使网络得到改善。
7. 原始的ns3的ecmp路由协议的限制太强，因此为了确保实验的成功，我们需要修改src/internet/model/global-route-manager-impl.cc文件。