计算机网络实验报告

--ns3网络模拟器篇

学校：哈尔滨工业大学

班级：11403202

学号：1140320206

姓名：霍峻杰

实验3.1

一、实验题目:

实验4.1：RTT与cwnd

运行Basic1.cc观察RTT与cwnd间的关系。运行多次模拟实验，每次采用不同的链路延迟（delayRB参数），绘图比较，给出分析

实验4.2：带宽与cwnd

参考4.1，分析不同带宽下cwnd的变化

需要修改Basic1.cc，使得可以设置带宽(bottleneckBW)

实验4.3：TCP格斗赛

通过模拟实验，分析各种TCP算法的cwnd：

Tahoe vs. Reno, Reno vs. Reno,

采用以下网络拓扑

A------\ /------B

R1---R2

C------/ \------D

A,C--R1 : 10 Mbps / 10 ms delay

R1--R2 : 500Kbps / 50 ms delay

R2--B,D : 10 Mbps / 10 ms delay

queue at R1: size 10

queue at R2: size 10

执行两个TCP流，A和B之间，C和D之间，共享同一个瓶颈R1---R2

绘制两个流cwnd的时间序列图，并分析

二、实验原理：

安装：

* ns3官网：https://www.nsnam.org
* 下载ns3：https://www.nsnam.org/release/ns-allinone-3.21.tar.bz2
* 进入ns-3.21目录
* ./waf configure --enable-examples --enable-tests
* 在ns-allinone-3.21下
* ./build.py
* 将basic1.cc拷贝到/home/ubuntu/ns-allinone-3.21/ns-3.21/scratch/下
* 添加以下代码到 ~/.bashrc，并执行 . .bashrc

NS3DIR='/home/lovebear96/acg/n3/ns-allinone-3.21/ns-3.21’

function waff {

CWD="$PWD"

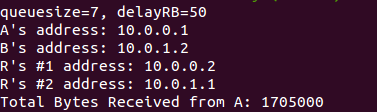
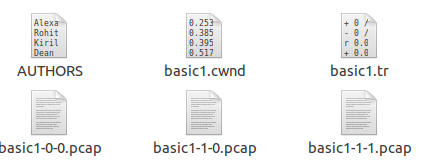
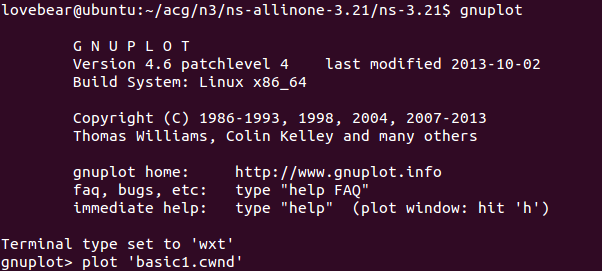
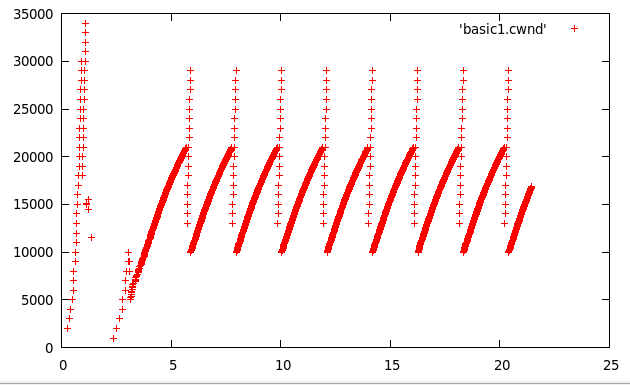
cd $NS3DIR >/dev/null

./waf --cwd="$CWD" $\*

cd - >/dev/null

}

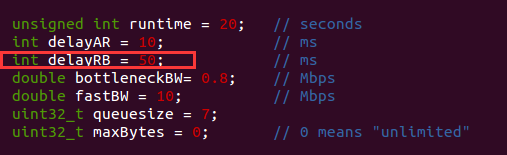
Source ./bashrc 使之生效

* 建立一个新目录，运行 waff --run basic1，会得到basic1.cwnd等文件
* 
* 
* 用gnuplot画图（需安装gnuplot-x11）: plot 'basic1.cwnd
* Gnuplot测试
* 
* 测试结果
* 
* 测试阶段完毕，开始实验

三、实验结果:

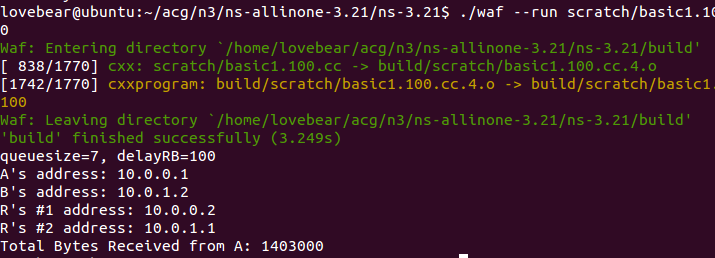
实验4.1

修改下面标红的delayRB参数值为50，100，150，200，250



生成对应的cwnd文件，画图（一张图中）

注意修改代码中的文件名，否则会一直覆盖生成文件

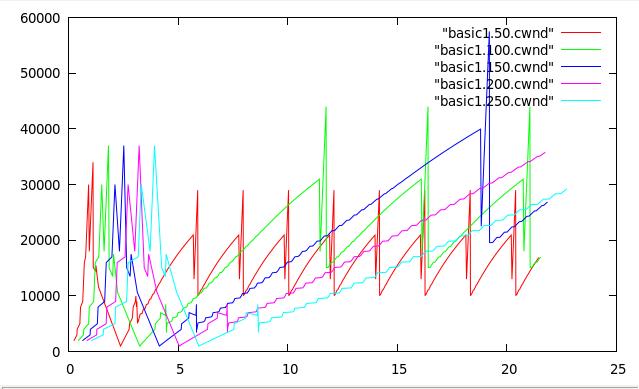


重复生成basic1.50.cwnd,basic1.100.cwnd, 150,200,250…….

生成5个cwnd文件后，用gnuplot画图



画图结果



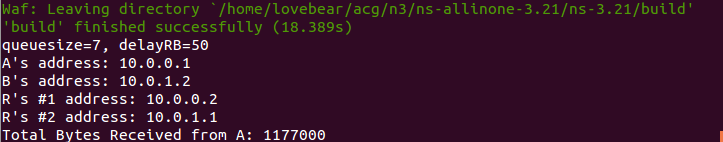
实验结果分析

每条曲线的最开始一段都先上升在下降，然后状态稳定为缓慢上升，突然下降。这是慢启动后拥塞避免，当发送方连续收到3个重复的确认后，进行快恢复，将ssthresh门限减半。

在带宽一定的情况下，链路延迟越大，rtt越大，cwnd越大，cwnd变化时间越长。

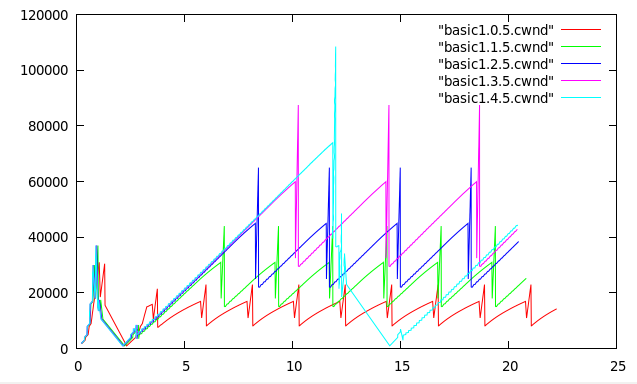
实验4.2

仿照实验4.1，生成basic1.0.5.cwnd,basic1.1.5.cwnd,2.5,3.5,4.5……..



Gnuplot画图





实验结果分析

传输延迟一定的情况下，rtt也一定，带宽越大，cwnd也越大，但是带宽过大会导致cwnd指数递减，传输效率变低，所以带宽设置不能太小，也不能太大。

实验4.3.1 Reno VS Reno

两种算法都是用Reno算法，分别输出到两个文件：



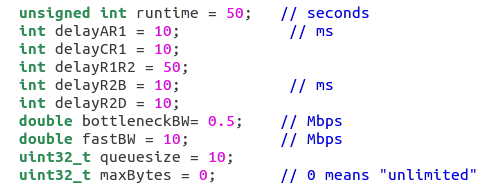
按题目要求设置好参数

采用以下网络拓扑

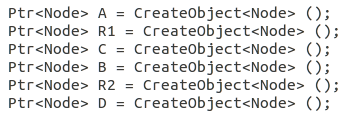
A------\ /------B

R1---R2

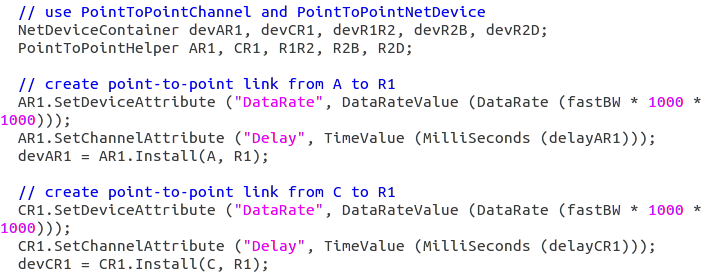
C------/ \------D



设置节点

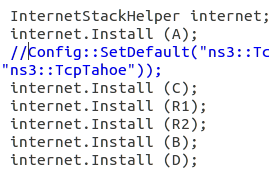


设置节点连接和信道

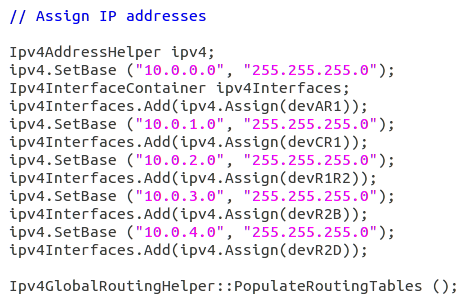


……

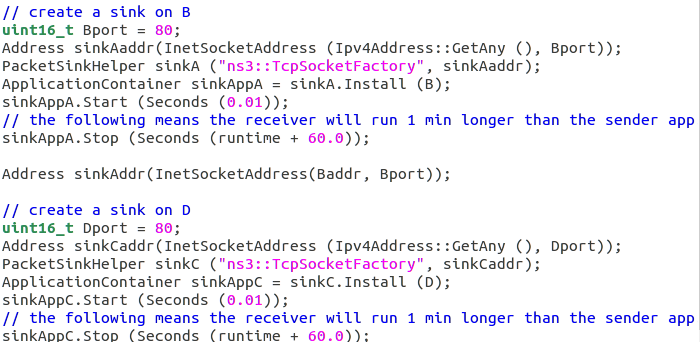
配置协议为Reno



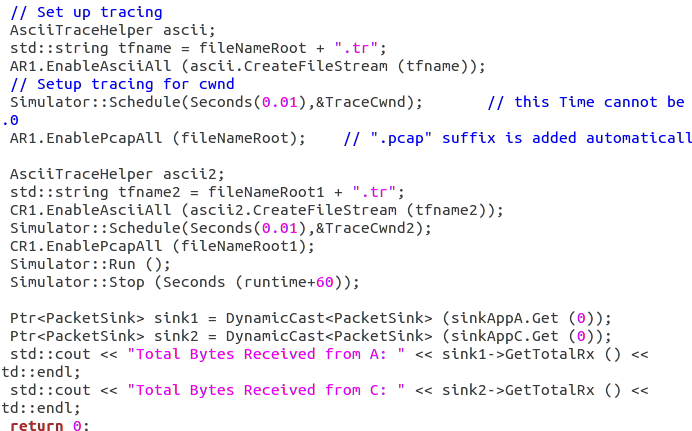
分配ip地址



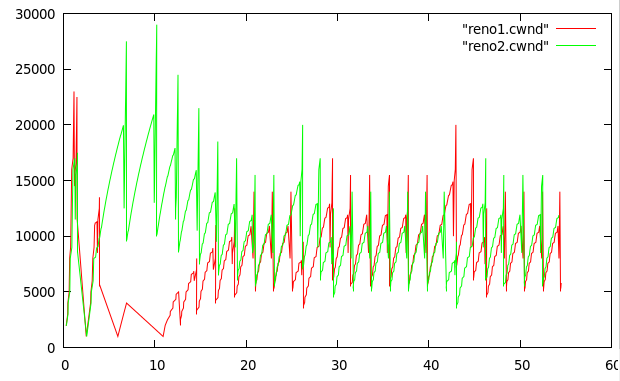
B, D节点上建立sink，A, C节点为源



发送，接收设置



绘图结果：

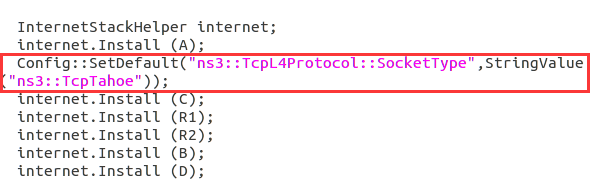


实验结果分析：

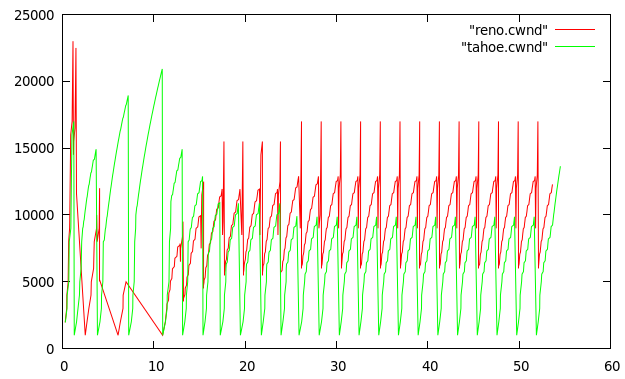
Reno1为A的传输cwnd，Reno2为C的传输cwnd，两者一开始的cwnd值会出现剧烈波动，但过一段时间后，两者的变化趋于稳定，且表现出趋于一致的现象，这表明都使用Reno算法的不同tcp传输流，其竞争能力相当，平均分配，比较公平，实验结果和预期的现象一致。

实验4.3.2 Tahoe vs. Reno

只需要在配置协议的时候加上一句话，将C的TCP传输设置为Tahoe即可



绘图结果：



实验结果分析：

一开始tahoe会占据比较主动的优势，但经过一段时间后，两者会趋于稳定状态，而reno的cwnd始终比tahoe高，而且上下变化幅度也较小，由此可以看出reno的性能明显好于tahoe。

实验总结：

一开始做实验的时候，自己按照原来的basic1瞎改，始终reno2不出数据，后来问学长，自己研究了一下说明文档，发现了一些非常低级的错误。这个实验，我了解了reno和tahoe的基本原理，通过实验，得到了rtt，带宽与cwnd的关系变化，也研究了两个算法的竞争关系和结果，得到了reno的更优结论。